Author's copy (February 11, 2022)
© 2022 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

A Formal Analysis of the Efficacy of Rebooting as a Countermeasure Against IoT Botnets

Alvi Jawad, Luke Newton, Ashraf Matrawy, Jason Jaskolka
Systems and Computer Engineering, Carleton University
1125 Colonel By Drive, Ottawa, ON K1S 5B6, Canada
{alvi.jawad, luke.newton, ashraf.matrawy, jason.jaskolka}@carleton.ca

***Abstract*— The Mirai botnet revolutionized the idea of IoT botnets by infecting numerous vulnerable IoT devices in 2016, leading to the rise of many Mirai variants and imitators that plague the current IoT ecosystem. Studying the botnet infection process can greatly aid us in understanding IoT botnet capabilities and the efficacy of currently available countermeasures. However, analyzing IoT botnets is difficult due to their massive scale and the numerous existing heterogeneous IoT devices that can be targeted for infection. In this paper, we model and simulate the dynamic behavior of a Mirai-like botnet infrastructure and various IoT device categories as a network of timed automata in UPPAAL-SMC. To determine the feasibility of rebooting as a countermeasure against botnets, we examine the effectiveness of rebooting on various IoT device networks. The resulting analysis provides a solid understanding of the efficacy and feasibility of rebooting on active and dormant botnet propagation processes.**

***Index Terms*—Network Security, Internet of Things (IoT), Botnets, Mirai, Modeling, Timed Automata, UPPAAL**

I. Introduction

The Internet of Things (IoT), with the proliferation of numerous vulnerable IoT devices, has become a tempting target for botmasters [1]. These malicious device herders use both inherent and user-induced vulnerabilities present in exposed IoT end-points to take control and use them in various (usually nefarious) purposes. One prime example is the Mirai botnet that managed to remotely control nearly half a million poorly configured IoT devices to stage massive distributed denial-of-service (DDoS) attacks in 2016 [2]. More recently, the largest ever DDoS attack was targeted at Amazon Web Services in February 2020, which saw sustained traffic at 2.3 Tbps [3]. Moreover, botnets have been known to be involved in spamming, malware dissemination, spying, port scanning, firmware corruption, and many more [2], [4], [5]. The last few years have seen a surge in the frequency and sophistication of botnet attacks, resulting in varying levels of impact on the individual devices, services, and the network alike [1].

Prominent botnet infrastructures (e.g., Mirai) and their infection process have been studied in great detail by the security community [2], [6]. However, the analyses performed are mostly evidentiary, where the data collection requires months of monitoring different network vantage points such as internet-wide scanning, telnet honeypots, logs of botnet attack commands, and passive DNS traffic [6]. Also, many gray areas still remain in determining all the distinct types of infected devices; the conclusion that the primary targets of Mirai were routers, cameras, and DVRs, is based on all the devices that could be identified, which for many ports, remained below 5% [6]. Therefore, we wish to draw more attention to not individual devices, but modeling IoT device categories (groups of IoT device classes) to examine and compare the efficacy of the botnet infection process on them.

Rebooting has been suggested in the literature as a countermeasure to curb the spread of Mirai [7]. The original Mirai binary did not persist across device reboots, essentially allowing a system reboot to restore the regular device behavior [6]. The core theme of our work is to expand upon this idea by doing a timed analysis of botnets on a large-scale network of different IoT device categories. The goal is to investigate whether rebooting alone is capable of preventing the attacker from amassing and/or maintaining their attacking capability over a certain period. If so, we wish to see what rate of rebooting is the most effective and whether such a frequency is feasible for modern IoT devices.

Contributions: Our main contributions are as follows:

- Development of a highly scalable formal model of a Mirai-like botnet and different IoT device categories using timed automata [8] in UPPAAL [9].
- A formal analysis of whether rebooting can be adopted as a feasible measure to thwart the spread of botnets instead of extensive security controls impractical for many resource-constrained IoT devices.

Our simulations with large IoT networks explore whether specific device categories are inherently more resilient to botnet infection than the others. In doing so, we examine the behavior of hibernating and non-hibernating botnets in both a fixed and variable network speed scenario.

The rest of this paper is structured as follows. Section II outlines the significance of Mirai and its underlying infection process. Section III discusses our contributions in contrast to the existing literature. Sections IV and V, respectively, present formal models of the Mirai botnet infrastructure and different IoT device categories. Section VI details model configurations and our experimental results whereas Section VII discusses the implications. Finally, Section VIII concludes our work.

II. The Significance of Mirai

In this section, we summarize the impact of Mirai and its successors and detail their underlying infection process.

A. *Mirai and its Successors*

Mirai, first identified in August 2016, is a prime example of how seemingly simple IoT vulnerabilities can be exploited to amass a large enough botnet to lead to catastrophic impacts. It launched large-scale Distributed Denial of Service (DDoS) attacks against security website KrebsOnSecurity and French cloud computing company OVH, with malicious traffic peaking at 620 Gbps and 1.1 Tbps, respectively [1]. At its peak, Mirai may have held up to 600,000 connected devices, hinting at a small picture of what could happen if all of these infected devices were used simultaneously to perform an attack [6].

The source code release[1] of Mirai in 2016 started a revolution, leading to considerable growth in different Mirai variants and imitators worldwide [1]. Many of these successors have adapted the relatively simple infection process provided by Mirai to perform more sophisticated attacks, often with varying architectural elements [2]. Some notable exploits include a 54-hour long app-layer DDoS attack on a US college [10], a windows-based spreader [11], router enslavement and denial of internet access for 900,000 customers [12], and bitcoin mining functionality [13]. All these exploits point to the fact that a better understanding of the origin, i.e., Mirai and its infection process, is crucial in developing suitable countermeasures for current and future Mirai successors.

B. *Understanding the Mirai Infection Process*

Mirai falls under the category of classical centralized botnets that revolve around a single central Command and Control (C&C) server to spread [14]. Fig. 1 illustrates the primary components of the centralized botnet infrastructure (depicted in violet), i.e., a *C&C server*, a *Report server*, and a *Loader program* [6] under the control of the *Mirai Botmaster*. Infected devices, also known as bots (depicted in red), are the primary threat actors that increase the botnet population by actively seeking out other vulnerable connected devices in the network to infect. The targets of Mirai (depicted in blue) are vulnerable IoT devices (primarily routers, DVRs, and cameras [6]) with default or common username-password combinations.

We outline the Mirai infection process in six simplified steps [1], [2], [6]. (1) The botmaster begins the infection process by setting up the remote C&C server and the Report server. (2) The C&C server activates the first bot, one that is infected from the beginning, to initiate the scan-attack-report cycle of a bot. (3) The bot statelessly scans the network for open Telnet ports at randomly selected IP addresses. After a successful identification, it initiates a brute force remote-login attempt by randomly selecting 10 out of the 62 credentials hard-coded in the Mirai binary. (4) If a brute force attack is successful, the attacking bot reports the victim *id* (device IP and the credentials that worked) to the Report server. (5) If the reported device is not part of the botnet, the report server adds it to its records and starts the Loader to load the malware binary into the reported vulnerable device. (6) The Loader determines the architecture of the target device and uploads a hardware-specific malware, transforming it into another bot. Steps 3 to 6 continue until the botnet is large enough for the attacker's purpose.

[1]https://github.com/jgamblin/Mirai-Source-Code

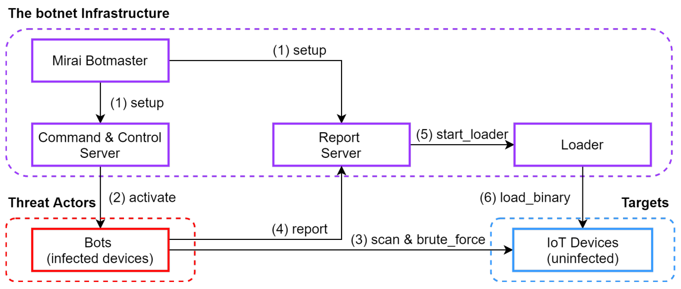


Fig. 1. Mirai infection process

III. Related Work

The basis of the developed model is formed by in-depth studies investigating each component of the Mirai botnet infrastructure [1], [6], and specific bot activities in propagating the malware [2]. Agent-oriented Petri nets have been used to model the Mirai and Hajime botnets in [7], [15], exploring the possibility of using rebooting and the existence of Hajime to reduce the Mirai infection rate. Both botnets are, however, modeled only as black box entities without the underlying infrastructure, and the scalability is fairly limited as the authors fail to extend the network beyond 25 nodes. Epidemiological approaches, e.g., [16] and [17], effectively model time, but are limited in the amount of detail that can be modeled. Other attempts involve the use of game theory, machine learning, and economic models [18]. Our approach aims to combine the benefits of formal and epidemiological models by considering time and concurrency while also presenting an extensible model with a fine-grained level of adjustable detail.

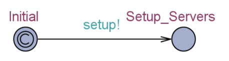
Other approaches to study botnets involve collecting data from real-world network traffic and devices, including the use of honeypots [4], [6], DNS traffic logs [6], [19], tracing DDoS attacks back to their source [6], and scanning for devices with bot-like behavior [6], [19], [20], [21]. These methods have the benefit of generating real-world data but the data collection process requires time and the presence of active botnets.

IV. Modeling the Mirai Infection Process

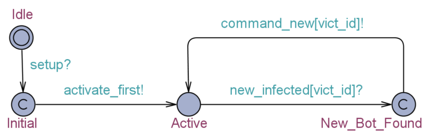
In this section, we present the necessary background and the modeling process of a Mirai-like botnet infrastructure.

A. *Timed Automata & UPPAAL*

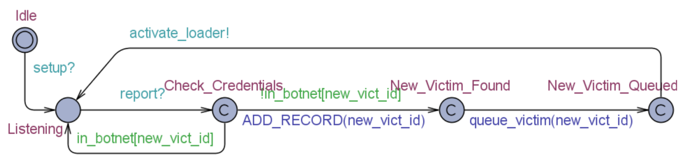
Timed automata is a hybrid mathematical modeling formalism in which a finite set of real-valued clocks are used to represent continuous time in a discrete-event system [8]. A timed automaton, a finite-state machine extended with clock variables, is a tuple (L, l_0, C, A, E, I) [9], where L is a set



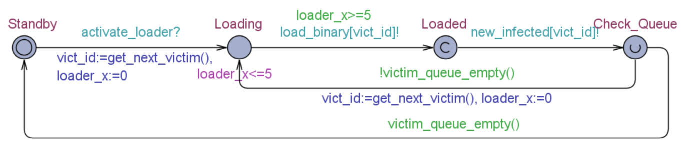
(a) The Botmaster Automaton (BM)



(b) The CnC Server Automaton (CNC)



(c) The Report Server Automaton (RPT)



(d) The Loader Automaton (LDR)

Fig. 2. The networked botnet architecture model adapted from [1], [6]

of locations, $l_0 \in L$ is the initial location, C is the set of clocks, A is a set of actions, co-actions, and the internal τ-action, $E \subseteq L \times A \times B(C) \times 2^C \times L$ is a set of edges between locations with an action, a boolean guard, a set of clocks to be reset, and $I : L \rightarrow B(C)$ assigns invariants to locations. The selection of a timed modeling technique is crucial since time is a critical factor for both bot activities (e.g., scanning) as well as individual device behaviors (e.g., rebooting).

The choice of timed automata allows us to use UPPAAL for our modeling activities. UPPAAL, an integrated tool environment, supports the modeling and verification of real-time systems as networks of timed automata [9]. The UPPAAL modeling language extends timed automata with various additional capabilities, including the capability to model binary and broadcast channel synchronizations [9]. These extensions allow one to model real-time networks with bounded delays, proven by successful applications in different time-critical case studies involving communication protocols [22], [23] or remote control of industrial processes [24]. We use UPPAAL-SMC (v4.1.25-5) available under a free academic license[2].

B. *Modeling the Botnet Infrastructure*

We start by modeling each key entity of the Mirai botnet infrastructure as an individual timed automaton in Fig. 2. Each automaton engages in multiple channel synchronizations to constitute a network of timed automata, reflecting the behavior of a botnet in a real-time network. A detailed explanation of the notations used to create timed automata models of systems in UPPAAL can be found in [9].

We use the Loader automaton (LDR) of Fig. 2(d) to briefly explain the modeling process. LDR, initially in the Standby state, makes a transition to the Loading state after receiving an activate_loader? channel synchronization from the Report server automaton (RPT) while getting the victim id (vict_id:=get_next_victim()) from a queue. In the Loading state, LDR communicates with the victim by sending out a load_binary[vict_id]! synchronization, and starts loading a hardware-specific Mirai binary. After a successful infection (assumed to take five time units), LDR moves to the Loaded state and informs the C&C server automaton (CNC) about the newly infected victim (new_infected[vict_id]!). From the Check_Queue state, LDR continues with the next infection or goes back to its idle behavior (Standby) depending on the state (victim_queue_empty()) of the queue.

[2]https://uppaal.org/downloads/#academic-licenses/

V. Modeling Various Device Categories

In this section, we discuss different IoT device categories and present two device category models involved in our study.

A. *IoT Device Categories*

A closer inspection of Mirai over five months reveals that the majority (for some ports, over 95%) of device types targeted by Mirai still remain unidentified [6]. An exhaustive identification is hardly practical and would require a large amount of time and resources. Instead, we may consider modeling groups of known IoT devices classes to examine how vulnerable each group is to botnet infections. Table I defines two IoT device categories by grouping different IoT device classes based on network connectivity. The device categories are adapted from the classification provided in RFC 7228 [25].

TABLE I
Categories of IoT Devices [25]

Category Name	Classes Based on Energy Limitations	Classes based on Communication Capability
Non-rebootable	E9, E2	P9, P1
Rebootable	E1, E0	P0

First, we can look at the *Non-rebootable* device category, representative of IoT devices that, once connected, stay connected to the network. Examples of this category include all devices of class E9 (mains-powered devices), E2 (lifetime-energy limited devices), P9 (always-on devices), and P1(low power connected devices) [25]. Similarly, we can model the behavior of *Rebootable* device categories that can reboot or reattach to the network either periodically or manually with user intervention. Devices of class E1 (period-energy limited devices), E0 (event-energy limited devices), and P0 (devices that are normally turned off, only attached to the network when needed) fall under this category and can be modeled similarly with simple attunement of timing constraints [25].

B. *Modeling Categories of Devices*

Fig. 3 illustrates the *Rebootable* Device Category Automaton (RDC) adapted for *Rebootable* devices from the botnet-device interactions outlined in [2]. Connected, Login_Successful, and Vulnerable are states where the device functions normally. Each RDC starts in the Connected state

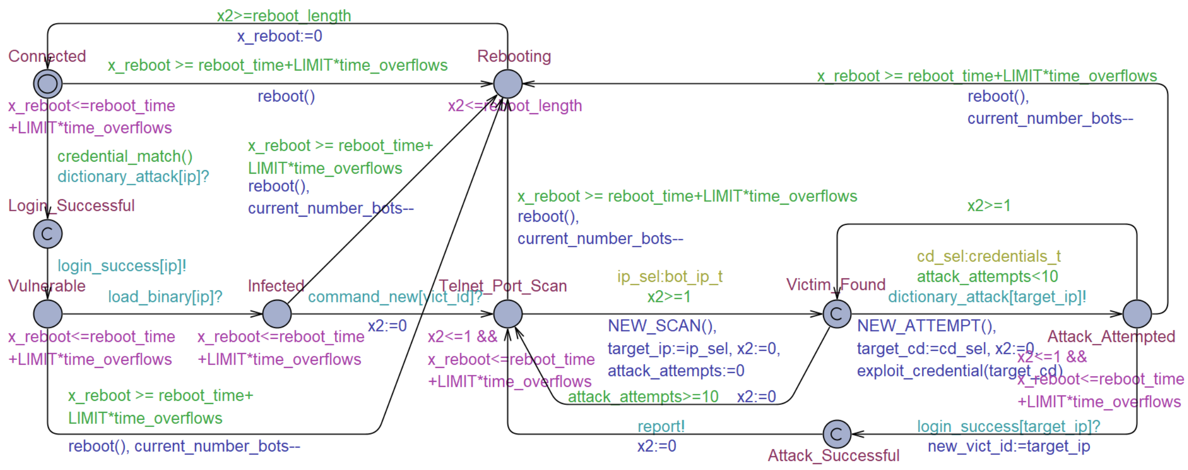


Fig. 3. *Rebootable* device category automaton (RDC) adapted from [2]

with a random IP address and a random weak credential (we only model vulnerable devices). If attacked with a matching credential, it moves to its Vulnerable state where it now has its device information (IP address and the credentials) leaked to the botnet report server. The Infected state represents the device behavior when it has been infected by the Mirai binary but has not received any commands from the C&C server. Two adjustable parameters model the rebooting behavior: (1) *reboot_time*, representing a predetermined time after which each device will turn off and move to the Rebooting state, and (2) *reboot_length*, determining how much time rebooting takes and/or the time a device stays off after turning off. Rebooting occurs irrespective of the infection status of the device.

The rest of the states represent the behavior of the device as a bot. Upon receiving commands from the C&C server, an infected RDC scans for pseudorandom IPv4 addresses (target_ip) for open Telnet 23 ports in the Telnet_Port_Scan state. Once identified, it tries to gain a working Linux shell, attempting to brute force ten out of 62 hardcoded credentials in the Victim_Found state. Ten attack attempts, each with a random credential (target_cd), are made on the target IP address in the Attack_Attempted state. Successful attempts are followed by notifying the Report server from the Attack_Successful state before going back to the scan-attack-report cycle again. *Non-rebootable* device categories (NRDC) are modeled by removing the reboot-related parameters and states from RDC (not shown).

VI. Simulating the Botnet

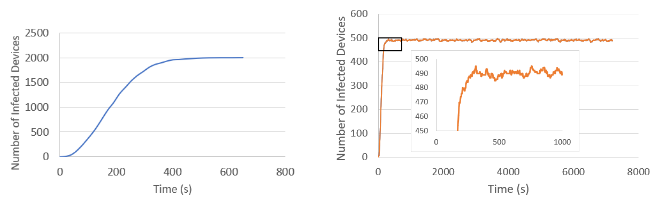
In this section, we simulate multiple instances of RDC and NRDC to observe the behavior of numerous vulnerable IoT devices in a network. By parallel execution of the Mirai-like botnet infrastructure components, we can then examine how such devices are infected and controlled to propagate the malware and whether rebooting is an effective countermeasure.

A. Simulation Parameters

There are a number of parameters that remain constant throughout all the simulations presented in this paper. Based on the description of Mirai in [6], each device has one of 62 possible credentials and bots will attempt different credentials on a target device up to 10 times before selecting a new target. When we present results for *Rebootable* and *Non-rebootable* devices, we model networks with 500 and 2000 devices respectively. It is possible to have any mix of different device categories in our model, though in the results presented we always use either only *Rebootable* or *Non-rebootable* devices.

B. Device Type Comparison

In our simulations, the amount of time simulated for *Rebootable* device networks is twice the reboot period. Each *Rebootable* device first reboots at a random time between zero and the given reboot period, then each subsequent reboot occurs exactly one reboot period after that initial time. For *Non-rebootable* devices, we stop the simulation once all devices are infected, since there will be no further change in behavior. Both simulations shown in Fig. 4 use a round-trip time of 100ms (one time unit). This time simulates network delay whenever one automaton needs to remotely communicate with another, e.g., when a bot scans ports or attempts credentials on a device.



(a) 2000 *Non-rebootable* Devices (b) 500 *Rebootable* Devices (magnification shows botnet population variation)

Fig. 4. Botnet growth for networks of different device categories

Fig. 4(a) shows how a botnet grows in a network of 2000 *Non-rebootable* devices, which follows a typical logistic growth curve [26]. As devices are infected, there are more bots to infect devices, which increases the rate of infection until the botnet begins to run out of new targets and the infection rate slows. The simulation in Fig. 4(a) ends after 648 seconds when all devices in the network are infected.

Fig. 4(b) shows the behavior of a botnet spreading in a network of 500 devices that reboot once every hour. The botnet exhibits behavior similar to *Non-rebootable* devices for a short time as devices are initially infected. Then it settles into a steady range as devices are rebooted and re-infected. The botnet rarely infects all 500 devices at once, but is consistently very close to this point. This indicates that under these model parameters, devices rebooting once an hour is not sufficient to prevent the botnet from spreading. The simulation ends after 7200 seconds, or two hours, of simulated time.

C. Impact of Rebooting

In this section, we focus on *Rebootable* devices. For each simulation, we report the average percentage of devices infected across the entire simulated time as an indicator of the botnet's ability to propagate. For example, in Fig. 4(b), the average percentage of devices infected is 95%.

It is common for malware to exhibit *stealthing* behaviors, in which they attempt to hide their presence [27]. While some use advanced stealthing techniques such as employing rootkits (e.g. Stuxnet [28]), with our focus on timing, we explore a simpler stealthing process by limiting the time a bot spends attempting to infect others. Compared to bots that spend all their time infecting other devices, bots that spend less time infecting may be harder to detect as there is less network traffic. Our simulations focus on observing the behavior of bots that spend only a certain percentage of time infecting other devices (Hibernating botnets) to compare against those that spend all their time doing so (Non-hibernating botnets).

1) Impact on Non-hibernating Botnets: Each simulation in Table II uses 500 *Rebootable* devices in a network with a fixed delay of 100ms. We simulate devices that reboot once every 5, 10, 30, or 60 minutes (each reboot is one minute in duration) to determine what frequency of rebooting can limit the botnet infection process in a network with these characteristics.

TABLE II
EFFECT OF REBOOT PERIOD IN FIXED AND VARIABLE SPEED NETWORKS

Reboot Period (min)	Average Percentage of Devices Infected (Fixed-Speed)	Average Percentage of Devices Infected (Variable-Speed)
60	95%	94%
30	91%	89%
10	78%	70%
5	61%	45%

As expected, rebooting more often results in fewer infected devices on average, since devices are cleared of infection more frequently. Despite this trend, results show that rebooting alone does not adequately control botnet size. Even when devices reboot every five minutes, meaning that after every five minutes of operation they are offline for one minute, 61% of the devices are infected on average at any given time.

We perform the simulations again for variable-speed networks to examine if a variation in the network speed impacts the botnet propagation. In these simulations, all round-trip times are uniformly randomly distributed between 0ms and 250ms. Such variations could occur in a real-world network due to a number of factors such as the distance between the botnet servers and the target device, network instability, or device mobility between cells. Simulations with variable-speed networks, however, result in a slightly lower percentage of infected devices (due to the higher expected network delay) compared to the fixed-speed network case.

TABLE III
EFFECT OF BOTNET STEALTHING ON BOTNET SIZE

Reboot Period (min)	Bot Activity (percentage)	Average Percentage of Devices Infected
60	100%	95%
60	50%	95%
60	10%	93%
60	1%	6.0%
5	100%	61%
5	50%	47%
5	10%	3.5%
5	1%	0.13%

2) Impact on Hibernating Botnets: We now repeat the experiments of Table II for botnets where infected devices spend only 50%, 10%, or 1% of the time attempting to infect other devices to see how different levels of botnet stealthing are impacted by different frequencies of rebooting.

The results from Table III indicate that a botnet's level of stealthing does in fact reduce its effectiveness, especially when frequent rebooting is considered. However, the botnet can have very high levels of stealth before seeing much reduction in size in some situations. We can see that when devices reboot once every hour, bots that are active all the time are only slightly more effective than bots which are active only 10% of the time. When the reboot period is significantly reduced, the differences between levels of activity are much greater, and botnets that are only active 1% of the time cannot amass many bots in any of our scenarios.

VII. DISCUSSION

The results in Tables II and III point towards rebooting as an ineffective countermeasure on its own in the environment that we simulated. The selected reboot periods are meant to highlight this issue. Hourly rebooting, assuming no faults occur, provides an effective up-time of 98.36% as for every 60 minutes of operation, a device is offline for one minute. This may be already too low for some IoT applications, yet over 90% of devices remain infected on average when a botnet exhibits no stealthing. Even rebooting devices every five minutes, which we argue is infeasible in most circumstances, allows for a large percentage of infected devices. If such frequent rebooting were to be used in typical targets of Mirai, such as routers and security cameras [6], significant loss of network connectivity or security monitoring capabilities would result, even which would not adequately limit botnet spread.

When we examine how botnet stealthing impacts botnet size, we find that botnets can achieve very high levels of stealthing before their ability to spread is significantly affected. In the highest levels of stealthing, the more frequent levels of rebooting can thwart most of the botnet propagation. However, it is important to note that the level of stealthing a botnet exhibits is a property of the botnet itself and is not something that can be controlled by network operators.

While the model presented in this work is specific to the Mirai botnet, with the appropriate information, it can be adapted to other Mirai variants or imitators. As of 2019, 63 Mirai variants had been identified by IBM X-Force [29]. Simulating any Mirai variant may be as simple as configuring the parameters presented in Section VI-A or adding relevant states in the device models and/or the C&C infrastructure. Other IoT device categories, such as those that can be patched, can be explored by reusing the C&C infrastructure and implementing an automaton for the new device category.

It is worth noting that our model idealizes network conditions. Factors such as network congestion or failure can negatively impact bot scanning and infection activities. In terms of scalability, while the number of devices in our model is significantly higher compared to other formal methods approaches [7], [15], we are still far from the hundreds of thousands of bots observed in reality as part of Mirai [6]. However, while Mirai varied between 100,000 to 600,000 bots from September 27, 2016 to February 28, 2017, 484 unique C&C servers were identified [6]. If we assume bots to be randomly distributed among those servers, then each would manage approximately 200 to 1,200 bots. As we model just one C&C server, the capability of our model to mimic the behavior of 2000 devices for a single C&C server is a good step towards modeling entire botnets with multiple C&C servers and thousands of nodes in the future.

VIII. Conclusions

In this paper, we developed timed automata models of a Mirai-like botnet infrastructure and categories of different IoT devices in UPPAAL. We simulated large networks of such device categories to examine the efficacy of rebooting as a countermeasure on different device configurations, finding that device rebooting, by itself, is ineffective at reducing the spread of botnets. Different levels of botnet dormancy are identified to be notable factors that affect the accumulation and maintainability of a large botnet over an extended period. We note that our observations are impacted by the environment in which we simulated, especially by network parameters.

In future work, we aim to explore other suggested countermeasures (e.g., patching) and device categories to analyze networks with different distributions of device categories.

References

[1] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "Ddos in the iot: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[2] G. Kambourakis, C. Kolias, and A. Stavrou, "The mirai botnet and the iot zombie armies," in *MILCOM 2017-2017 IEEE Military Communications Conference (MILCOM)*, pp. 267–272, IEEE, 2017.

[3] Cloudflare, "Famous ddos attacks — the largest ddos attacks of all time."

[4] S. S. Silva, R. M. Silva, R. C. Pinto, and R. M. Salles, "Botnets: A survey," *Computer Networks*, vol. 57, no. 2, pp. 378–403, 2013.

[5] M. Mahmoud, M. Nir, A. Matrawy, *et al.*, "A survey on botnet architectures, detection and defences.," *Int. J. Netw. Secur.*, vol. 17, no. 3, pp. 264–281, 2015.

[6] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the mirai botnet," in *26th USENIX Security Symposium (USENIX Security 17)*, (Vancouver, BC), pp. 1093–1110, USENIX Association, Aug. 2017.

[7] H. Tanaka, S. Yamaguchi, and M. Mikami, "Quantitative evaluation of hajime with secondary infectivity in response to mirai's infection situation," in *2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*, pp. 961–964, IEEE, 2019.

[8] R. Alur and D. L. Dill, "A theory of timed automata," *Theoretical computer science*, vol. 126, no. 2, pp. 183–235, 1994.

[9] G. Behrmann, A. David, and K. G. Larsen, "A tutorial on uppaal," in *Formal methods for the design of real-time systems*, pp. 200–236, Springer, 2004.

[10] D. Bekerman, "New mirai variant launches 54 hour ddos attack against us college," Oct 2019.

[11] M. M. Mimoso, "Windows botnet spreading mirai variant," Feb 2017.

[12] B. Krebs, "New mirai worm knocks 900k germans offline," Nov 2016.

[13] D. McMillen and M. Alvarez, "Mirai iot botnet: Mining for bitcoins?," Apr 2017.

[14] M. Shanmughapriya, G. Sumathi, and K. Aarthi, "Bot net of things–a survey," *International Journal of Engineering and Computer Science*, vol. 7, no. 05, pp. 23926–23930, 2018.

[15] H. Tanaka and S. Yamaguchi, "On modeling and simulation of the behavior of iot malwares mirai and hajime," in *2017 IEEE International Symposium on Consumer Electronics (ISCE)*, pp. 56–60, IEEE, 2017.

[16] J. O. Kephart and S. R. White, "Measuring and modeling computer virus prevalence," in *Proceedings 1993 IEEE Computer Society Symposium on Research in Security and Privacy*, pp. 2–15, IEEE, 1993.

[17] J. A. Jerkins and J. Stupiansky, "Mitigating iot insecurity with inoculation epidemics," in *Proceedings of the ACMSE 2018 Conference*, pp. 1–6, 2018.

[18] P. Wainwright and H. Kettani, "An analysis of botnet models," in *Proceedings of the 2019 3rd International Conference on Compute and Data Analysis*, pp. 116–121, 2019.

[19] M. Feily, A. Shahrestani, and S. Ramadass, "A survey of botnet and botnet detection," in *2009 Third International Conference on Emerging Security Information, Systems and Technologies*, pp. 268–273, IEEE, 2009.

[20] S. Herwig, K. Harvey, G. Hughey, R. Roberts, and D. Levin, "Measurement and analysis of hajime, a peer-to-peer iot botnet.," in *NDSS*, 2019.

[21] D. Acarali, M. Rajarajan, N. Komninos, and I. Herwono, "Survey of approaches and features for the identification of http-based botnet traffic," *Journal of Network and Computer Applications*, vol. 76, pp. 1–15, 2016.

[22] P. R. D'Argenio, J.-P. Katoen, T. C. Ruys, and J. Tretmans, "The bounded retransmission protocol must be on time!," in *International Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 416–431, Springer, 1997.

[23] H. Lonn and P. Pettersson, "Formal verification of a tdma protocol start-up mechanism," in *Proceedings Pacific Rim International Symposium on Fault-Tolerant Systems*, pp. 235–242, IEEE, 1997.

[24] A. Jawad and J. Jaskolka, "Analyzing the impact of cyberattacks on industrial control systems using timed automata," in *2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS)*, pp. 1–12, IEEE, 2021. (To Appear).

[25] C. Bormann, M. Ersue, and A. Keranen, "Terminology for constrained-node networks," 2014.

[26] G. Serazzi and S. Zanero, "Computer virus propagation models," in *International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pp. 26–50, Springer, 2003.

[27] W. Stallings and L. Brown, *Computer security: principles and practice*. Pearson, 3 ed., 2015.

[28] T. M. Chen and S. Abu-Nimeh, "Lessons from stuxnet," *Computer*, vol. 44, no. 4, pp. 91–93, 2011.

[29] C. DeBeck, J. Chung, and D. McMillen, "I can't believe mirais: Tracking the infamous iot malware," Jul 2019.