Modelling Correct Operation of Webcams for Security Purposes

Blaine Billings
College of Charleston
Charleston, South Carolina
billingsbt@g.cofc.edu

ABSTRACT

In October of 2016, the world saw a Denial of Service (DoS) attack, the Mirai botnet, which made use of machines on a global scale, primarily targeting often-unprotected devices such as webcams and routers. Due to the widespread use of the Internet of Things (IoT), and, more specifically therein, webcams, the attack surface available to malicious actors has increased dramatically. Whereas some researchers tackle this problem by measuring and increasing the efficiency of existing Intrusion Detection Systems (IDSs) or by creating models for the purpose of characterizing cyber-attacks, such solutions do not investigate the problem of identifying when a system itself is behaving under incorrect operation. Through our research, we establish a set of deterministic models that are able to accurately and efficiently model the correct operation and behavior of webcams. In order to verify the efficacy and validity of such models, we run a multitude of normal-operation scenarios and cyber-attacks against webcams using an isolated network. Using the data from these emulated experiments, we correlate data extracted from network traffic and audit logs to verify the correctness and accuracy of our models, eventually generalizing the methodology used therewith to present its extensibility to other IoT devices.

CCS CONCEPTS

• **Networks** → Network security;

KEYWORDS

Webcam, IoT, Models, Security

ACM Reference Format:
Blaine Billings. 2018. Modelling Correct Operation of Webcams for Security Purposes. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 PROBLEM AND MOTIVATION

The Internet of Things (IoT) is becoming more mainstream in public usage, creating opportunities for improved quality of life with applications in healthcare, home security, and beyond. At the same time, though, this inadvertently allows for a drastic increase in security vulnerabilities. The increase in the attack surface offers new vectors with which devices can be weaponized for performing sophisticated spam attacks, for acting as pivot machines to reach high value targets, and for being used as bots in scalable Denial of Service (DoS) attacks, such as in the infamous Mirai botnet [1]. The question then arises of how one might be able to secure such commonly targeted machines, both for privacy and for the prevention of subsequent and related attacks.

Consequently, IoT Security has been explored in the past few years by industry and academia alike. Babar et. al. [3] propose a threat model and taxonomy, through which they discuss security requirements for the IoT: mobility, wireless, embedded use, diversity. Xu et. al. [13] describe a novel security approach for IoT with Physical Un-clonable Functions (PUFs) that will help in data privacy and integrity. Riahi et. al. [9] take a different approach to IoT security, one that is systemic and cognitive. Specifically, their approach can be represented by a triangular-based pyramid with the concepts of person, technological ecosystem, process, and intelligent object as its vertices. They analyze this pyramid with regards to safety, security, access, and cyber-security. Zhao and Ge [14] present a survey in IoT security that spans from key management to secure routing, data fusion, and authentication technologies.

Different to the prior work, we study extensively the fundamental characteristics of the IoT and create models of correct operation with the goal of identifying anomalies representative of improper behavior while under attack. In order to tackle such a problem, we aim to investigate a simple, but very common, medium for modern cyber-attacks - webcams. In a previous project [8], we established a general understanding of the IoT ecosystem, focusing on the classification of IoT devices into an organized taxonomy. We use this research as a launching point in order to focus on the creation of models for the correct operation of webcams. Through the use of deterministic modeling, we investigate such commonly targeted devices and create models of their proper operation. After creating these models, we establish their robustness by observing the webcam under correct operation, following up with various attacks, emulating a variety of real-world scenarios characteristic of both normal and abnormal behavior. The data collected is thusly used to calibrate thresholds to describe the states within each model. Finally, we generalize this methodology and present it accordingly insofar as to be easily applicable for all IoT devices (webcams and otherwise) so that other researchers or industry professionals may arrive at similar models for their own machines.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 RELATED WORK

Taking a broad look, models of correct operation are commonly used in engineering for fault detection. Venkatasubramanian et. al. [11] present a three-part survey that extensively analyzes these models. In the first part, they discuss quantitative models related to desirable characteristics of fault detection and discuss the layers of these models: measurement, feature, decision and class space. In the second part, they present qualitative models of correct operation for fault detection, such as topographic models that can be divided into anomaly, function, and structural groups of modeling. In the third and final part, they present data-driven models of correct operation that use statistical analysis, employing such techniques as PCA, Clustering, Observer digraphs, Qualitative Trend Analysis, and hybrid techniques.

The idea of creating models of correct operation is a widely pursued topic in the security field, as well. At the same time, it is one of the hardest to quantify, as it involves both the machine and human. LeMay et. al. [7] tackle a similar problem in a paper focusing on the creation of a deterministic adversary model and a corresponding framework for evaluating system security. However, whereas these models focus on system security, they are not able to discern the specifics of correct and incorrect operation. Forrest et. al. [5] use the temporal correlation of small groups of processes to build a regular profile database for the correct operation of Unix systems. On a similar note, Sekar et. al. [10] create a correlation with system calls and software behavior but do so by building a finite state automaton that captures long and short-term temporal relationships. We tackle our problem in an analogous fashion as what is presented in these two works. However, our model uses network and machine characteristics extracted solely from within the device in question in order to derive the regular operation states for a webcam. Furthermore, the approach we take in this project is unique in three main aspects:

- We model solely what is known, i.e., the network components, rather than the unknown adversary and unpredictable attacks. This approach is inspired by behavioral network IDSs and host IDSs, but had not been previously deployed to the IoT based on their prevalent features.
- We hypothesize model states based on specifications and generic IoT features derived by the standard on the "Network of Things" as set out by the National Institute of Standards and Technology (NIST) [12].
- We emulate the component behavior to derive the model states with real experimentation and instrumentation on a webcam. A side effect of our approach is instrumentation on IoT that needs to be lightweight in computation and memory. Another important derivative is the generation of unique datasets that are valuable to both the security and data science communities, which still use datasets such as the twenty year old KDD cup security dataset [2].

Often times, behavior models, as defined above, are used as a basis for anomaly-based Intrusion Detection Systems (IDSs), which rely heavily on a predetermined model against which the operation of a system under attack can be compared, thereby detecting intrusion. Cheung et. al. [4] created such a model-based IDS, which focused on the correct operation of Modbus, classifying anything that was outside of the scope of their model as anomalous. In addition, Jyothsna et. al. [6] investigated the efficiency of anomaly-based IDSs, revealing the importance and applicability of behavior models in reference to security. While the behavior models described herein could be used as a foundation for one of these anomaly-based IDSs, the creation of one such IDS is not necessary for demonstrating the efficacy of our models and is outside of the scope of this project.

3 MODELING CORRECT OPERATION OF WEBCAMS

In this section, we present our methodology used in constructing a model of correct operation for webcams, later detailing the experimentation and verification of the model's correctness. This methodology, as will be shown, may then be generalized and extended, making it applicable to additional IoT devices outside of the rather constrained field of webcams.

3.1 Methodology for Model Construction

Our primary goal was to create generic models for the IoT, as compared to one specialized model per device. To this end, we applied an abstraction of the Internet of Things depicted by Voas [12] as our starting point for grouping devices. Voas elaborated on the building blocks, or *primitives*, that are found in the Networks of Things (NoTs), those being sensors, aggregators, communication channels, eUtilities, and decision triggers. Based on these NoT primitives, we created a decision tree taxonomy described in [8]. In this taxonomy we allow the root of the tree to be any device and the first level to consist of the general features that belong to all IoT devices. This level includes characteristics related to the NoT primitives. Specifically, `Purpose` and `Mobility` associate to actuators; `Communication Channels` associates to communication channels; and `Purpose` to eUtilities, Aggregators, Decision Triggers, and Sensors. The second and further levels of the decision tree taxonomy consist of binary opposite branches and leaves. These later levels allow the set of IoT devices to be partitioned into distinct groups. Therein, we proved the completeness, correctness, and timelessness of the taxonomy before using it here as the foundation of our models of correct operation.

Looking through the lens of our previous research on an IoT taxonomy, we first investigated the inherent characteristics of a webcam; how it acts and how it communicates. We started by defining multiple Finite State Machines (FSMs) based on the abstract operations that were characterized by our taxonomy. In order to decrease overlap between the independent FSMs, we created each to be based on only

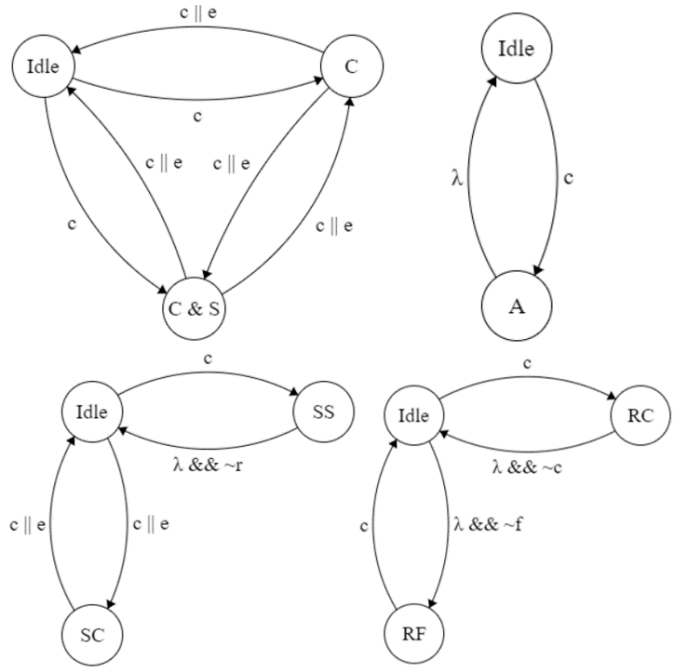

Figure 1: Model of correct operation for webcams FSMs.

one characteristic of the webcam, those being: I. Network Reception, II. Data Transmission, III. Data Collection, and IV. Command Response, as seen in Figure 1. Furthermore, we defined transitions, such as Command Processing, Event Triggers, File Transfers, and Request Fulfillments, to model the temporal interconnectivity between the different states within each model. The state definitions are given below:

- *C - Collecting:* collect information through sensors, whether from the environment or from the network,
- *S - Saving:* save information to main memory,
- *A - Action:* take action, such as rotate or produce a sound to scare an intruder,
- *SS or SC - Send Saved or Collecting:* send information that is being collected by sensors directly to the network, or send information that was previously saved, i.e., an old video,
- *RC or RF - Receive Command or File:* receive a command or receive a file through the network.

Finally, we defined features to characterize each state and each transition, including bytes in and out per second, packet inter-arrival delay, and CPU and memory utilization, along with corresponding thresholds in order to differentiate between states. We describe these experiments, which lead to a data-driven characterization of the FSM states, in the next subsection.

3.2 Experimental Model Construction

In order to calibrate the thresholds that differentiate the individual states, we decided to collect data on a webcam set up within an isolated network. This allowed us to complete experiments of both regular and attack operation. Our network consisted of a single, vulnerable webcam and an insider that was either operating this camera as a regular operator or as an attacker. The webcam model that we use is a popular, plug-and-play, affordable IP camera - the Avacom H5060W - chosen based on a specific telnet vulnerability that makes the camera part of a remotely operated botnet. Furthermore, this camera offers root privileges through telnet, which makes it susceptible to password changes and malware installation. As operation was limited by the memory and the firmware of the device, we used the default installed software, such as ping, top, and netstat, for data collection and experimentation.

Our experimentation was twofold. Firstly, we tested states of correct operation by using regular commands that reflected proper usage of the webcam, such as recording, transmitting, saving, and receiving and responding to a command from a network-connected device. We emulated the following regular operation scenarios: 1. Home usage: weekday operation differs from weekend and vacation operation. In the former, we assumed the webcam operates during business hours and streams to owners, weekend may include scarce recording, and vacation may include additional streaming. 2. Enterprise operation: we assumed frequent usage from security and owners with differing intervals of streaming frequency to simulate varying levels of employee diligence. 3. Public operation: continuous operation with moderately random and infrequent streaming was assumed. Secondly, we set up experiments by running multiple different types of attacks against the webcam, including DoS, remote to local telnet access, and probing. Our experiments consisted of: 1. The webcam being scanned using nmap, since this is the usual start of an active attack, 2. The webcam being part of a botnet, running a fast flooding DoS attack with multiple ping instances executed simultaneously, 3. The webcam being a part of a botnet, sending very large, but slow, ICMP packets, and 4. The webcam being on the receiving side of a DoS attack made with hping3.

Processing this data offline, post-experimentation, we correlate what was collected with the various known states of regular operation flagged during experimentation, identify thresholds that differentiate states, and analyze deviations to calibrate our models.

4 RESULTS

Through this work, we were able to arrive at a set of four deterministic, data-driven models which, altogether, are able to model the behavior of webcams. Using the data collected from our series of experiments, we formulated a data vector that would characterize the states of the webcam. This vector consisted of fields such as number of network connections, CPU usage, and bytes in per second, amongst others that were all extracted through the use of ps, top, netstat, and

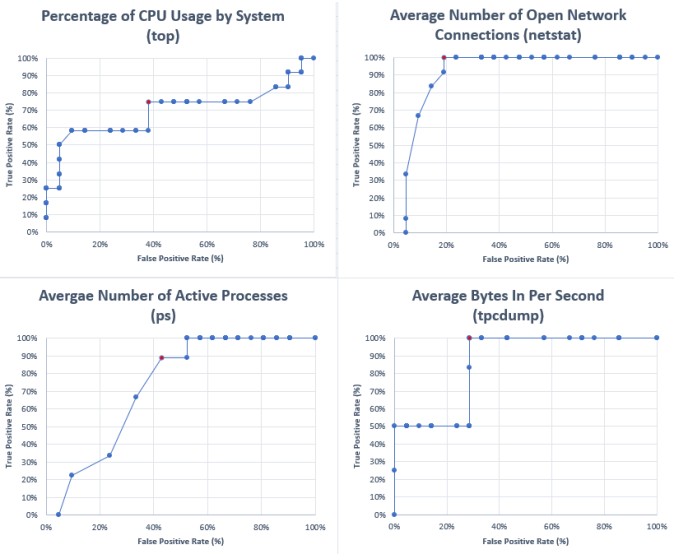

Figure 2: Receiver Operating Characteristic Curves for various fields of the webcam correct operation model data vector.

tcpdump. Receiver Operating Characteristic (ROC) curves were used to show the detection rate of anomaly behavior that deviated from our regular operation models. These curves plot false positive rates against true positive rates for different thresholds of the data fields within the vector, where true positives are characterizations of normal states and false positives are incorrect classifications of irregular states as regular.

We plotted the ROC curves for a variety of metrics, such as bytes in/out per second, packet inter-arrival delay, CPU usage, memory usage, and number of protocols used. These plots helped us calibrate the thresholds used for differentiating correct operation states from the abnormal. These results can be seen in Figure 2. In the interest of space, we have included ROC curves for only four of these thresholds. The data points in the graphs represent thresholds that were within the following intervals: Percentage of CPU Usage by the System [10%, 40%] in steps of 0.1%, Average Number of Open Network Connections during Capture [20, 50] in steps of 0.1, Average Number of Active Processes During Capture [42, 48] in steps of 0.02, and Average Bytes in Per Second [400, 2500] in steps of 25. The threshold that maximizes true positive rate while at the same time minimizing false positives is selected from each plot. These thresholds are marked in red in the graphs. One can see that, although the operation scenarios were limited, we were able to reach acceptable levels of accuracy.

Beyond threshold calibration, we used the data in order to check the models that we had created, analyzing their completeness. Were we to see patterns in the data that suggested the presence of another state for which our models did not account, we would be able to further refine them to better fit the device and its actual operation. This allowed us to see whether or not there were any missing states as well as incorrect state transitions and unused states (states that our model accounted for but were never actually used by the device).

Detailing our process, we provide a methodology by which other researchers or industry professionals may arrive at 1) similar models for a given device and 2) a way for calibrating thresholds to describe the correct operation models and a data vector that tie the two together. By looking at the device itself, as compared to outside the machine or the network, this provides quicker response time, as there is nothing else with which the device needs to communicate. The behavior can be quickly and easily evaluated through such lightweight models, as they are constituted by only a minimal number of states and transitions, whether they be used by an IDS located elsewhere in the network or on the device itself (given it is an IoT device with sufficient computation power).

5 CONCLUSIONS

We have described a methodology to construct models of correct operation for IoT devices based on their generic features and for creating data-driven states based on emulation and device instrumentation. These models can indicate anomalies to evaluate IoT security and alert when a device is being used as part of a botnet or as a pivot machine to access assets in a network. The models defined here are completely attack-independent; i.e., one need not know about certain attacks ahead of time in order to defend from them, as the models only measure correct operation. Furthermore, all data is collected by the device itself, so outside knowledge is also rendered unnecessary for attack detection. Therefore, anything signaled as being out of the norm would implicate malicious action. In addition, these models prove useful in two ways: 1) as standalone modules for deriving objective metrics in the security evaluation of a network, and 2) as part of a behavioral IDS whose goal is to detect intruders based on anomalies that deviate from regular operation.

In our future work, we plan to verify these models both through formal techniques, such as bounded model checking algorithms, and with data caputred from a real-world setting, with which we would be able to test the validity of our models. Furthermore, our goal is to use these models in order to derive security metrics for IoT components. Because the behavior of IoT devices deviates from the model when under attack, we plan to use the objective metrics of machine behavior to evaluate how secure a device is under different security stack configurations, such as isolating the device in question in a subnet, putting it behind a firewall, etc. Relative entropy and other distance techniques can be used to derive these differential metrics. We also plan to further pursue this work in order to gain insight into the applications of models, such as finite state machines, and metrics, similar to the data vector and its calibrated thresholds use to describe machine state, in reference to their applications to the field of cyber security. In addition, we hope to show the applicability and practicality of this device-centered approach in order to achieve faster

detection and response time with respect to network and device intrusion.

ACKNOWLEDGMENTS

The author would like to thank that National Science Foundation for partially funding this research project under the award #1700254 as well as Dr. Xenia Mountrouidou who guided this research project.

REFERENCES

[1] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. 2017. Understanding the Mirai Botnet. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 1093–1110.

[2] The UCI KDD Archive. 1999. KDD Cup 1999 Data. (1999). http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[3] Sachin Babar, Parikshit Mahalle, Antonietta Stango, Neeli Prasad, and Ramjee Prasad. 2010. Proposed Security Model and Threat Taxonomy for the Internet of Things (IoT). In *Recent Trends in Network Security and Applications*. Springer Berlin Heidelberg, Berlin, Heidelberg, 420–429.

[4] Steven Cheung, Bruno Dutertre, Martin Fong, Ulf Lindqvist, Keith Skinner, and Alfonso Valdes. 2007. Using Model-based Intrusion Detection for SCADA Networks. In *Proceedings of the SCADA Security Scientific Symposium*. Miami Beach, Florida.

[5] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. 1996. A Sense of Self for Unix Processes. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, Los Alamitos, CA, 120–128.

[6] V Jyothsna, V V. Rama Prasad, and K Munivara Prasad. 2011. A Review of Anomaly based Intrusion Detection Systems. 28 (Aug 2011), 26–35.

[7] Elizabeth LeMay, Michael D. Ford, Ken Keefe, William H. Sanders, and Carol Muehrcke. 2011. Model-based security metrics using ADversary VIew Security Evaluation (ADVISE). In *Proceedings of the 2011 8th International Conference on Quantitative Evaluation of Systems, QEST 2011*. 191–200.

[8] Xenia Mountrouidou, Blaine Billings, and Luis Mejia. [n. d.]. *Elsevier Journal of Network and Computer Applications (submitted April 2018)* ([n. d.]).

[9] A. Riahi, Y. Challal, E. Natalizio, Z. Chtourou, and A. Bouabdallah. 2013. A Systemic Approach for IoT Security. In *2013 IEEE International Conference on Distributed Computing in Sensor Systems*. 351–355.

[10] R Sekar, M Bendre, P Bollineni, and Dinakar Dhurjati. 2001. A fast automaton-based approach for detecting anomalous program behaviors. (Jan 2001), 144–150.

[11] Venkat Venkatasubramanian, Raghunathan Rengaswamy, Kewen Yin, and Surya N. Kavuri. 2003. A Review of Process Fault Detection and Diagnosis. 27 (2003), 293–346.

[12] Jeffrey Voas. 2016. Network of 'Things'. (Jul 2016).

[13] Teng Xu, James B. Wendt, and Miodrag Potkonjak. 2014. Security of IoT Systems: Design Challenges and Opportunities. In *Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD '14)*. IEEE Press, Piscataway, NJ, USA, 417–423.

[14] K. Zhao and L. Ge. 2013. A Survey on the Internet of Things Security. In *2013 Ninth International Conference on Computational Intelligence and Security*. 663–667.