SunBlock: Cloudless Protection for IoT Systems

Vadim Safronov[1(✉)], Anna Maria Mandalari[2], Daniel J. Dubois[3], David Choffnes[3], and Hamed Haddadi[1]

[1] Imperial College London, London, UK
{v.safronov,h.haddadi}@imperial.ac.uk
[2] University College London, London, UK
a.mandalari@ucl.ac.uk
[3] Northeastern University, Boston, USA
d.dubois@northeastern.edu, choffnes@ccs.neu.edu

Abstract. With an increasing number of Internet of Things (IoT) devices present in homes, there is a rise in the number of potential information leakage channels and their associated security threats and privacy risks. Despite a long history of attacks on IoT devices in unprotected home networks, the problem of accurate, rapid detection and prevention of such attacks remains open. Many existing IoT protection solutions are cloud-based, sometimes ineffective, and might share consumer data with unknown third parties. This paper investigates the potential for effective IoT threat detection locally, on a home router, using AI tools combined with classic rule-based traffic-filtering algorithms. Our results show that with a slight rise of router hardware resources caused by machine learning and traffic filtering logic, a typical home router instrumented with our solution is able to effectively detect risks and protect a typical home IoT network, equaling or outperforming existing popular solutions, without any effects on benign IoT functionality, and without relying on cloud services and third parties.

1 Introduction

Internet of Things (IoT) devices are increasingly popular, pervasive, and promise a wide range of functionality including remote health monitoring, adaptive climate and lighting, automated control of various appliances and assets within smart spaces. The heterogeneity of IoT device types, functionality, and their applications lead to major security and privacy threats for smart home users. IoT devices are manufactured all over the world, send data globally, may be updated over-the-air, and use a myriad of third-party software libraries [47]. A single vulnerability in one part of the IoT supply chain can have broad security and privacy impacts for IoT users. For example, compromised IoT devices can compromise user privacy, *e.g.,* exposing sensitive user activities on personal computers by reading power consumption from a compromised smart plug [27], among many other serious threats described in recent literature [32,48,49,51]. Also, the combination of IoT device sensors with Internet connectivity poses substantial risks of personally identifiable information (PII) exposure [47].

P. Richter et al. (Eds.): PAM 2024, LNCS 14538, pp. 322–338, 2024.
https://doi.org/10.1007/978-3-031-56252-5_15

To address such risks, several IoT protection solutions have been proposed so far, but they pose several issues, including their lack of completeness. Evidence from prior research [39] suggests that popular IoT protection solutions fail to accurately detect and mitigate a significant proportion of threats, particularly those available with basic subscriptions. Moreover, most IoT protection solutions largely rely on cloud-based ML models for detecting attacks. Such cloud-based models necessarily require vendors to collect personal and sensitive data from the networks they protect, and this itself poses the risk of such data being shared with third-parties, used for purposes beyond threat detection, and even being compromised by attacks on cloud infrastructure. In contrast, implementing IoT protection solutions locally can enhance user privacy, obviating the need to disclose personal and potentially sensitive data to other parties outside the local network. Such an edge protection approach can also improve robustness and stability of operations by eliminating the dependence on cloud services. However, a key open question is how feasible such locally run solutions are, and how well they work compared to cloud-based ones.

In this paper, we investigate the possibility of using threat detection and prevention algorithms on a home router, as well as measure quantitative implications on the router's memory (RAM), processing power (CPU), bandwidth (BW), and whether consuming these resources has any impact on the smooth operation of IoT devices and applications. Specifically, we demonstrate the feasibility of a completely local IoT protection solution, named SunBlock, which combines existing rule-based traffic filtering tools and machine-learning libraries to provide a resource-efficient prototype combining both intrusion detection (IDS) and intrusion prevention (IPS) systems. SunBlock operates on a typical home router, thus improving user privacy without the need for end users to share potentially sensitive data with cloud-based IDS/IPS solutions as well as bypassing the expense of their premium subscriptions. We conclude this paper by exposing our SunBlock prototype to a set of various IoT threats, demonstrating that it can detect a spectrum of threats more than twice as large as that detected by commercial IoT protection solutions [39]. The paper's code is available online [17].

Goals. The paper targets two main research questions (RQs) detailed below.
RQ1: How can we replace commercial cloud-based safeguards by a threat detection and prevention software running locally on a home router? We address this by designing a new cloudless threat detection/prevention approach (Sect. 3) and implementing it on popular off-the-shelf router hardware (Sect. 4).
RQ2: What is the performance of our solution in terms of overhead and threat detection capability? We address this by establishing a rigorous evaluation methodology (Sect. 5) and performing an extensive evaluation (Sect. 6) to evaluate: (*i*) the overhead of running AI and rule-based IDS locally in terms of router memory, processing power, and bandwidth; (*ii*) how successfully our cloudless solution detects popular attacks compared to existing cloud-based IoT security solutions.

Non-goals. The following topics are not the focus of this paper.
Novel ML/AI algorithms for anomaly detection. We assess existing ML methods

for their feasibility to run on constrained devices performing IPS functions. *Classification of end-user traffic, such as smartphones, tablets and PCs.* The main focus of the paper is on protecting smart home IoT devices.

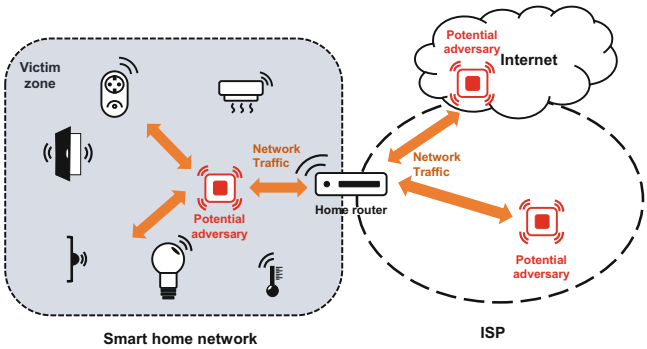

Fig. 1. Threat model diagram. Victim zone is a smart home with connected IoT devices. A potential adversary can analyze traffic and conduct privacy/security attacks from smart home, ISP and wider zones, such as the Internet.

2 Threat Model

In this paper we investigate the problem of protecting a typical home network from threats deriving from its IoT devices. We focus on instrumenting the home router providing connectivity to the IoT devices with existing algorithms (*i.e.,* combining IDS/IPS functions with a classic rule-based approach) to investigate the feasibility of running anomaly detection at the edge. This paper is not about proposing novel ML/AI algorithms or about surveying and comparing different AI/ML models for IoT anomaly detection, as done in previous studies [33,34,55].

Our threat model (Fig. 1) assumes a typical smart home deployment, with a home router connected to the Internet (WAN interface) creating and providing connectivity to a local smart home network ((W)LAN interface), where IoT devices are deployed. In this setting, we assume the victim to be a user of IoT devices, and the adversary any entity that can gain network access either from the WAN side (*e.g.,* an attacker in the ISP network or the Internet) or the (W)LAN side (*e.g.,* a compromised IoT device). In both cases the adversary can abuse network access and capture network data (such as traffic rates, protocols used, source/destination IP addresses, *etc.*) that can result in security and privacy threats for the victim. For example, network data can be used to conduct analyses to infer victim's sensitive data (*e.g.,* user activities [47]), and assess and exploit vulnerabilities [44]. The threats we focus on in this paper are the following:

Security: anomalous traffic, anomalous upload as well as various flooding (*e.g.,* DoS) and scanning attacks (e.g. port-scanning).

Privacy: PII exposure, unencrypted network/application data.

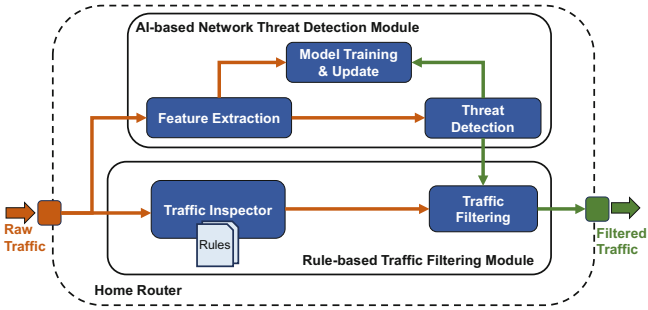

Fig. 2. SunBlock architecture running on a home router.

3 System Design

In this section we propose the design of SunBlock, *i.e.,* an approach for detecting IoT threats at a home router without relying on the cloud. Figure 2 shows the architecture of our system. SunBlock runs on a home router and is composed of: (*i*) a rule-based traffic filtering module; and (*ii*) an AI-based network threat detection module. Raw IoT traffic traversing the home router is inspected against a defined set of IPS rules [26], the features are extracted and fed to the trained threat detection model. Both rule-based and AI threat detection steps are performed in parallel and described in detail below.

3.1 Rule-Based Traffic Filtering Module

This module is responsible for blocking unwanted or potentially harmful traffic while allowing safe and necessary traffic to pass, thereby enhancing the overall security of the network. This module is composed of the following components.

Traffic Inspector: As network packets arrive, they are inspected against a defined set of rules at various levels: IP (*i.e.,* the source and destination IP addresses are checked), transport level (TCP or UDP source and destination ports), and/or at the application level (where the actual content of the packets is inspected, if the HTTP protocol is used, for example).

Traffic Filtering: Based on the set of pre-defined rules, allowed traffic passes further, while all the inspected traffic matching the blocking rules is dropped.

3.2 AI-Based Network Threat Detection Module

AI anomaly detection identifies irregular traffic patterns that could indicate potential security threats. The module consists of the following components.

Feature Extraction. Similar to rule-based traffic filtering, the first step includes collecting and processing raw network data, including information about source and destination IP addresses, port numbers, packet sizes, time durations, *etc.* The collected data is then transformed into a set of features, which are measurable properties or characteristics of the observed network traffic. These could include rate of data transmission, number of connections to a specific server, or frequency of a certain type of packet.

Model Training. An ML model is trained locally on the extracted dataset of network features. At the training phase, the model learns what 'normal' network traffic looks like.

Threat Detection. SunBlock uses the trained model for comparing live incoming network traffic to the 'normal' patterns learned during training. If the live traffic deviates significantly from normal patterns, SunBlock reports a threat.

Model Update. To support high threat detection accuracy, the model continues to learn and refine its understanding of what constitutes normal behavior.

4 Implementation

This section answers our first research question by describing a SunBlock prototype, demonstrating that IoT protection can run at the edge, on a home router.

4.1 Hardware

To demonstrate our approach, we use an off-the-shelf middle-range home router: a LinkSys WRT3200ACM [8]. The reason for this choice is its popularity and support for OpenWrt [12], a Linux-based OS targeting embedded devices. Due to the router's limited flash memory (256 MB), we use a USB stick to allocate 4GB and 512 MB partitions for AI libraries/dependencies and swap space, respectively.

4.2 Rule-Based Traffic Filtering

We use Snort3 [15] as a rule-based IDS/IPS. The reason for this choice is the extensive support for rule-based detection techniques, deep packet inspection and various customization options, making it flexibly well-suited for IoT threat prevention use cases. We prefer Snort3 over other solutions like Fail2ban [4], Zeek [21], and Suricata [18] due to its comprehensive rule-based detection techniques, extensive customization options, and flexibility. Other tools have either narrower focus (*e.g.,* Fail2Ban focuses on log-based IDS/IPS) or require more

scripting overhead to match Snort3's functionality. Besides utilizing the Snort3 community rules [16], we enhanced them by adding blocking logic for DoS, scanning attacks, and unencrypted HTTP traffic.

We configure Snort3 to run in NFQ IPS mode, which allows it to operate as an inline IPS. In this mode, Snort3 intercepts network traffic using Netfilter [10], a packet filtering and manipulation framework in the Linux kernel. In NFQ IPS inline mode, Snort3 receives network packets before they reach their target, thus allowing the system to react promptly to any suspicious traffic.

Upon receiving packets, SunBlock performs packet inspection by applying the pre-defined rules serving as criteria for identifying and blocking security and privacy threats. The inspection process involves protocol decoding, traffic analysis, and pattern matching against the defined rule set. Based on the results of the inspection, SunBlock applies rule-based actions. It determines if a packet matches any rules in its rule set and takes actions accordingly, such as dropping the packet or blocking the entire malicious traffic flow.

4.3 AI-Based Network Threat Detection

For our threat detection model, we utilize the netml library [11,55], a robust tool designed for feature extraction and threat detection in network traffic traces.

While the netml library is primarily intended for powerful machines with ample RAM and CPU resources, we successfully adapt it to be used on home routers. As netml is initially dedicated to x86 architectures, we adapt it for the ARMv7l architecture, commonly used in modern Wi-Fi routers and other embedded networked systems with limited resources. This portability enhances SunBlock's accessibility and flexibility, allowing it to run on a wider range of router configurations. We incorporate netml into a Docker image based on Debian version 10. The resulting image is open-source and accessible online [17]. After a series of deployment installations and tests, we selected the following versions of netml dependencies, a combination of which allows for the successful running of ML anomaly detection logic on the ARMv7l architecture: numba = 0.56.0, netaddr = 0.8.0, numpy = 1.22.0, pandas = 1.5.3, pyod = 1.0.9, scapy = 2.4.5 and scikit-learn = 0.24.1.

For anomaly detection tasks, SunBlock employs the One-Class Support Vector Machine (OCSVM) model in unsupervised mode with the packet interarrival time (IAT) feature. The OCSVM model is recognized for its outlier detection capabilities, lightweightness and efficacy [38,46,50]. The precision of the netml-based OCSVM implementation, used by SunBlock, has been extensively evaluated by the netml authors on various datasets, showing high precision for IoT anomaly classification tasks when using the IAT feature [55]. The Area Under the ROC Curve (AUC) metric is 0.95 for detecting infected devices, 0.96 for detecting novel devices, and 0.81–0.97 for detecting abnormal device activity. The chosen OCSVM configuration has been wrapped into a real-time threat detection logic having the following characteristics:

- Our system is configured to capture batches of 200 packets[1], thus capturing and classifying network traffic every batch.
- Our model is set up to learn from the most recent 7 days of network traffic. This period is chosen based on research indicating a decrease in model accuracy beyond a week without retraining [34].
- Upon detection of a threat, SunBlock uses iptables [7] for blocking malicious sources of traffic.

5 Evaluation Methodology

In this section we describe how we assess the performance of SunBlock in terms of overhead and threat detection capability. We first describe the IoT testbed we use for analyzing SunBlock, then how we emulate the threats, and finally describe the experiment setting and the evaluation metrics.

5.1 Evaluation Testbed

To assess the efficacy of SunBlock, we build an IoT testbed that emulates a typical smart home setting. According to previous research [31] that examined smart home traffic from 4,322 households, an average home tends to host around 10 IoT devices. Based on this number, and considering popular IoT device models and categories, we build our testbed using the 10 IoT devices detailed below.

- **Smart speaker:** Amazon Echo Spot, Google Home.
- **Video:** Amazon Fire TV.
- **Camera:** Yi Home Camera, Blink Home Camera.
- **Home automation:** Nest thermostat, TP-Link Kasa plug, WeMo plug, Gosund bulb, TP-Link Kasa bulb.

The IoT devices are performing daily activities within a 2.4 GHz WiFi home network in a location similar to a two-bedroom flat. We trigger the IoT device functionality by using a methodology similar to [40] for a period of one week.

5.2 Threats Emulation

For triggering the threats, we employ a Raspberry Pi Model B+ (RPi) as a threat generation node, equipped with 4 GB RAM and running a Debian image. The RPi is connected to the (W)LAN side of the home network along with the IoT devices. In order to check the SunBlock performance in realistic scenarios, the RPi used real attack scripts from an open GitHub repository [14] containing threat emulation scripts from recent research on the efficiency of IoT protection systems [39]. The threats we emulate are the following.

[1] We obtained this number empirically after extensive testing showing a good trade-off between ML accuracy and reaction time.

Anomalous Traffic. RPi acts on behalf of the Echo Spot device by spoofing its IP and MAC addresses. However, instead of sending typical Echo Spot traffic, the RPi injects previously captured traffic from the Google Home. We inject this traffic using tcpreplay [19], where we replace the IP and MAC addresses of the Google Home with Echo Spot's ones.

Anomalous Upload. We emulate anomalous upload behavior by making the RPi upload a video to an external unauthorized server while spoofing the IP and MAC addresses of the Yi home camera using the tcpreplay tool [19].

DoS Flooding. For SYN, UDP, DNS and HTTP flooding threats, the RPi uses suits of attacks from Kali Linux emulating a typical Mirai-infected device [23].

Privacy Script. The RPi acts on behalf of Google Home (by spoofing its IP and MAC addresses) and transmits unencrypted sensitive data, such as login details, passwords and other private data, to an external server through HTTP.

Scanning Attacks. We emulate OS and port scanning using Nmap [37]. This information can be utilized by a potential attacker to determine the most effective attack strategy for the detected OS, taking advantage of the open ports.

5.3 Experiments Description and Evaluation Metrics

Prior to executing the threat emulation scripts, we benchmark the resource utilization (CPU, RAM, and bandwidth) of the home router running SunBlock with respect to a router not running it. We continuously monitor these metrics using a Python-based script, which captures resource consumption data over a span of 3.5 h of regular IoT traffic.

To evaluate SunBlock performance, we time the training process under four distinct conditions: both modules operational, only the AI-based module active, only the Rule-based module active, and an unprotected scenario. Under each condition, the model underwent five training iterations. A background script records CPU usage, RAM (actual RAM used by the system, excluding buffers/cache), bandwidth consumption, and the start and end times of each training instance.

We conduct each attack for 100 s in 10 separate iterations. After each attack, we reset the network to its normal operational state before we start the next iteration. The RPi logs both the start and end times for each individual attack. We conduct all experiments in a home network environment equipped with the SunBlock router. We then repeat the same set of experiments in a network with the same router, but with SunBlock disabled.

To measure resource utilization of the home router, both with and without SunBlock, we execute a background Python-based script during each attack, which records the utilization of *CPU, RAM, and BW* every second.

To measure *threat prevention latency*, we execute a script to capture SunBlock's notifications, which provided timestamps and types of blocked threats. We calculate the prevention latency as the time between the start of the attack and the arrival of a SunBlock's notification indicating a blocked threat.

6 Evaluation

In this section we answer our second research question by evaluating the performance of SunBlock in terms of overhead and threat detection capability.

6.1 Performance Overheads

Table 1 shows model training performance overhead in distinct conditions: fully protected (Rule-based and AI-based enabled modes), partially protected (either Rule-based or AI-based mode is enabled), and unprotected (disabled SunBlock).

Table 1. ML training time and resource usage for various protection levels.

Protection Level	CPU (%)	RAM (MB)	swap (MB)	Training Time (s)
Rule-based & AI-based	18 ± 3	444 ± 4	296 ± 21	924 ± 253
AI-based only	26 ± 2	442 ± 6	197 ± 28	429 ± 171
Rule-based only	32 ± 4	423 ± 9	132 ± 20	180 ± 22
Unprotected	39 ± 2	410 ± 3	55 ± 1	113 ± 10

When deployed in tandem, Rule-based and AI-based modules constitute the most secure mode. Although the fully protected mode takes the longest time for training—up to 20 min—it has the lowest CPU utilization among all modes. Compared to AI-based/Rule-based only modes, the fully protected mode has increased training time by a factor of 1.5–6.5, and up to a factor of 10.5 when compared to the unprotected mode. However, the duration of fully protected training remains within an acceptable range and can be scheduled during off-peak hours to minimize disruptions.

Memory usage is also critical for a home router. During the fully protected mode, swap memory usage peaks at around 300MB, which is manageable for a typical home router equipped with sufficient flash memory (either internally or through an external USB stick). It is noticeable that swap is required only for model training and is not used in SunBlock's daily behavior. Swap should be allocated as a separate partition for its isolation from router's storage memory.

Figure 3 provides a performance comparison between the router in SunBlock's protected mode and the same router in the unprotected mode, during their routine operation without network threats. SunBlock exhibits a wider CPU utilization range compared to the unprotected router; however, the utilization peaks at only ∼15%. The median CPU utilization remains at an approximate 3%, thus maintaining a negligible impact on overall performance. SunBlock has 1.7 times higher RAM consumption than the unprotected router. While this increment is notable, the overall RAM usage remains well within the router's capacity, leaving about 30% of free RAM. The bandwidth usage is the same in both modes

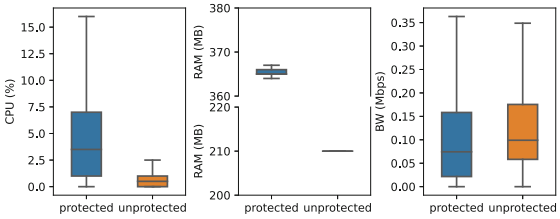

Fig. 3. Resource usage in protected vs. unprotected mode for regular IoT traffic.

due to the same traffic conditions under which both protected and unprotected modes were tested.

Figure 4 illustrates the Empirical Cumulative Distribution Functions (ECDFs) representing the times taken by SunBlock to prevent threat types listed in Sect. 5.2. It can be observed that most threats are blocked within ∼5 s. However, SunBlock requires more time to detect anomaly threats, namely ∼10 s for anomalous upload and ∼50 s for anomalous traffic.

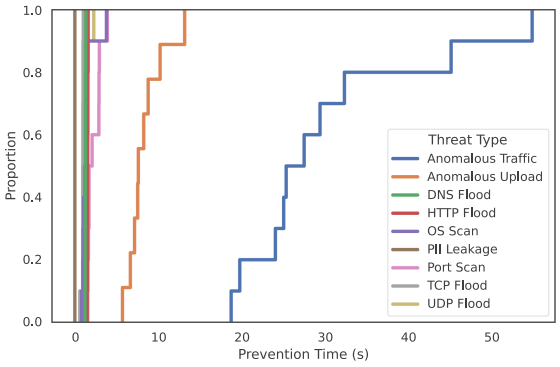

Fig. 4. ECDFs of SunBlock's prevention time per each attack.

Figures 5, 6 show RAM, CPU and BW overhead of SunBlock under a number of traffic-heavy attacks (DNS, HTTP, SYN, UDP flooding and anomalous upload), compared to an unprotected mode.

As anticipated, SunBlock exhibits an elevated CPU/RAM usage due to the operational load imposed by Rule-based and AI-based modules actively performing threat mitigation tasks. SunBlock's median CPU consumption varies between 3–40%, while the RAM usage remains fairly stable within a range of 365–375 MB. Conversely, an unprotected router exhibits significantly lower RAM/CPU usage, ∼220 MB of RAM and 1–25% of median CPU utilization.

The protective effects of SunBlock are evident in the bandwidth consumption under heavy DoS attacks. The box plots indicate a substantial bandwidth

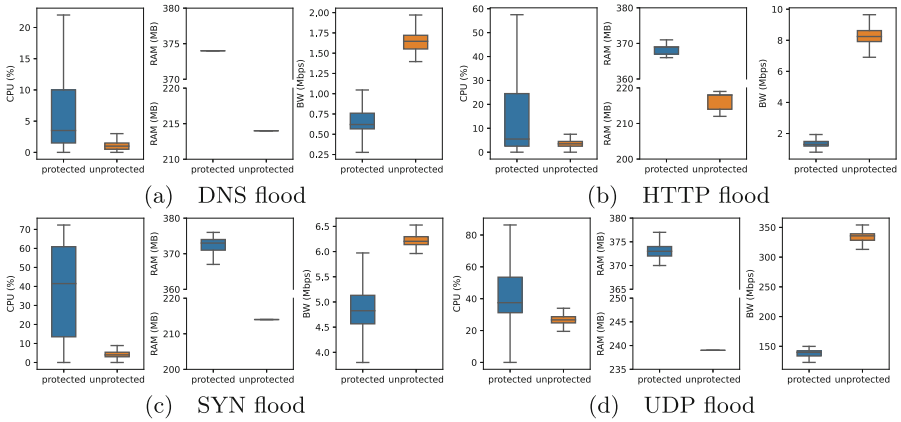

Fig. 5. Resource usage in protected vs. unprotected mode under DoS attacks.

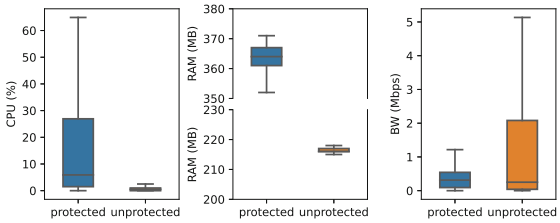

Fig. 6. Resource usage in protected vs. unprotected mode for anomalous upload.

reduction of up to ∼8 times during intense DoS attacks, with the strongest reductions during HTTP and UDP flooding. During the attacks, SunBlock maintains an adequate buffer of resources, with 50% of CPU and 30% of RAM always available.

6.2 Threat Detection Capability

Table 2 presents a comprehensive comparison of SunBlock's threat detection capability with existing IoT protection solutions. The data on threat detection capabilities of existing solutions is sourced from prior comparative research [39], which evaluated a number of most popular IoT protection solutions (Avira [20], Bitdefender [2], F-Secure [3], Fingbox [5], Firewalla [6], McAfee [9], RATtrap [13], and TrendMicro [1]) using the same threat generation scripts as our evaluation. For the sake of brevity, the table shows merged IoT threat detection results of these protection systems. A check mark against a threat means at least one system can detect it. Results show that SunBlock outperforms IoT protection solutions in terms of threat detection capabilities.

Table 2. Threats detected by existing IoT solutions vs. detected by SunBlock.

Threat	IoT Protection Systems	SunBlock
Anomalous Traffic	✗	✓
Anomalous Upload	✗	✓
SYN Flooding	✓	✓
UDP Flooding	✗	✓
DNS Flooding	✗	✓
HTTP Flooding	✓	✓
Port Scanning	✓	✓
OS Scanning	✓	✓
PII Leakage	✗	✓

7 Limitations and Future Work

Hardware. Our approach was tested on an off-the-shelf router with 512MB of RAM. We choose this since full-featured routers offered by the most popular US ISPs have more memory than that (*e.g.,* the Xfinity xFi XB6 from Comcast has 768 MB [42] and the CR1000A [53] from Verizon has 2 GB). However, some older routers and lower-tier router offerings may still have a lower amount of RAM. In such case, our approach would be slower due to requiring swap space. Future improvements involve developing a more efficient Docker image for ML anomaly detection by utilizing lighter Linux distributions such as Arch Linux, which require more custom configuration compared to Debian-based images. Additionally, it is worthwhile to monitor emerging ML/AI methods and associated compression and pruning techniques that may reduce RAM and CPU usage while maintaining high classification performance.

Zero-Day Threat Protection. Zero-day threats pose a common challenge for ML systems, as learning models require time to gather relevant training data during their first run. Shortening the initial training period can reduce exposure but might compromise accuracy, necessitating more user input for custom classification. This is not always feasible as home users are not typically security experts. SunBlock's Rule-based Traffic Filtering, functioning from day one, reduces the risk of zero-day exposure by protecting against most severe threats, such as flooding and scanning. Additionally, other security measures during the initial setup may include factory resetting all present IoT devices, ensuring they are updated with the latest software versions, and, if feasible, placing them on a separate network segment with limited Internet access.

Model Updates. Since our approach is cloudless, it does not include external (cloud-based) means of updates. The capability of our approach to learn and self-adapt partially address this limitation; however, faster updates can be achieved by allowing the periodic download of updated Snort community rules [16] and/or

by building and sharing AI models in a privacy-preserving crowdsourced way (*e.g.,* as discussed in [25,56]).

Configuration. As discussed before, default parameters of our approach were chosen empirically for our prototype. However, depending on the particularities of deployment scenario and user needs, such parameters may need to be tuned to achieve the desired trade-off between false negative and false positive threat reporting. This limitation can be addressed, for example, by using a interactive ML approaches [29,54], where the user is asked to confirm what to do for each threat until the system learns how to make this decision on its own.

8 Related Work

The growing privacy and security issues in the consumer IoT domain have prompted the creation of various tools [36,40] designed to block malicious traffic originating from IoT devices. Despite the promising advances these tools represent, there remains an area for their potential improvement by incorporating AI for the detection of threats.

Other attempts [28,43,45] to shift IoT protection to the edge have limitations. Implementations of [28,45] rely solely on classic rule-based traffic filtering, which is less sensitive to anomalous traffic threats and requires additional AI-based classification. The evaluations in [28,43,45] were conducted on non-router-specific superior hardware and may not reflect consumer router performance.

There have been recent introductions of AI-based methodologies dedicated to device identification tasks at the edge [33,35,41,52]. However, they alone are insufficient for comprehensive protection against major IoT threats and would benefit from rule-based and AI-based threat detection techniques.

Numerous studies have examined IoT threats, assessing smart home vulnerabilities, attack scenarios, and defense strategies [22,24,30]. Recent research [39] introduced a methodology to assess the efficacy of current IoT protection solutions in responding to prevalent security and privacy threats. The associated threat emulation scripts and benchmarks were adopted in our study for testing the effectiveness of our prototype against popular IoT protection solutions.

9 Discussion and Conclusion

The growing ubiquity of IoT devices in residential settings is transforming our homes into sophisticated, interconnected digital environments. As such, the importance of robust and proactive protection measures for these devices is vital, ensuring not only their functionality but also the privacy and security of the vast amounts of personal data they handle.

Our research reveals that IoT threats can be rapidly mitigated on a home router, equipped with ML/AI anomaly detection and rule-based traffic filtering algorithms. Most types of threats are promptly identified and blocked within the first 5 s. Nonetheless, certain threats such as anomalous traffic and anomalous

upload require a longer time due to their initial similarity with normal network behavior of other IoT devices in the network. The presented local threat detection approach eliminates the need for dependence on cloud-based IoT security solutions and thus blocks extra channels of potential PII and other user-sensitive data exposure. It is important to note that integrating ML with rule-based traffic analysis in our off-the-shelf router (which has inferior hardware with respect to full-featured routers currently offered by top ISPs in the US [42,53]) cause a slight increase in CPU utilization (up to ∼15% with an average use of ∼3%) and RAM consumption (by ∼150 MB) during normal operation, without affecting main router functions. Our plans for further enhancements involve intensive beta testing and precise performance benchmarking against existing cloud-based security solutions.

Acknowledgements. We thank the anonymous reviewers and our shepherd Roland van Rijswijk-Deij for their constructive and insightful feedback. This work was supported by the EPSRC Open Plus Fellowship (EP/W005271/1), the EPSRC PETRAS grant (EP/S035362/1), and the NSF ProperData award (SaTC-1955227).

References

1. Asus (TrendMicro). https://www.asus.com/Content/AiProtection/. Accessed 03 Nov 2023
2. Bitdefender Box 2. https://www.bitdefender.com/smart-home/#products. Accessed 03 Nov 2023
3. F-Secure. https://www.f-secure.com/gb-en/home/products/sense. Accessed 03 Nov 2023
4. Fail2ban. https://www.fail2ban.org/wiki/index.php/Main_Page. Accessed 03 Nov 2023
5. Fingbox. https://www.fing.com/products/fingbox. Accessed 03 Nov 2023
6. Firewalla. https://firewalla.com/. Accessed 03 Nov 2023
7. iptables. https://linux.die.net/man/8/iptables. Accessed 03 Nov 2023
8. LinkSys — WRT3200ACM Data Sheet. https://downloads.linksys.com/downloads/datasheet/WRT3200ACM_WiFiRouter_EN.pdf. Accessed 03 Nov 2023
9. McAfee Secure Home Platform. https://www.mcafee.com/support/?page=shell&shell=article-view&locale=en-US&articleId=TS102712. Accessed 03 Nov 2023
10. netfilter. https://www.netfilter.org/. Accessed 03 Nov 2023
11. netml. https://github.com/noise-lab/netml. Accessed 03 Nov 2023
12. OpenWrt. https://openwrt.org/. Accessed 03 Nov 2023
13. RATtrap. https://www.myrattrap.com/. Accessed 03 Nov 2023
14. Safeguards study: threat simulation scripts. https://github.com/IoTrim/safeguards-study. Accessed 03 Nov 2023
15. Snort 3. https://www.snort.org/snort3. Accessed 03 Nov 2023
16. Snort3 community rules. https://snort.org/downloads/community/snort3-community-rules.tar.gz. Accessed 03 Nov 2023

17. SunBlock project page. https://github.com/SunBlock-IoT/SunBlock_router. Accessed 11 Jan 2023
18. Suricata. https://suricata.io/. Accessed 03 Nov 2023
19. Tcpreplay Official Site. https://tcpreplay.appneta.com/. Accessed 03 Nov 2023
20. TP-Link HomeShield (Avira). https://www.tp-link.com/us/homeshield/. Accessed 03 Nov 2023
21. Zeek. https://zeek.org/. Accessed 03 Nov 2023
22. Alrawi, O., Lever, C., Antonakakis, M., Monrose, F.: SoK: security evaluation of home-based IoT deployments. In: 2019 IEEE Symposium on Security and Privacy (SP), pp. 1362–1380 (2019). https://doi.org/10.1109/SP.2019.00013
23. Antonakakis, M., et al.: Understanding the Mirai Botnet. In: 26th USENIX Security Symposium (USENIX Security 2017), Vancouver, BC, pp. 1093–1110. USENIX Association (2017). https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis
24. Babun, L., Denney, K., Celik, Z.B., McDaniel, P., Uluagac, A.S.: A survey on IoT platforms: communication, security, and privacy perspectives. Comput. Netw. **192**, 108040 (2021). https://doi.org/10.1016/j.comnet.2021.108040. https://www.sciencedirect.com/science/article/pii/S1389128621001444
25. Briggs, C., Fan, Z., Andras, P.: A review of privacy-preserving federated learning for the internet-of-things. In: Federated Learning Systems: Towards Next-Generation AI, pp. 21–50 (2021)
26. Chakrabarti, S., Chakraborty, M., Mukhopadhyay, I.: Study of snort-based IDS. In: ICWET 2010, pp. 43–47. Association for Computing Machinery, New York (2010). https://doi.org/10.1145/1741906.1741914
27. Conti, M., Nati, M., Rotundo, E., Spolaor, R.: Mind the plug! Laptop-user recognition through power consumption. In: Proceedings of the 2nd ACM International Workshop on IoT Privacy, Trust, and Security, IoTPTS 2016, pp. 37–44. Association for Computing Machinery, New York (2016). https://doi.org/10.1145/2899007.2899009
28. Dua, A., Tyagi, V., Patel, N., Mehtre, B.: IISR: a secure router for IoT networks. In: 2019 4th International Conference on Information Systems and Computer Networks (ISCON), pp. 636–643 (2019). https://doi.org/10.1109/ISCON47742.2019.9036313
29. Dudley, J.J., Kristensson, P.O.: A review of user interface design for interactive machine learning. ACM Trans. Interact. Intell. Syst. (TiiS) **8**(2), 1–37 (2018)
30. He, W., et al.: SoK: context sensing for access control in the adversarial home IoT. In: 2021 IEEE European Symposium on Security and Privacy (EuroS&P), pp. 37–53 (2021). https://doi.org/10.1109/EuroSP51992.2021.00014
31. Huang, D.Y., Apthorpe, N., Li, F., Acar, G., Feamster, N.: IoT inspector: crowdsourcing labeled network traffic from smart home devices at scale. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. **4**(2) (2020). https://doi.org/10.1145/3397333
32. Karale, A.: The challenges of iot addressing security, ethics, privacy, and laws. Internet Things **15**, 100420 (2021). https://doi.org/10.1016/j.iot.2021.100420. https://www.sciencedirect.com/science/article/pii/S2542660521000640
33. Kolcun, R., et al.: Revisiting IoT device identification. In: Bajpai, V., Haddadi, H., Hohlfeld, O. (eds.) 5th Network Traffic Measurement and Analysis Conference, TMA 2021, Virtual Event, 14–15 September 2021. IFIP (2021). http://dl.ifip.org/db/conf/tma/tma2021/tma2021-paper6.pdf
34. Kolcun, R., et al.: The Case for Retraining of ML Models for IoT Device Identification at the Edge. arXiv preprint (2020). https://arxiv.org/abs/2011.08605

35. Kotak, J., Elovici, Y.: IoT device identification using deep learning. In: Herrero, Á., Cambra, C., Urda, D., Sedano, J., Quintián, H., Corchado, E. (eds.) CISIS 2020. Advances in Intelligent Systems and Computing, vol. 1267, pp. 76–86. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-57805-3_8
36. Lastdrager, E., Hesselman, C., Jansen, J., Davids, M.: Protecting home networks from insecure IoT devices. In: NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, p. 1–6. IEEE Press (2020). https://doi.org/10.1109/NOMS47738.2020.9110419
37. Lyon, G.F.: Nmap network scanning: the official Nmap project guide to network discovery and security scanning. Insecure. Com LLC (US) (2008)
38. Mahdavinejad, M.S., Rezvan, M., Barekatain, M., Adibi, P., Barnaghi, P., Sheth, A.P.: Machine learning for internet of things data analysis: a survey. Digit. Commun. Netw. **4**(3), 161–175 (2018). https://doi.org/10.1016/j.dcan.2017.10.002. https://www.sciencedirect.com/science/article/pii/S235286481730247X
39. Mandalari, A., Haddadi, H., Dubois, D.J., Choffnes, D.: Protected or porous: a comparative analysis of threat detection capability of IoT safeguards. In: 2023 2023 IEEE Symposium on Security and Privacy (SP) (SP), pp. 3061–3078. IEEE Computer Society, Los Alamitos (2023). https://doi.org/10.1109/SP46215.2023.00151. https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.00151
40. Mandalari, A.M., Dubois, D.J., Kolcun, R., Paracha, M.T., Haddadi, H., Choffnes, D.: Blocking without Breaking: Identification and Mitigation of Non-Essential IoT Traffic (2021)
41. Meidan, Y., et al.: ProfilIoT: a machine learning approach for IoT device identification based on network traffic analysis. In: Proceedings of the Symposium on Applied Computing, SAC 2017, pp. 506–509. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3019612.3019878
42. Modems, A.: Comcast Infinity xFi XB6 Review (2023). https://approvedmodems.org/xfinity-xfi-xb6-review/. Accessed 03 Nov 2023
43. Palmese, F., Redondi, A.E., Cesana, M.: Feature-sniffer: enabling IoT forensics in OpenWrt based Wi-Fi access points. In: 2022 IEEE 8th World Forum on Internet of Things (WF-IoT), pp. 1–6. IEEE (2022)
44. Paracha, M.T., Dubois, D.J., Vallina-Rodriguez, N., Choffnes, D.: IoTLS: understanding TLS usage in consumer IoT devices. In: Proceedings of the Internet Measurement Conference (2021)
45. Patel, N., Mehtre, B., Wankar, R.: A snort-based secure edge router for smart home. Int. J. Sens. Netw. **41**(1), 42–59 (2023). https://doi.org/10.1504/IJSNET.2023.128505. https://www.inderscienceonline.com/doi/abs/10.1504/IJSNET.2023.128505
46. Razzak, I., Zafar, K., Imran, M., Xu, G.: Randomized nonlinear one-class support vector machines with bounded loss function to detect of outliers for large scale IoT data. Future Gener. Comput. Syst. **112**, 715–723 (2020). https://doi.org/10.1016/j.future.2020.05.045. https://www.sciencedirect.com/science/article/pii/S0167739X19313913
47. Ren, J., Dubois, D.J., Choffnes, D., Mandalari, A.M., Kolcun, R., Haddadi, H.: Information exposure for consumer IoT devices: a multidimensional, network-informed measurement approach. In: Proceedings of the Internet Measurement Conference (IMC) (2019)
48. Sadek, I., Rehman, S.U., Codjo, J., Abdulrazak, B.: Privacy and security of IoT based healthcare systems: concerns, solutions, and recommendations. In: Pagán, J., Mokhtari, M., Aloulou, H., Abdulrazak, B., Cabrera, M. (eds.) ICOST 2019.

LNCS, vol. 11862, pp. 3–17. Springer, Heidelberg (2019). https://doi.org/10.1007/978-3-030-32785-9_1
49. Setayeshfar, O., et al.: Privacy invasion via smart-home hub in personal area networks. Pervasive Mob. Comput. **85**, 101675 (2022). https://doi.org/10.1016/j.pmcj.2022.101675
50. Shorman, A., Faris, H., Aljarah, I.: Unsupervised intelligent system based on one class support vector machine and Grey Wolf optimization for IoT botnet detection. J. Ambient Intell. Humaniz. Comput. **11**, 2809–2825 (2020). https://doi.org/10.1007/s12652-019-01387-y
51. Swessi, D., Idoudi, H.: A survey on internet-of-things security: threats and emerging countermeasures. Wirel. Pers. Commun. **124**(2), 1557–1592 (2022). https://doi.org/10.1007/s11277-021-09420-0
52. Thompson, O., Mandalari, A.M., Haddadi, H.: Rapid IoT device identification at the edge. In: Proceedings of the 2nd ACM International Workshop on Distributed Machine Learning, DistributedML 2021, pp. 22–28. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3488659.3493777
53. Verizon: VerizonRouter CR1000A Datasheet (2023). https://www.verizon.com/supportresources/content/dam/verizon/support/consumer/documents/internet/verizon-router_datasheet.pdf. Accessed 03 Nov 2023
54. Wu, X., Xiao, L., Sun, Y., Zhang, J., Ma, T., He, L.: A survey of human-in-the-loop for machine learning. Futur. Gener. Comput. Syst. **135**, 364–381 (2022)
55. Yang, K., Kpotufe, S., Feamster, N.: A Comparative Study of Network Traffic Representations for Novelty Detection. arXiv preprint (2020). https://arxiv.org/abs/2006.16993v1
56. Zhou, C., Fu, A., Yu, S., Yang, W., Wang, H., Zhang, Y.: Privacy-preserving federated learning in fog computing. IEEE Internet Things J. **7**(11), 10782–10793 (2020)