دانشگاه آزاد اسلامی واحد تهران شمال – دانشکده برق و کامپیوتر
کارشناسی ارشد مهندسی نرم‌افزار
درس: ارزیابی کارایی سیستم‌های کامپیوتری

عنوان

گزارش ارزیابی کارایی سیستم‌های اینترنت اشیا پزشکی و اینترنت اشیا براساس مجموعه داده‌های CICIoMT2024 و مدلینگ ریاضیاتی

نگارش
علیرضا سلطانی نشان

استاد راهنما
دکتر مهدی امینیان

۲۶ دی ۱۴۰۳

فهرست مطالب

فهرست تصاویر

فهرست جداول

مجوز

۱ دیتاست حملات سایبری دستگاه‌های اینترنت اشیا پزشکی CICIoMT2024

مهم‌ترین انگیزه برای توسعه این پژوهش [۱] وجود کمبود در داده‌های موجود ارزیابی کارایی تجهیزات اینترنت اشیا پزشکی و پیشرفت امنیتی تمام شبکه‌هایی که در خصوص جریان‌های داده‌ای و پردازش داده‌های پزشکی کار می‌کنند، می‌باشد بخصوص برای دستگاه‌های اینترنت اشیا پزشکی به دلیل اطلاعات حیاتی‌ای که می‌توان به واسطه آن‌ها از بیماران با بیماری‌های مختلف مانیتور و دریافت کرد. نتیجه این پژوهش دیتاستی از تمامی حملاتی مهم می‌باشد که روی دستگاه‌های IoMT انجام داده‌اند تا از طریق دیتاست بدست آمده بتوان به صورت خودکار با استفاده از مدل‌های یادگیری ماشین وجود هر گونه حمله در سیستم‌های IoMT را تشخیص داد و از بروز آن جلوگیری کرد.

۱.۱ اهداف اصلی پژوهش

۱. کمک به پژوهشگران برای ایجاد سیستم‌های بهداشت و درمان ایمن با استفاده از سیستم‌های خودکار یادگیری ماشین و یادگیری عمیق.

۲. ارائه بنجمارک‌های واقعی برای ارزیابی و توسعه راهکارهای امنیتی

۳. فراتر از شبیه‌سازی حملات، محققان تمام فرایندها و چرخه حیات دستگاه‌های IoMT را از ورود به شبکه تا خروج از طریق پروفایل‌های امنیتی رصد می‌کنند و به سیستم‌های خودکار مانند سیستم‌های طبقه‌بندی کننده حملات، اجازه می‌دهد تا ناهنجاری‌های امنیتی داخل سیستم‌های بهداشت و درمان را شناسایی کنند.

۴. با دیتاست بدست آمده [۲] که به صورت آزاد در دسترس عموم می‌باشد محققان راه‌های هوشمندانه‌ای را برای طبقه‌بندی حملات سایبری فراهم کرده‌اند.

۲.۱ پروتکل‌های استفاده شده

برای حملات سایبری، محققان از پروتکل‌های پر استفاده در حوزه IoT استفاده کرده‌اند که عبارت‌اند از:

۱. Wi-Fi

۲. MQTT

۳. Bluetooth

این پژوهش در دسته‌بندی Predictive models برای جلوگیری از حملات و حتی فالت‌های نرم‌افزار می‌تواند قرار گیرد.

۳.۱ دسته‌بندی حملات سایبری

در این پژوهش ۱۸ حمله سایبری متفاوت روی ۴۰ دستگاه IoMT صورت گرفته تا هم بتوانند داده‌های مربوط به حملات را به صورت منظم و مهندسی شده فراهم کنند و هم عملکرد دستگاه‌های IoMT مورد نظر را با حملات سایبری مورد ارزیابی قرار دهند.
دسته‌بندی حملات:

- DoS (Denial of Service)
- DDoS (Distributed Denial of Service)
- Spoofing
- Recon (Reconnaissance)
- MQTT (Message Queuing Telemetry Transport) attacks

این ۱۸ حمله عبارت‌اند از:

- DoS TCP
- DoS ICMP
- DoS SYN
- DoS UDP
- DDoS TCP
- DDoS ICMP
- DDoS SYN
- DDoS UDP
- MQTT Malformed Data
- MQTT DoS Connect flood
- MQTT DoS Publish flood
- MQTT DDoS Connect flood
- MQTT DDoS Publish flood
- ARP Spoofing (Man-in-the-Middle)
- Recon attacks
- Spoofing attacks
- Flooding campaigns (various types)
- Other targeted attacks specific to IoMT protocols

۴.۱ مقایسه CICIoMT2024 با کارهای پیشین

۱.۴.۱ دیتاست ECU-IoHT

این دیتاست [۳] آسیب‌پذیری دستگاه‌های IoT را در محیط‌های مراقبت‌های بهداشت و درمان بررسی کرده است. در این پژوهش دلیل اصلی حملات سایبری به روز شدن تدابیر امنیتی دستگاه‌های واقعی بوده است که در حوزه مراقبت‌های بهداشتی و درمانی توسعه یافته‌اند. این دستگاه از قبیل دستگاه‌های زیر بوده‌اند:

- MySignals
- Temp sensor
- BP sensor
- HR sensor
- Bluettoth and wireless adapter

- Kali and windows laptop

حملات سایبری انجام شده در این پرژوهش:

- ARP spoofing
- DoS
- Smart and injection

۲.۴.۱ حملات مخصوص روی فناوری بلوتوث

در مطالعه دیگر [۴، ۵] دیتاستی فراهم شده است که نشان می‌دهد حملات سایبری روی دستگاه‌های IoMT در توپولوژی بلوتوث به چه شکلی انجام می‌شوند. در این پژوهش بحث‌های تخصصی زیادی در رابطه با جنبه‌های استفاده از این پروتکل شده است به گونه‌ای که اتصال چندین دستگاه به یک منبع، و حملات مختلفی را روی این فناوری اجرا کرده‌اند. خروجی این حملات به گونه‌ای بوده است که می‌توان عملکرد دستگاه‌ها را با استفاده از الگوریتم‌های یادگیری ماشین مانند SVM (Support Vector Machine)، K-Means و شبکه‌های عصبی عمیق ارزیابی کرد.

۳.۴.۱ شبیه‌سازی ترافیک با ابزار IoTFlock

در پژوهش دیگر [۶] محققان با استفاده از IoTFlock، ابزاری برای شبیه‌سازی ترافیک شبکه‌ای در سیستم‌های IoT و شناسایی نقاط ضعف امنیتی، به‌ویژه در حوزه سلامت، ارائه کرده‌اند. این ابزار به پژوهشگران کمک می‌کند تا راه‌حل‌های امنیتی قوی‌تری برای مقابله با حملات سایبری توسعه دهند. ابزار IoTFlock شبیه‌سازی ترافیک عادی شبکه و ترافیک مخرب را انجام می‌دهد. حملاتی که در این پژوهش انجام شده‌اند عبارت‌اند از:

- DDoS
- Brute force
- SlowITE
- MQTT Publish Flood

حمله SlowITE حمله هوشمندانه‌ای است که منابع سیستم را به آرامی خالی می‌کند به گونه‌ای که هیچ چیز مشکوک به نظر نرسید.

۵.۱ فرایندها

۱.۵.۱ آزمایشگاه IoT

برای انجام این پژوهش، محققان به دنبال یک آزمایشگاه مجهز به دستگاه‌های IoT بودند تا بتوانند آزمایش‌های مورد نظر را بر روی طیف عظیمی از دستگاه‌های پر استفاده انجام دهند. این امر برای محققان کمی دشوار بود زیرا تنها به دستگاه‌های IoT نیاز نداشتند بلکه وجود تجهیزاتی مانند روترها، اکسس‌پوینت‌ها، سویچ‌ها و تیمی که بتوانند تجهیزات تحت شبکه را پیکربندی و راه‌اندازی کنند، ضروری بود. مشکل بعد علاوه‌بر تامین منابع، راه‌اندازی این دستگاه‌ها در مقیاس بزرگ و هزینه‌های متغیر بود که نیازمند سرمایه‌گذاری بودند تا با وجود اهمیت پژوهش، تجهیزات مورد نظر را تهیه و پشتیبانی کند. از این رو موسسه کانادایی امنیت سایبری یا CIC به صورت داوطلب درخواست محققان را پذیرفت و آزمایشگاه مورد نظر را برای انجام این تحقیقات ایجاد کرد که شامل ده‌ها دستگاه IoT برای اهداف متنوعی مانند مراقبت از بهداشت، خانه هوشمند را به همراه سیستم‌های شبکه‌ای و کیت‌های Arduino و RPi را در اختیار آن‌ها گذاشت. در پی تجهیزات یک تیم فنی متعهد به نگهداری و مدیریت دستگاه‌های IoT و تمام تجهیزات تحت شبکه را به این کار دعوت کردند تا محققان درگیر راه‌اندازی‌ها و پیکربندی‌های شبکه نشوند.

شکل ۱: تجهیزاتی که موسسه کانادایی امنیت سایبری در اختیار محققان قرار داد.

۲.۵.۱ توپولوژی شبکه دستگاه‌های IoMT

توپولوژی که محققان این پژوهش برای انجام آزمایشات خود راه‌اندازی کردند شامل المان‌های زیر بوده است:

- یک iPad جهت کنترل فرایندها و درخواست‌ها
- ۴ عدد RPi که تمامی حملات از طریق آن‌ها انجام می‌شود.
- دستگاه‌های IoMT که به وسیله یک Access point به شبکه متصل هستند. با استفاده از این اتصال دستگاه‌های IoMT می‌توانند به اینترنت دسترسی داشته باشند.
- تبادل داده‌ها بین دستگاه‌های IoMT به گونه‌ای است که ۱۵، ۷ و ۱۴ دستگاه به ترتیب با پروتکل‌های MQTT، Wi-Fi و بلوتوث ارتباط دارند.

۳.۵.۱ تولید ترافیک مخرب

برای تولید ترافیک مخرب روی دستگاه‌های IoMT از سناریوهای واقعی در محیط‌های مراقبت‌های بهداشتی استفاده شده است تا دیتاست کاملاً واقعی باشد. هدف این فرایند ثبت ویژگی‌های ترافیک بدخیم و خراب‌کننده جهت تسهیل توسعه راهکارهای امنیت سایبری برای محیط‌های IoMT می‌باشد. حملاتی که در این مرحله انجام شده عبارت‌اند از:

- DoS
- DDoS
- ARP Spoofing
- Flooding campaings (Wi-Fi devices)

- ICMP
- SYN
- TCP
- UDP

Wi-Fi ۴.۵.۱

برای ایجاد ترافیک در این پروتکل ارتباطی، محققان حملات ARP Spoofing را برای Man in the middle و بسیاری از حملات، TCP، CYN، ICMP و UDP انجام دادند که همگی در دسته‌بندی DoS و DDoS قرار دارند.

MQTT ۵.۵.۱

در طی فعالیت‌ها در این پروتکل سه حمله انجام شده است:

- MQTT Connect Flood: در این حمله درخواست اتصال به Broker با شدت زیاد ارسال می‌شود.
- MQTT Publish Flood: ارسال بسته‌های مختلف به Topicهای مختلف و رندوم را انجام می‌دهد.
- MQTT Malformed Data Attack: ارسال داده‌های نادرست را به Broker برای تجزیه و تحلیل رفتار و جمع‌آوری اطلاعات Topicهای استفاده می‌کند.

در حمله آخر از ابزار MQTTSA استفاده شد و سعی در Sniff و شنود Broker با ارسال بسته‌های مشخص را داشتند تا رفتار دستگاه‌ها را بررسی کنند. در این فرایند حمله، حمله‌کننده نام تمام تاپیک‌های MQTT را که به صورت نهایی منتشر شده‌اند را بدست می‌آورد و سعی می‌کند که به هر تاپیک داده نادرست را ارسال کند. هدف اصلی این حملات بررسی و ارزیابی آسیب‌پذیری دستگاه‌های IoMT در پروتکل MQTT بود. شبیه‌ساز با استفاده از ماشین مجازی VMware که سیستم عامل لینوکس اوبونتو نسخه 18.04.6 راه‌اندازی شده بود و از نسخه IoTFlock GUI که با زبان C++ نوشته شده بود مورد استفاده قرار گرفت.

Bluetooth BLE 5.0 ۶.۵.۱

به طور کلی این ارزیابی برای بررسی مقاومت و تاب‌آوری دستگاه‌ها در برابر تهدیدات امنیتی و اختلالات محیطی انجام شده است.
معمولاً حمله در بستر BLE دو شرط الزامی را به همراه دارد:

۱. اجرای حمله

۲. جمع‌آوری ترافیک شبکه به روش‌ها و تکنیک‌های خاص

فناوری BLE به دلیل طراحی خاص خود که برای مصرف پایین انرژی بهینه شده است، رفتاری متفاوت از پروتکل‌های بی‌سیم یا شبکه‌های معمولی دارد. بنابراین، اجرای حملات یا ضبط ترافیک در این فناوری نیازمند ابزارها و تکنیک‌های متفاوتی است [۷]. در این پژوهش، برای اجرای این حملات از یک تلفن همراه هوشمند که به دستگاه BLE متصل شده، استفاده شده است و سپس فعالیت‌های مخرب با استفاده از کامپیوتر صورت گرفته است.

برای توسعه برنامه‌ای که قادر باشد تمامی بلوتوث‌های موجود در شبکه را جست‌وجو کند، از کتابخانه Bleak در زبان پایتون استفاده شده است. این برنامه توسعه‌یافته می‌تواند به دستگاه‌های IoMT متصل شده و تمامی سرویس‌ها و مشخصات آن‌ها را دریافت کند. مشخصات دستگاه‌های IoMT این امکان را فراهم می‌کند که دستگاه را به شیوه‌ای خاص از عملکرد صحیح خود خارج کرد.

در ابتدا، برنامه بسته‌هایی با طول‌های مختلف ارسال می‌کند. طول هر بسته بین ۲۰ تا ۸۱۰ بایت، با افزایش ۱۰ بایتی تنظیم شده است. داده‌های ارسالی به شکل الگوی تکراری مانند ...0123456789012345 هستند که اعداد به صورت پشت سر هم تکرار می‌شوند. هر بار که

ارسال موفقیت‌آمیز انجام می‌شود، لاگ مربوطه ثبت می‌گردد. سپس برنامه مشخص می‌کند که هر بسته با موفقیت روی کدام UUID ارسال شده است.

آدرس UUID که مشابه SSID در شبکه‌های بی‌سیم Wi-Fi است، برای مشخص کردن آدرس شبکه بلوتوث استفاده می‌شود. پس از این مرحله، برنامه وارد حلقه‌ای می‌شود که داده‌ها روی UUIDهای شناسایی‌شده ارسال می‌شوند و در نهایت در یک متغیر ذخیره می‌گردند. این ارسال‌های پی‌درپی به منظور ایجاد بار اضافی (Overload) در دستگاه انجام می‌شوند که منجر به اختلال در عملکرد صحیح دستگاه شده و حمله‌ای شبیه به DoS را شبیه‌سازی می‌کند.

داده‌ها به دو روش جمع‌آوری می‌شوند. در روش اول، لاگ‌های سمت دستگاه اندروید که به دستگاه IoT متصل است، بررسی می‌شوند. روش دوم شامل استفاده از Sniff شبکه بلوتوث با استفاده از برد Ubertooth است که از طریق کامپیوتر متصل شده و قابلیت شناسایی تمامی بلوتوث‌های موجود در محدوده را فراهم می‌کند.

هدف اصلی از تحلیل این دو نوع لاگ، دستیابی به یک درک جامع و کلان از رفتار دستگاه‌ها در حالت عادی و هنگام وقوع حمله است. این حملات به شکل‌های مختلف آنقدر تکرار می‌شود تا نتیجه‌اش را به عنوان بررسی انعطاف‌پذیری و آسیب‌پذیری دستگاه در بستر حمله‌های مبتنی بر BLE ثبت کنند.

دستگاه‌های IoMT که در این حمله شرکت داشته‌اند به صورت زیر می‌باشد که هر کدام رفتار متفاوتی را نسبت به حملات داشته‌اند:

نام دستگاه	نتیجه بررسی
Lookee Sleep Ring	بدون وقفه عملکرد دستگاه در طی حملات مورد تأیید قرار گرفت.
Powerlabs HR Monitor Arm Band	دستگاه باید طبق استانداردها، حتی تحت حمله عملکرد صحیح خود را داشته باشد.
COOSPO HW807 Armband	حملات منجر به ایجاد اختلال و خاموش شدن دستگاه شدند که نشان‌دهنده شکست سخت‌افزار در برابر عوامل خارجی است.
Livlov Heart Rate sensor	سنسور بدون وقفه در برابر حملات مقاومت کرد.
Wellue O2 Ring	دستگاه بدون وقفه به عملیات خود ادامه داد.
Lookee O2 Ring	دستگاه تحت تأثیر حملات دچار اختلال شدید شد و خاموش شد.
Checkme BP2A	دستگاه داده‌ها را فقط در صورت اتصال پایدار بلوتوث ارسال می‌کرد، که نشان‌دهنده طراحی ایمن برای حفظ امنیت داده‌ها است.
SleepU Sleep Oxygen Monitor	در برابر حملات مقاومت کرده و بدون وقفه عمل کرده است.
Rhythm+2.0 (Scosche)	دستگاه به شدت تحت تأثیر حمله قرار گرفت و خاموش شد.
Wellue Pulsebit EX	دستگاه در برابر حملات مقاومت کرد و بدون قطعی به عملیات خود ادامه داد.
Checkme O2 Smart Pulse Oximeter	دستگاه مقاومت کرده و به عملیات استاندارد خود ادامه داده است.
Kinsa Thermometer	تحت تأثیر حملات قرار گرفت و تنها راه ریست کردن اتصال، تخلیه باتری بود.

جدول ۱: نتایج بررسی دستگاه‌ها تحت حملات مختلف

لازم به ذکر است که تمام حملات صورت گرفته کاملاً در محیطی ایزوله از هر گونه سیگنال خارجی انجام شده است تا با ترافیک شبکه جمع‌آوری شده هیچ سیگنال مزاحمی وجود نداشته باشد.

۶.۱ IoMT profiling

درک کامل جنبه‌های مختلف رفتار عملیاتی دستگاه‌های IoMT برای بهبود امنیت سیستم بسیار حیاتی است. ارزیابی پروفایل‌های IoMT قابلیت کلاسیفایرها را در تشخیص ناهنجاری‌های عملکردی دستگاه‌های مراقبت بهداشتی تقویت کرده و امکان تمایز بین رفتار عادی و

غیرعادی شبکه را فراهم می‌سازد. در تسک‌هایی که کلاسیفای نشده‌اند، الگوهای مشاهده‌نشده می‌توانند نشان‌دهنده فعالیت‌های مخرب و حملات zero-day باشند.

حملات zero-day به نوعی حمله سایبری اشاره دارند که از یک آسیب‌پذیری ناشناخته یا به‌تازگی کشف‌شده در نرم‌افزار، سخت‌افزار یا سیستم بهره می‌برند. نام این نوع حمله از آنجا گرفته شده است که توسعه‌دهنده یا مالک سیستم پیش از وقوع حمله برنامه‌ای برای شناسایی و رفع آسیب‌پذیری نداشته است. در این تحقیق، تمامی حملات موردنظر شبیه‌سازی و بررسی شده‌اند تا توسعه‌دهندگان و شرکت‌ها بتوانند از قرار گرفتن در معرض حملات zero-day جلوگیری کنند.

ویژگی‌های اصلی حملات zero-day [۸]:

- آسیب‌پذیری ناشناخته: آسیب‌پذیری که هنوز توسط تیم امنیتی یا تیم توسعه کشف یا رفع نشده است.
- غالفگیری: به دلیل آگاهی از آسیب‌پذیری، سیستم‌ها در برابر این نوع حملات کامل آسیب‌پذیر هستند.
- سوء استفاده یا Exploit: مهاجمان معمولاً از کد یا تکنیک‌هایی برای بهره‌برداری از آسیب‌پذیری استفاده می‌کنند.

۱.۶.۱ آزمایش‌های مبتنی بر انرژی و باتری

آزمایش‌های این قسمت مربوط به مصرف انرژی دستگاه‌های مجهز به Wi-Fi بوده است. این آزمایش‌ها تلاش می‌کنند تا رفتار دستگاه‌های Wi-Fi را از نظر مصرف انرژی و ارسال داده‌ها در هنگام روشن و یا خاموش بودن را به دقت تحلیل کنند. این آزمایش تنها روی هفت دستگاه Wi-Fi تمرکز دارند و روی دستگاه‌های MQTT شبیه‌سازی نشده‌اند. برای انجام این آزمایش:

تمام دستگاه‌ها از شبکه خارج شده‌اند و به جایی متصل نخواهند بود. تنها دستگاه متصل به شبکه یک دستگاه iPad خواهد بود که برای کنترل و نظارت بر سایر دستگاه استفاده می‌شود. حتی دستگاه‌های RPi نیز در این آزمایش متصل نبودند. مراحل زیر در ادامه انجام شده‌اند:

۱. در مرحله اول، دستگاه مورد نظر روشن شد و رفتار آن برای مدت ۲ دقیقه با استفاده از فیلتر آدرس MAC ثبت شد.

۲. در مرحله دوم، دستگاه خاموش می‌شود و فرایند ثبت داده‌ها برای ۳ دقیقه دیگر ادامه پیدا می‌کند تا هر بسته اطلاعاتی باقی‌مانده شناسایی شوند و اطمینان حاصل شود که دیگر هیچ بسته‌ای ارسال نمی‌شود.

فرایند انجام شده بالا دقیقاً همانند فرایندی است که در جمع‌آوری دیتاست CICIoT2021 انجام شده بود. نکته قابل توجه آن است که برخی از دستگاه‌ها کلید روشن و خاموش نداشتند و برای انجام آزمایش نیاز به انجام تنظیمات مخصوص داشتند:

- Singcall Sensor
- SOS Multifunctional Page
- Sense U Baby

۲.۶.۱ آزمایشگاه‌های مبتنی بر حالت Idle

آزمایشاتی است که تنها در زمان روشن بودن دستگاه ولی در حالی که هیچ داده‌ای منتقل نمی‌کردن، انجام می‌شود. انجام این آزمایش به محققان اجازه داد که رفتار حالت عادی و baseline شبکه رو بهتر متوجه بشن. در این آزمایش ۱۳ ساعت عملیات بررسی و آزمایش طول کشید. آزمایشات بین دو شب از ساعت ۶ عصر تا تا ۷ صبح برای اطمینان از اینکه هیچ تعاملی دستگاه‌ها ندارند انجام شد.

۳.۶.۱ آزمایش‌های فعال

شامل دستگاه‌هایی می‌شود که وظایف مورد نظر خود را انجام می‌دادند و ترافیک عادی را در شبکه بابت عملیات خود ایجاد می‌کردند.

۴.۶.۱ آزمایش‌های مبتنی بر تعامل

این قسمت از آزمایشات روی تعامل با دستگاه‌ها متمرکز است که به صورت زیر انجام شده‌اند:

- فیزیکی: کاربر مستقیماً با دستگاه ارتباط برقرار می‌کند.
- دیجیتالی:برنامه‌ها بایکدیگر ارتباط می‌گیرند و ارسال داده انجام می‌دهند.

سه نوع تعامل بررسی شده است:

- تعامل فیزیکی:
  - این آزمایشات زمانی انجام شده‌اند که دستگاه‌ها دارای دکمه فیزیکی بوده‌اند.
  - آزمایشات با ترکیب تعامل فیزیکی و شبکه‌های محلی یا شبکه‌های گسترده صورت گرفته است:
    - در شبکه محلی: اپلیکیشن و دستگاه در یک شبکه قرار دارند.
    - در شبکه WAN: اپلیکیشن از شبکه‌ای متفاوت نسبت به دستگاه متصل بوده است.
- تعامل در شبکه محلی:
  - در این آزمایشات از اپلیکیشن‌های همراه دستگاه استفاده شده است. اپلیکیشن و دستگاه هر دو در یک شبکه با استفاده از Wi-Fi بوده‌اند. برای مثال روشن و خاموش کردن دستگاه از طریق اپلیکیشن در خانه.
  - در تعامل با شبکه گسترده نیز همینطور بوده است، اپلیکیشن همراه دستگاه به شبکه‌ای متفاوت متصل بوده و کنترل دستگاه از راه دور مثلاً روشن کردن دستگاه خانه از دفتر کار صورت گرفته است.

۷.۱ ارزیابی‌های مبتنی بر یادگیری ماشین

آزمایشاتی که انجام شده آنقدر باعث تولید داده شد که می‌توان با استفاده از آن‌ها ارزیابی‌هایی را برای تشخیص و جلوگیری حمله در سیستم‌های IoMT انجام داد. از جمله این مدل‌های یادگیری ماشین می‌توان به موارد زیر اشاره کرد:

- Logistic Regression
- Random Forest
- Adaboost
- Deep Neural Networks (DNN)

دسته‌بندی و طبقه‌بندی داده‌ها نیز می‌تواند به صورت attack و Benign یا به صورت مشخص‌تر، Dos, MQTT, recon و یا DDoS انجام شود.

۲ رابطه مدل‌های ریاضی با شاخص‌های کلیدی کارایی

برای مدل‌سازی جهت ارزیابی عملکرد سیستم‌های مختلف از قبیل سیستم‌های IoT بایستی شاخص‌های کلیدی کارایی یا KPI [۱] را بشناسیم. این KPIها شامل مجموعه‌ای از فرمول‌های ریاضیاتی هستند که نسبت به نیازمندی‌های غیرعملیاتی یا Non-functional requirements متفاوت می‌باشند. نیازمندی‌های غیرعملیاتی مانند میزان مصرف انرژی، میزان مصرف حافظه، سرعت انتقال اطلاعات و گذردهی، میزان دقت، میزان دردسترس بودن، میزان قابلیت اطمینان و غیره می‌باشد. برای هر کدام از این نیازمندی‌های غیرعملیاتی فرمول‌هایی وجود

[۱] Key Performance Indicators

دارد که می‌تواند با فراگیری آن‌ها به مقادیری رسید که در مرحله بعد می‌توان آن‌ها را در مدل‌های ریاضیاتی استفاده کرد. مرحله دوم در حقیقت ارزیابی کارایی سیستم‌ها با استفاده از مدل‌های شناخته شده ریاضیاتی می‌باشد که رابطه مستقیمی با KPIها دارند. یعنی نمی‌توان مقادیر KPI را برای این مدل‌ها نادیده گرفت. در نهایت با مدل‌های مطرح شده می‌توان کارایی سیستم‌های IoT در دامنه‌های مختلف را ارزیابی کرد و بین آن‌ها مقایسه و Trade off نسبت به Benchmarkهای بدست آمده انجام داد.

KPIها و مدل‌هایی که در این گزارش مورد بررسی قرار گرفته‌اند:

۱.۲ KPIها

-

۲.۲ مدل‌ها

-

یکپارچه‌سازی سیستم‌های IoT با سیستم‌های ابری چالش‌های زیادی داشته مثل:

- تاخیرهای شبکه‌ای
- گذردهی
- مصرف انرژی
- قابلیت اطمینان

یه سری مفاهیم جدیدی در حوزه پردازش‌ها مطرح شده که حتی می‌تواند کاربردهای مختلفی در استفاده از اینترنت اشیا باشه. این مفاهیم جدید مثل Fog computing، edge computing، mobile edge computing، mobile cloud computing و Cloudlet ها هستش. در این مقاله یک مدل ریاضیاتی برای توصیف رسمی سیستم‌های IoT ارائه داده شده است. علاوه‌بر این یک ارزیابی آنالیز شده برای طراحی این سیستم‌ها با استفاده از مطابقت با معماری، تکنولوژی‌ها، پروتکل‌ها و مدل‌های یکپارچه‌سازی برای بهینه‌سازی عملکرد نیز ارائه می‌دهد.

Approach of this article:

بعد از خوندن این مقاله به یک روش بهینه برای بهینه‌سازی کارایی مبتنی بر فرایندهای offloading مانند load balancing آشنا می‌شیم. مدلینگ ریاضیاتی سیستم‌های IoT یک نمایی از سیستم ایجاد می‌کنند که به فهمیدن المان‌ها، تعاملاتشون، و رفتارهاشون کمک می‌کند.

۱. مدل مفهومی یا conceptual model یک ساختار سطح بالایی برای توصیف عملیاتی است که در سیستم‌های IoT انجام می‌شود.

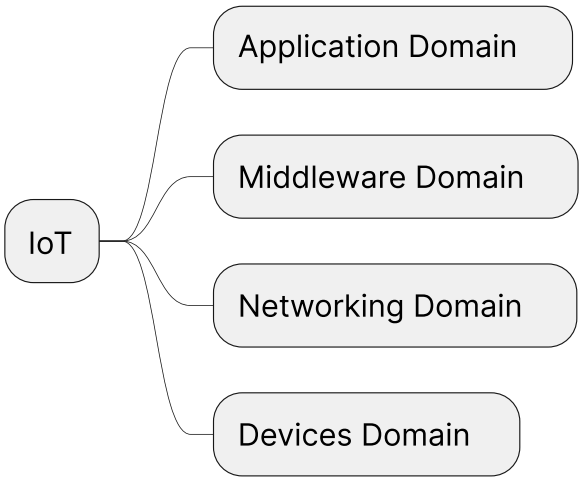
۲. مدل رفتاری یا behavioral model ممکنه شامل جزیئیات باشه. مثل جریان داده بین المان‌ها.

به طور کلی مودلینگ به مشخص شدن و پاسخ به مسائل مربوط به کارایی کمک بسزایی می‌کنه و اجازه میده که سیستم‌ها بهینه‌تر، کاراتر و مطمئن‌تر باشن.

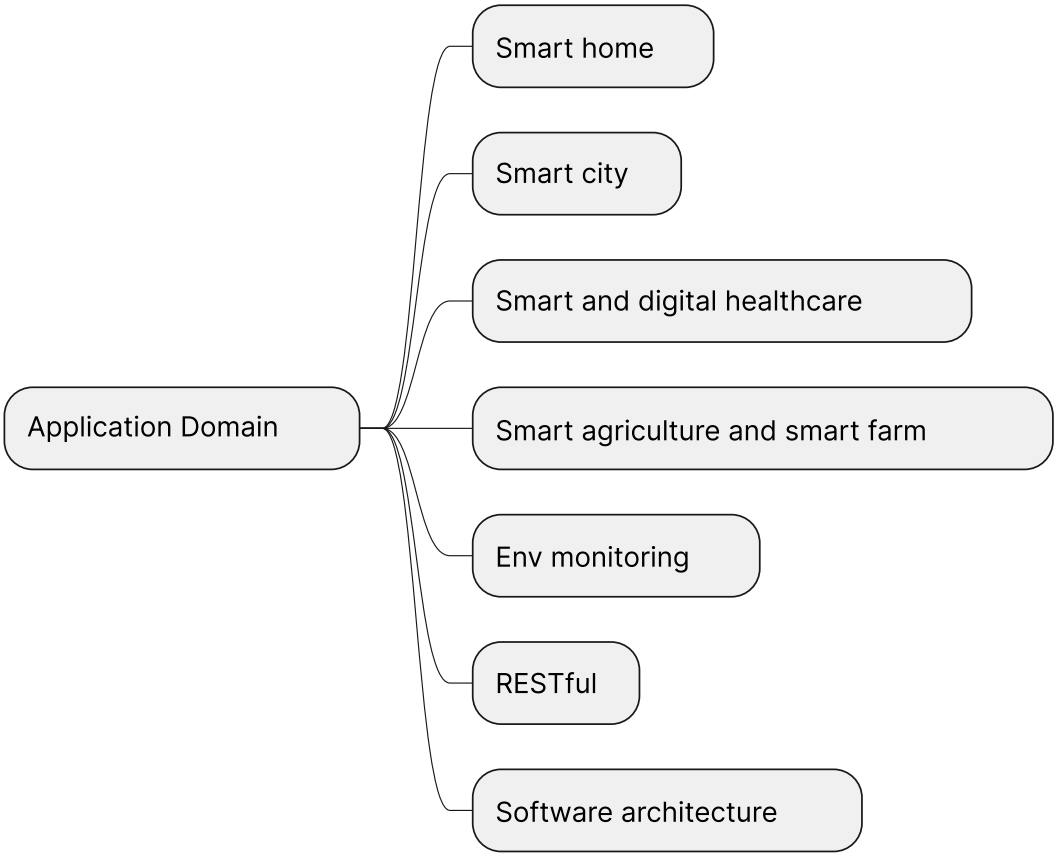
هر موقع در مورد مودلینگ یک سیستم IoT صحبت میشه در حقیقت قراره یه چهارچوبی درست بشه که بتونیم باهاش تست کنیم، تایید یا validation انجام بدیم و یا بتوانیم سیستم را به تقاضاهایی که داریم optimize کنیم.

فرایند سیستم‌های IoT معمولاً شامل شناسایی، دریافت اطلاعات ،(Sensing) فعالیت‌های تحت شبکه، و محاسبات کوچک هستش که باعث میشه با محیط فیزیکی و هر اشیایی ارتباط برقرار کند.

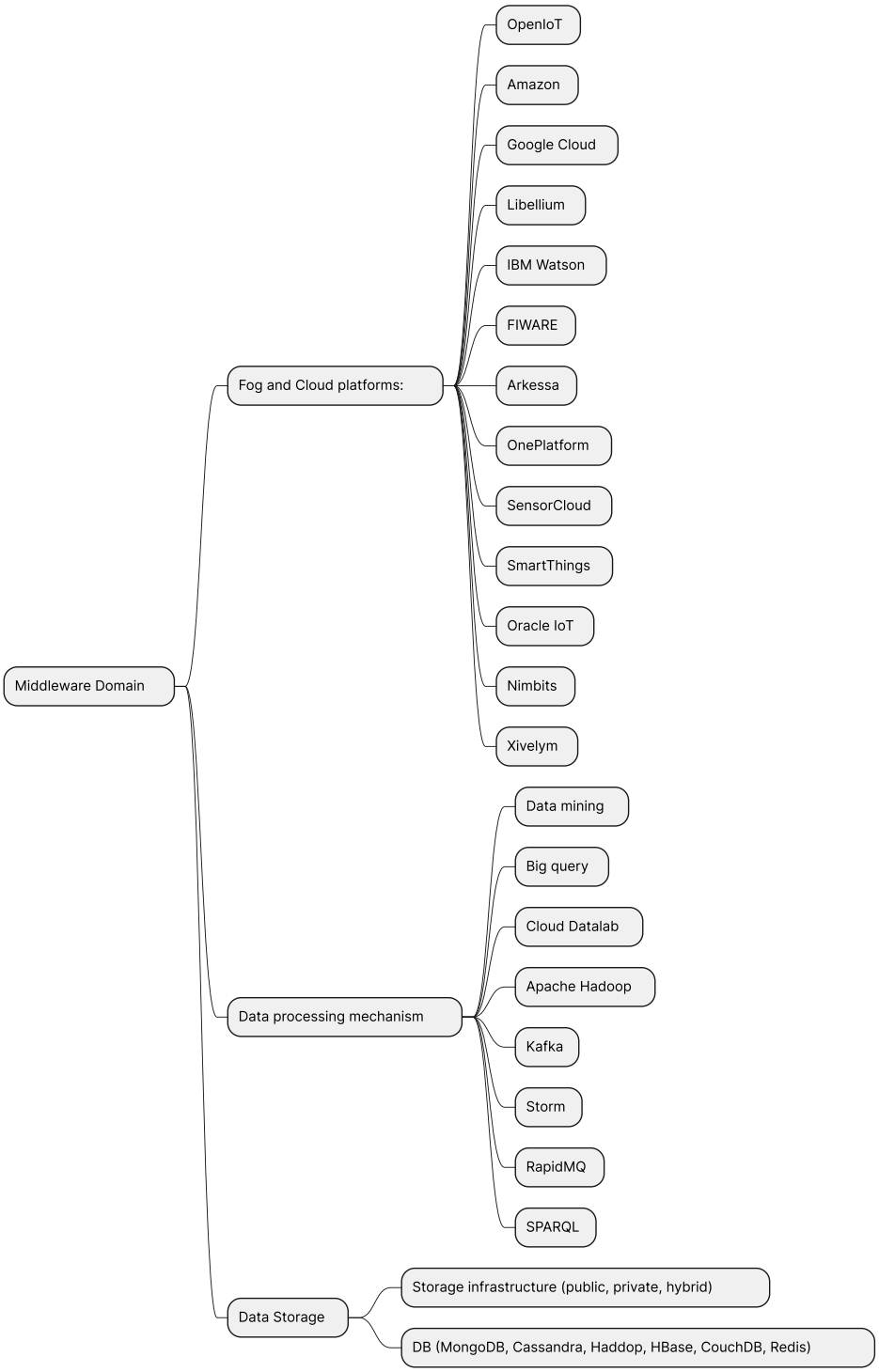
در این مقاله سیستم‌های IoT با کاربردی که دارند به ۴ دسته تقسیم می‌شوند:



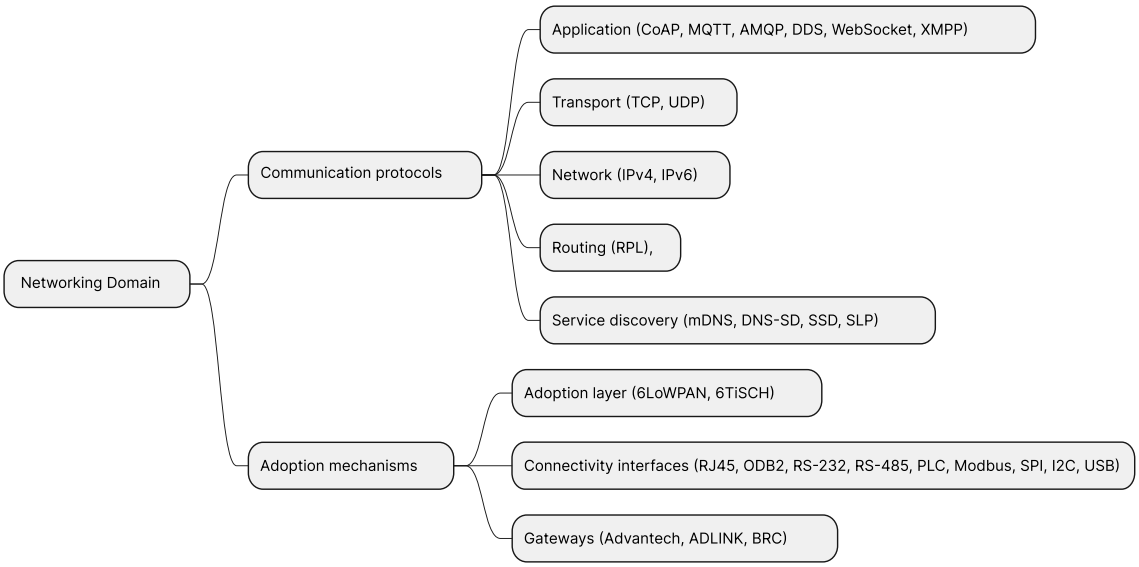
شکل ۲: ۴ دسته‌بندی دامنه استفاده از سیستم‌های IoT



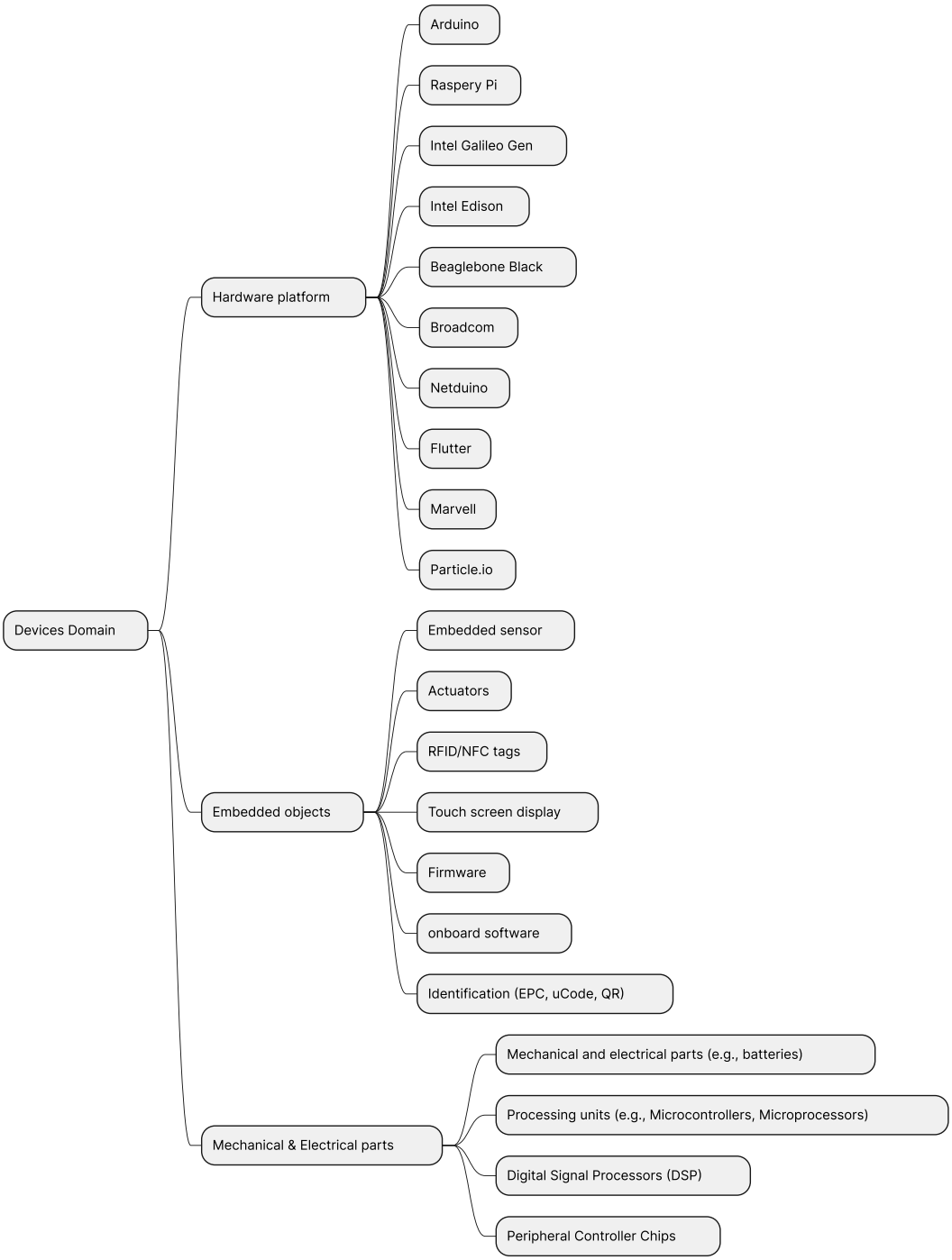
شکل ۳: حوزه‌های تخصصی بخش اپلیکیشن در IoT



شکل ۴: حوزه‌های بخش میان‌افزار در IoT



شکل ۵: حوزه‌های بخش شبکه در IoT



شکل ۶: حوزه‌های بخش Embedded در IoT

عمومیت جریان داده در سیستم‌های IoT:

- دریافت داده
- انتقال داده
- پردازش داده

- ذخیره‌سازی داده
- آنالیز و معنادار کردن داده

۳ شاخش‌های محاسبه و ارزیابی عملکرد

ثابت‌ها	تعاریف
D_{rate}	نرخ ورود اطلاعات
E_{dev}	مصرف انرژی
$T_{exe.}$	تاخیر سرویس‌دهی
$IoT_{sys_{sp}}$	مشخصات سیستم IoT
t_{ws}	زمان ورکلودها
k	تعداد دستگاه‌ها
β	درصد داده‌ای که در سیستم i پردازش می‌شود
τ	مقدار گذردهی
$IoT_{sys}(P) = \{P_{i_{req}}, P_i\}$	نیازمندی مشخص کارایی $P_{i_{req}}$ جایی که i^{th} P_i مقدار از کارایی است
$P_i = \langle T_{exe.}, E_{dev} \rangle$	مدت زمان سرویس‌دهی
E_{loc}	مصرف انرژی داخلی
E_{off}	انرزی مورد نیاز برای offloading
E_{idle}	انرژی مصرفی در زمان بیکاری
P_L	پردازش تسک‌های داخل دستگاه
t_L	زمان پردازش تسک‌های داخلی

جدول ۲: تعریف ثابت‌های مورد استفاده در فرمول‌ها

۱.۳ فرمول شانون

نرخ داده‌ها می‌تواند از طریق فرمول شانون محاسبه شود.

$$D_{rate} = B_{i,j} \log_2(1 + \frac{|h_{ij}|^2.P_{tx}}{P_{Nj}}) \quad (۱)$$

- B_{ij}: پهنای باند
- h_{ij}: بهره کانال بین دستگاه مبدا و مقصد که نشان می‌دهد سیگنال چگونه در مسیر بین فرستنده و گیرنده تقویت یا تضعیف می‌شود.
- P_{tx}: توان ارسال
- P_N: میزان نویز مقصد

کاربرد زیادی در سناریوهایی دارد که در آن یک ارسال کننده و یک دریافت کننده وجود دارد.

۲.۳ فرمول محاسبه بار سیستم یا System load

مجموع بار سیستم از طریق فرمول زیر بدست می‌آید:

$$D = \sum_{k=1}^{N} D_{rate,k} \times T_{w,k} \quad (۲)$$

- $D_{rate,k}$: نرخ تولید داده توسط دستگاه IoT
- $t_{w,k}$: مدت زمان ورودی در دستگاه IoT یا به عبارتی دیگر، مدت زمانی که طول می‌کشد یک دستگاه IoT ورودی را دریافت و سپس آن را پردازش و هندل کند.

۳.۳ تاخیر سرویس‌دهی یا Service latency

مسئله service latency یا service execution time ($T_{exe.}$) مدت زمانی است که طول می‌کشد سیستم IoT تمام درخواست‌های پردازشی و ارتباطی را اجرا کند (the total application exe time). مدت زمان کل مصرف شده از، مدت زمان سرویس یک ریکوئست به مدت زمان تمام تسک‌هایی که با موفقیت پردازش شده‌اند. بنابراین، این فاصله زمانی بین درخواست برنامه و بدست آوردن نتایج می‌باشد.

۱.۳.۳ بخش‌هایی که زمان سرویس‌دهی دارند

- مدت زمان انتقال داده از دستگاه IoT به زیر ساخت فاگ
- مدت زمان انتقال داده از دستگاه IoT به سرورهای ابری
- مدت زمان انتقال داده از فاگ به کلاد
- مدت زمان انتقال اعلانات از کلاد به فاگ
- مدت زمان انتقال اعلانات از کلاد به دستگاه‌های IoT
- مدت زمان انتقال اعلانات از فاگ به دستگاه IoT
- مدت زمان محاسبات در دستگاه IoT
- مدت زمان محاسبات در لایه فاگ
- مدت زمان محاسبات در سرورهای ابری

نکته: مدت زمانی که برای هر کاری در سیستم‌های IoT سپری می‌شود به نوع و شیوه پیاده‌سازی معماری دستگاه‌ها و نرم‌افزار بخش‌ها بستگی دارد و می‌تواند کاملاً متفاوت باشند. عموماً سرویس لیتنسی بین المان‌های سیستم IoT توزیع شده هستش و شامل دستگاه‌های اینترنت اشیا، شبکه‌ها و سیستم‌های پردازشی می‌شود.
تاخیر سرویس‌دهی:

$$T_{exe.} = T_{cm} + T_{cp} \quad (۳)$$

- T_{cm}: مدت زمان تاخیر در ارتباطات
- T_{cp}:مدت زمان تخیر در پردازش

مدت زمان اجرا بایستی کمتر از زمان‌بندی تسک‌ها در فاگ یا کلاد باشد. یعنی سرویس تایم باید کمتر از نیازمندی‌های IoT application (T_req) باشد.
برای کاهش service latency از فرمول زیر بایستی پیروی کند:

$$Objective : \min(T_{exe.}) = T_{cm} + T_{cp} \leq T_{req} \quad (۴)$$

موقعی که داری در مورد Execution time می‌نویسی فرمول communication latency رو باید داشته باشی که بگی از کجا بدست میاد. و همچنین فرمول computation latency رو هم بعدش. یعنی ثابت‌ها خودشون از زیر ثابت‌های اصلی بدست میان که جمع میشن و میشه T_{exe}.

۲.۳.۳ زمان ارتباطی

$$T_{cm} = \sum_{i=1}^{N}(d_{proc} + d_{queue} + d_{trans} + d_{prop}) \quad (۵)$$

- d_{prop}: تاخیر پردازشی
- d_{queue}: تاخیر در صف
- d_{trans}: تاخیر انتقال
- d_{prop}: تاخیر توزیع

تاخیر مربوط به Propagation مجموع زمان مورد نیاز برای داده جهت ارسال از منبع به مقصد که مبتنی بر طول لینک فیزیکی و سرعت رسانا می‌باشد.

$$d_{trans} = \frac{P_s}{R_L} \quad (۶)$$

که در آن:

- P_s: اندازه بسته در واحد bits
- R_L: سرعت لینک ارتباطی bps

$$d_{prop} = \frac{l_{ij}}{c} \quad (۷)$$

- l_ij: لینک فیزیکی
- c: سرعت توزیع media

۳.۳.۳ زمان پردازشی

$$T_{cp} = T_L + \sum_{i=1}^{k} t_{offi} \quad (۸)$$

که در آن:

- t_L: اجرا و پردازش‌های داخلی
- t_offi: اجرا و پردازش‌های خارج از دستگاه IoT مانند برنامه‌هایی که در سیستم‌های ابری یا Fog مستقر شده‌اند که وظیفه پردازش تسک‌های Offloading را دارند.

به بیان دیگر می‌توان آن را به صورت مدل زیر محاسبه کرد:

$$T_{cp} = t_L + t_F + t_C \quad (۹)$$

$$T_{cp} = t_L + \max_{i=1,\dots,k} t_{F_i} + \max_{j=1,\dots,n} t_{C_i} \quad (۱۰)$$

که در آن:

- t_L: مدت زمان پردازش‌های داخلی
- t_{F_i}: مدت زمان پردازش در نود i^{th} در Fog
- t_{C_i}: مدت زمان پردازش در سرور i^{th} ابری

عموماً مصرف پردازشی بستگی به سرعت و معماری پردازنده مرکزی (CPU)، حافظه رم (RAM)، سرعت حافظه ذخیره‌ساز (HDD) یا (SSD)، سرعت پردازنده گرافیکی یا (GPU) و غیره. دارد.

۴.۳.۳ زمان پردازش محلی t_L یا زمان پردازش در هر زیر سیستم t_{pi}

برای بدست آوردن زمان پردازش در هر زیر سیستم از فرمول زیر استفاده می‌شود:

$$t_{pi} = \frac{I_{CC_i}}{f_{cpu,i}} \tag{۱۱}$$

- t_{pi}: زمان پردازشی در زیر سیستم i
- I_{CC_i}: تعداد سایکل‌های CPU که برای اجرای یک برنامه نیاز است.
- $f_{cpu,i}$: نرخ کلاک (فرکانس کاری CPU) زیر سیستم i

$$t_{Pi} = t_{CPU_i} + t_{I/O_i} \tag{۱۲}$$

۵.۳.۳ تابع محاسبه CPU time

مدت زمانی که در CPU برای اجرا برنامه در نظر گرفته می‌شود به دو دسته تقسیم می‌شود:

۱. User CPU time

۲. System CPU time: t_{OS}

محاسباتی که در CPU time انجام می‌شود خالصانه در قسمت پردازشگر مرکزی صورت می‌گیرد و هیچ محاسبه جانبی مانند مدت زمان I/O و مدت زمان اجرای دیگر برنامه‌ها در نظر گرفته نمی‌شود.

$$t_{cpu_i} = \frac{I_{CC_i}}{f_{cpu_i}} + t_{OS} = I_{CC_i} \times t_{cc_i} + t_{OS} \tag{۱۳}$$

حاصل این تابع معمولاً بسیار کوچک است و می‌تواند نادیده گرفته شود زیرا به سمت صفر میل می‌کند ($t_{OS} \to 0$). به همین خاطر بیشتر روی User CPU time تمرکز می‌کند که توسعه‌دهنده بر روی آن کدهای خود را اجرا می‌کند و سیستم IoT را راه‌اندازی می‌کند.

تابع مطرح شده بر اساس قدرت محاسباتی دستگاه $i(f_{cpu_i})$ و تعداد کلاک CPU برای اجرای یک برنامه (I_{CC_i}) می‌باشد. مقدار f_{cpu_i} بر واحد (Hz) می‌باشد و t_{cc_i} زمان چرخه کلاک است. لازم به ذکر است که I_{CC_i} به نوع دستورالعمل که شامل اندازه داده ورودی، زبان برنامه نویسی، میزان پیچیدگی الگوریتم نرم‌افزاری مورد استفاده، و دیگر موارد می‌باشد.

بخش CPI (Clock cycles per instruction) به عنوان میانگین تعداد چرخه کلاک است که هر دستورالعمل به آن نیاز دارد. اگر I_{app_j} تعداد دستورالعمل‌ها برای یک برنامه باشد آن وقت I_{CC_i} از طریق معادله زیر بدست می‌آید.

$$I_{CC_i} = \sum_{j=1}^{k} I_{app_j} \times CPI_j \tag{۱۴}$$

۶.۳.۳ زمان پردازش محلی با توجه به اندازه داده (D)

$$I_{CC_i} = X \times D \tag{۱۵}$$

- D: اندازه ورودی داده بر حسب بیت
- X: شدت پردازش (تعداد چرخه‌های مورد نیاز برای هر بیت داده)

بنابراین خواهیم داشت:

$$t_{pi} = \frac{\beta_i \times D \times X}{f_{cpu,i}} \tag{۱۶}$$

که در آن β_i درصد داده‌ای است که در سیستم i پردازش می‌شود.

۴.۳ مصرف انرژی

۱.۴.۳ مجموع مصرف انرژی E_{dev}

برای محاسبه مصرف کل انرژی در سیستم‌های IoT می‌توان از فرمول زیر استفاده کرد:

$$E_{dev} = F(E_{cp}, E_{cm}, E_{idle}, E_{other}) \tag{۱۷}$$

- E_{cp}: انرژی مصرف شده طی محاسبات
- E_{cm}: انرژی مصرف شده در طی ارتباطات
- E_{idle}: انرژی مصرف شده در حالت نرمال و بیکار سیستم
- E_{other}: انرژی مصرف شده توسط بقیه فرایندها مانند سنسورها، صفحه نمایش، کارت گرافیک و غیره.

یک مدل برای بررسی مصرف انرژی توسط دستگاه‌های IoT که شامل فرایندهای پردازشی و ارتباطی می‌شود عبارت است از:

$$E_{dev} = E_{loc.} + E_{off.} \tag{۱۸}$$

میزان انرژی مورد نیاز برای پردازش داخلی به تسک‌های داخلی وابسته می‌باشد:

$$E_{loc.} = P_L \times t_L \tag{۱۹}$$

محاسبه میزان انرژی پردازشی از حاصل ضرب I_{CC_i} و انرژی مصرفی CPU به ازای هر چرخه CPU بدست می‌آید:

$$E_{loc.} = k \times I_{CC_i} \times f_{cpu_i}^2 = k \times D \times X \times f_{cpu_i}^2 \tag{۲۰}$$

- k: ثابتی است که به مشخصات سخت‌افزار مربوط است.
- D: اندازه داده ورودی بر مبنای bits

- X: شدت یا داده ورودی محاسباتی

محاسبه انرژی برای انجام تسک‌های offloading نیازمند انرژی برای ارسال داده‌ها و دریافت نتایج آن می‌باشد که با E_{comm} نمایش می‌دهند. لازم به ذکر است مدت زمانی که طول می‌کشد سیستم نتایج را دریافت کند سیستم در وضعیت idle باقی مانده است. به همین صورت برای بدست آوردن مصرف انرژی برای تسک‌های offloading از فرمول زیر استفاده می‌شود:

$$E_{off.} = E_{cm.} + E_{idle} \quad (۲۱)$$

در واقع E_{id} انرژی مورد استفاده دستگاه IoT در زمانی که دستگاه در حالت بیکار می‌باشد. این بیکاری به منظور آن است که دستگاه IoT در حال انتظار برای دریافت نتیجه از سرورها می‌باشد.

$$E_{idle} = P_{idle} \times t_{off} \quad (۲۲)$$

$$t_{off} = T_{exe.} - (t_L + T_{cm}) \quad (۲۳)$$

مجموع انرژی مصرفی جهت انتقال داده‌ها با انرژی مصرفی در هنگام دریافت داده‌ها ما را به انرژی مصرفی ارتباطی می‌رساند:

$$E_{cm} = E_{tx} + E_{rx} \quad (۲۴)$$

در یکی از کارها [۹] مدلی برای مصرف انرژی نسبت به انتقال و جا به جایی داده‌ها مطرح شده که سطوح مختلف مصرف باتری را برای uplink و downlink شامل می‌شود:

$$P_{tx} = p_u \tau_u + \beta \quad (۲۵)$$

$$P_{rx} = p_d \tau_u + \beta \quad (۲۶)$$

گذردهی uplink، τ_u می‌باشد و گذردهی downlink، τ_d است. در حالی که p_u و p_d میزان انرژی مورد نیاز برای انتقال داده‌ها در uplink و downlink می‌باشد. ثابت β میزان مصرف انرژی در حالت idle می‌باشد. این مقادیر کاملاً به تکنولوژی ارتباطی، پروتکل‌ها و دستگاه‌هایی که روی آن برنامه مستقر شده است وابسته می‌باشد. برای انتقال همزمان uplink و downlink سطح انرژی می‌تواند با فرمول زیر محاسبه شود:

$$P_{trx} = p_u \tau_u + p_d \tau_d + \beta \quad (۲۷)$$

نسبت معادله uplink بر روی downlink به همراه پهنای باند، می‌تواند فرمول را برای محاسبه انرژی بهینه شبکه برای انتقال یک مقدار داده معین (energy per bit). مقدار انرژی مورد نیاز برای ارسال از طریق فرمول $E(D)_{rx}$ میزان انرژی مورد نیاز برای دریافت داده‌ها از طریق فرمول $E(D)_{rx}$ حاصل می‌شود.

$$E(D)_{tx} = p_u + \beta \tau_u^{-1} \quad (۲۸)$$

$$E(D)_{rx} = p_d + \beta \tau_d^{-1} \quad (۲۹)$$

مقدار داده‌هایی که بر واحد بیت توسط دستگاه‌های IoT ارسال و دریافت می‌شوند به ترتیب D_{tx} و D_{rx} می‌باشند. با در نظر گرفتن محاسبات پیشین، می‌توان در نهایت میزان مصرف انرژی توسط دستگاه‌های IoT را به شکل زیر بدست آورد:

$$E_{dev} = (P_L \times t_L) + (P_{tx} \times t_{tx}) + (P_{rx} \times t_{rx}) + (P_{id} \times t_{off}) \quad (۳۰)$$

$$E_{dev} = (P_L \times t_L) + ((p_u \tau_u + \beta) \times t_{tx}) + ((p_d \tau_d + \beta) \times t_{rx}) + (P_{id} \times t_{off}) \quad (۳۱)$$

$$E_{dev} = (P_L \times t_L) + ((p_u + \beta \tau_u^{-1}) \times D_{tx}) + ((p_d + \beta \tau_d^{-1}) \times D_{rx}) + (P_{id} \times t_{off}) \quad (۳۲)$$

یکی از مهم‌ترین چالش‌های دستگاه‌های IoT مربوط به مصرف باتری آن‌ها می‌باشد. در بعضی مواقع دستگاه‌های IoT از باتری‌هایی استفاده می‌کنند که شرایط جایگزین کردن آن‌ها وجود ندارد. هر دستگاه IoT حتی در حالت بیکار انرژی بابت، پردازش داده‌ها، ارسال و دریافت داده‌ها مصرف می‌کنند. انرژی موجود $E_{dev}(t)$ در طی زمان کاهش پیدا می‌کند. به همین ترتیب انرژی باقی‌مانده $E_{dev}(r)$ یا مدت زمانی که سیستم می‌تواند روشن بماند از طریق فرمول زیر بدست می‌آید.

$$E_{dev}(r) = E_{dev}(i) - E_{dev}(t) \quad (۳۳)$$

- $E_{dev}(i)$: مقدار اولیه انرژی دستگاه

مقدار باتری باقی‌مانده $T(sys)$ به میزان ظرفیت باطری یا انرژی باقی‌مانده و انرژی مورد نیاز دستگاه برای انجام تمام سرویس‌های دستگاه، بستگی دارد. انرژی مصرفی وابسته به قدرت مورد نیاز برای پردازش‌های داخلی P_{cp} انتقال داده‌ها P_{cm} و بقیه فرایندها P_{other} می‌باشد.

$$T(sys) = \frac{E_{dev}(r)}{P_{cp} + P_{cm} + P_{other}} \quad (۳۴)$$

یکی دیگر از چالش‌های دستگاه‌های IoT مربوط به منبع‌تغذیه آن‌ها می‌باشد. در مواقعی که دستگاه‌های IoT از باتری استفاده می‌کنند و به دلیل محیط‌های مختلف شرایط به گونه‌ای است که امکان تعویض باتری وجود ندارد، محدودیت‌های باتری اغلب به عنوان شاخص طول عمر دستگاه‌های IoT استفاده می‌شود و می‌تواند به عنوان یکی از مهم‌ترین معیارهای QoS مورد استفاده قرار گیرد. هدف اصلی در این دستگاه‌ها این است که مصرف انرژی به حداقل برسد و طول عمر کلی سیستم به بیشترین حد ممکن.

۴ مدل‌های ارزیابی کارایی

در این بخش بررسی می‌شود که چگونه می‌توان عملکرد سیستم‌های IoT را ارزیابی کرد وقتی چندین شاخص کلیدی عملکرد KPIs با واحدها و اهمیت‌های مختلف وجود دارد.

۱.۴ ارتباط بین عملکرد و ویژگی‌های زیرساخت IoT

عموماً عملکرد یک برنامه به ویژگی‌های زیرساخت IoT مانند تواند محاسباتی، تاخیر شبکه، مصرف انرژی، و غیره وابسته است. هر تغییری در زیرساخت IoT می‌تواند تاثیر مستقیمی بر عملکرد برنامه داشته باشد.

۲.۴ مشکل مقایسه KPIs مختلف

هر KPI مانند تاخیر شبکه، گذردهی، پهنای باند، مصرف انرژی واحد و مقیاس خاص خود را دارد. برای مثال تاخیر بر حسب میلی‌ثانیه ms اندازه‌گیری می‌شود و گذردهی بر حسب mips و مصرف انرژی بر حسب وات‌ساعت. این تفاوت در واحدها باعث می‌شود مقایسه مستقیم آن‌ها دشوار شود. علاوه‌بر مختلف بودن واحدها و معیارها، برخی از KPIها ممکن است از اهمیت بیشتری برخوردار باشند که معمولاً به هر شاخص وزنی یا weight اختصاص می‌یابد.

۳.۴ استفاده از تابع سودمندی Utility function

در اینجا یک تابع کابردی به نام $IoT_{sys}(p)$ وجود دارد که عملکرد سیستم IoT را براساس KPIهای نرمال‌شده و وزن‌دهی شده ارزیابی می‌کند. این تابع می‌تواند برای مقایسه مدل‌های مختلف سیستم IoT استفاده شود و مشخص کند که کدام مدل بهتر است.

$$IoT_{sys}(p) = \sum_{i=1}^{n} w_i \times f(p_i) \tag{۳۵}$$

- n: تعداد شاخص‌های کلیدی کارایی
- (p_i): ارزیابی کارایی سیستم‌های IoT
- $f(p_i)$: مطابقت با تابع utility مشخص برای ارزیابی هر KPI که بین دو عدد ۰ و ۱ تبدیل (نرمال‌سازی) شده است.
- w_i: وزن ضرایب برای تعیین آن که کدام KPI در سیستم IoT مورد نظر مهم‌تر می‌باشد.

به عنوان مثال، یک سیستم IoT داریم که برای پایش سلامتی بیماران طراحی شده است. می‌خواهیم عملکرد این سیستم را ارزیابی کنیم. در اینجا چند KPI مهم وجود دارد:

۱. زمان تاخیر یا Latency: مدت زمانی که طول می‌کشد داده‌ها از سنسور به سیستم مرکزی برسند. (بر حسب میلی‌ثانیه)

۲. مصرف انرژی یا Energy consumption: انرژی مصرف شده توسط دستگاه‌ها (بر حسب وات‌ساعت).

۳. درصد دسترسی یا Availability: درصد زمانی که سیستم آنلاین و قابل استفاده می‌باشد. (بر حسب درصد)

۴. دقت یا Accuracy: درصد صحت داده‌های جمع‌آوری شده از سنسورها

گام‌های استفاده از مدل

نرمال‌سازی KPIها

چون شاخص‌ها واحدهای متفاوتی دارند، ابتدا باید همه مقادیر را به یک بازه یکسان به عنوان مثال ۰ و ۱ تبدیل کنیم. این کار باعث می‌شود تا همه KPIها بتوانند بدون واحد باشند. به عنوان مثال اگر زمان تاخیر بین ۰ تا ۱۰۰ میلی‌ثانیه است، مقدار ۵۰ میلی‌ثانیه به ۰/۵ نرمال‌سازی شود. یا اگر میزان دسترسی بین ۰ تا ۱۰۰ درصد است، مقدار ۸۰ درصد به ۰/۸ نرمال‌سازی شود.

وزن‌دهی

براساس اهمیت هر KPI یک وزن w_i به آن اختصاص می‌دهیم:

- $w_1 accuracy$ = 0.4
- $w_2 latency$ = 0.3
- $w_3 energy$ = 0.2
- $w_4 availability$ = 0.1

این وزن‌ها نشان‌دهنده اولویت و اهمیت KPI مورد نظر ما برای سیستم IoT می‌باشد.

ارزیابی عمکرد سیستم مذکور

با استفاده از این مدل همانطور که قبلاً گفته شد می‌توان سیستم‌های IoT را با وجود KPIهای مختلف با یکدیگر مقایسه کرد:
در فرمول ۳.۴ پارامتر $f(p_i)$ مقدار نرمال‌سازی‌شده هر KPI می‌باشد.
مقادیر KPIهای مورد نظر برای این سیستم پایش سلامت به صورت زیر می‌باشد:

- $f(p_1)$: 0.9 دقت
- $f(p_2)$: 0.6 زمان تاخیر
- $f(p_3)$: 0.7 مصرف انرژی
- $f(p_4)$: 0.8 میزان دسترسی

لازم به ذکر است ه مقادیر $f(p_i)$ توسط مدل‌هایی که تاکنون توضیح داده شد بدست آمده است و سپس مقادیر آن‌ها بین ۰ و ۱ نرمال‌سازی شده است.
با توجه به مدل ۳.۴ خواهیم داشت:

$$(0.8 \times 0.1) + (0.7 \times 0.2) + (0.6 \times 0.3) + (0.9 \times 0.4) = 0.78 \quad (۳۶)$$

برای سیستم مورد نظر با توجه به مقادیری که برای KPIهای مورد نظر بدست آمده بود مقدار 0.78 در حقیقت مقدار مدل سیستم پایش سلامت شد. با استفاده از این مقدار می‌توان عملکرد سیستم‌های پایش سلامت دیگر (یا دیگر سیستم‌های IoT مربوط به آن دامنه) را نیز به همین ترتیب ارزیابی کرد و با مقادیری که در نهایت با استفاده از مدل ۳.۴ بدست می‌آید، هر کدام از آن‌ها را با یکدگیر مورد مقایسه و Trade off قرار داد.
دو KPI در سیستم‌های IoT وجود دارد که می‌تواند سطح کارایی آن‌ها را مشخص کند. اول تاخیر در سرویس‌دهی، دوم مصرف انرژی. به همین ترتیب برای محاسبه سطح کلی کارایی سیستم‌های IoT از مجموع مقادیر تابع سودمندی برای هر KPI استفاده می‌شود:

$$IoT_{sys}(p) = w_{T_{exe}} \times f(T_{exe}) + w_{e_{dev}} \times f(E_{dev}) \quad (۳۷)$$

توابع $f(T_{exe.})$ و $E_{con.}$ توابع سودمندی هستند برای مدت زمان اجرا و مصرف انرژی و در ادامه آن $w_{T_{exe.}}$ و $w_{E_{conn.}}$ وزن‌های ضریب توابع هستند.

۴.۴ مدل‌سازی توابع سودمندی برای هر KPI

در این بخش بررسی می‌کنیم که چگونه می‌توان توابع سودمندی یا Utility functions را برای هر KPI مدل‌سازی کنیم. برای KPIهایی مانند تاخیر شبکه، مصرف انرژی، سرعت انتقال داده از توابع سیگموئیدی استفاده می‌شود که محدوده خروجی آن‌ها بین ۰ و ۱ می‌باشد و معیار عملکرد KPI را به صورت بی‌واحد نمایش می‌دهد.
نکات مهم:

- $f(p_t)$ یا $f(p_i \uparrow)$: برای KPIهایی که مقدارهای بیشتر بهتر است مانند طول عمر باتری یا گذردهی.
- $f(p_u)$ یا $f(p_i \downarrow)$: برای KPIهایی که مقدار کمتر بهتر است مانند تاخیر یا مصرف انرژی.

۵.۴ فرمول تابع سیگموئید

$$f(p_i \uparrow) = L(\frac{A}{1 + e^{-\alpha_k \frac{(p_i - r_i)}{z_i}}} - U) \quad (۳۸)$$

$$f(p_i \downarrow) = 1 - L(\frac{A}{1 + e^{-\alpha_k \frac{(p_i - r_i)}{z_i}}} - U) \quad (۳۹)$$

$$L = \frac{1 + e^{\frac{\alpha_k \times r_i}{z_i}}}{e^{\frac{\alpha_k \times r_i}{z_i}}}.U = \frac{1}{1 + e^{\frac{\alpha_k \times r_i}{z_i}}} \quad (۴۰)$$

- α_k: شیب منحنی است که حساسیت را به تغییرات KPI نشان می‌دهد (شب بیشتر برابر است با تغییر سریع‌تر عملکرد).
- r_i: نقطه مرکز یا مرجع KPI است (مثلاً یک مقدار استاندارد بهینه برای KPI که IoT باید به آن نزدیک باشد).
- L: بیشترین مقدار سودمندی
- U: کمترین مقدار سودمندی

۶.۴ اهداف فرمول‌های سیگموئید

- برای KPIهایی که افزایششان مطلوب است مانند گذردهی بیشتر بایستی $f(p \uparrow)$ را ماکزیمم کنیم.
- برای KPIهایی که کاهششان مطلوب است باید $f(p \downarrow)$ را مینیمم کنیم.

این مدل به ما اجازه می‌دهد که عملکرد KPIهای مختلف را بدون واحد قابل مقایسه کنیم و KPIها را براساس وزنی که به آن‌ها داده‌ایم مرتب کنیم و تصمیم‌گیری‌های بهینه‌تری برای بهبود عملکرد IoT انجام دهیم (برای مثال کدام KPIها بیشتر نیاز به بهینه‌سازی دارند).
روشی که در بالا ذکر شد اندازه‌گیری انعطاف‌پذیری را برای پارامترها ارائه می‌دهد حتی زمانی که هیچ مقدار مرجعی برای دستگاه IoT نداشته باشیم. این مدل به دلیل بی‌واحد بودن مقادیر به KPI خاصی وابسته نمی‌باشد. با این روش امکان تنظیم تابع سودمندی براساس حساسیت پارامترهای فردی فراهم می‌شود.

مراجع

[1] Dadkhah, Sajjad, Neto, Euclides Carlos Pinto, Ferreira, Raphael, Molokwu, Reginald Chukwuka, Sadeghi, Somayeh, and Ghorbani, Ali A. Ciciomt2024: A benchmark dataset for multi-protocol security assessment in iomt. *Internet of Things*, 28:101351, 2024.

[2] Dadkhah, Sajjad, Neto, Euclides Carlos Pinto, Ferreira, Raphael, Molokwu, Reginald Chukwuka, Sadeghi, Somayeh, and Ghorbani, Ali A. Cic iot and iomt dataset, 2024. `http://205.174.165.80/IOTDataset/CIC_IOT_Dataset2023/Dataset`.

[3] Ahmed, Mohiuddin, Byreddy, Surender, Nutakki, Anush, Sikos, Leslie F, and Haskell-Dowland, Paul. Ecu-ioht: A dataset for analyzing cyberattacks in internet of health things. *Ad Hoc Networks*, 122:102621, 2021.

[4] Zubair, Mohammed, Ghubaish, Ali, Unal, Devrim, Al-Ali, Abdulla, Reimann, Thomas, Alinier, Guillaume, Hammoudeh, Mohammad, and Qadir, Junaid. Secure bluetooth communication in smart healthcare systems: a novel community dataset and intrusion detection system. *Sensors*, 22(21):8280, 2022.

[5] Unal, Devrim. Bluetack, 2021. `https://raw.githubusercontent.com/MohammedZubair-lab/Bluetooth-BR-EDR-BlueTack/refs/heads/main/Bluetooth_dataset.csv`.

[6] Hussain, Faisal, Abbas, Syed Ghazanfar, Shah, Ghalib A, Pires, Ivan Miguel, Fayyaz, Ubaid U, Shahzad, Farrukh, Garcia, Nuno M, and Zdravevski, Eftim. A framework for malicious traffic detection in iot healthcare environment. *Sensors*, 21(9):3025, 2021.

[7] Brown, Matt. Bluetooth low energy hacking. `https://www.youtube.com/watch?v=dsZN0dqh81k`, 2023.

[8] Wikipedia contributors. Zero-day vulnerability — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Zero-day_vulnerability&oldid=1268124149`, 2025. [Online; accessed 15-January-2025].

[9] Huang, Junxian, Qian, Feng, Gerber, Alexandre, Mao, Z Morley, Sen, Subhabrata, and Spatscheck, Oliver. A close examination of performance and power characteristics of 4g lte networks. in *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 225–238, 2012.