Sentinel: A Robust Intrusion Detection System for IoT Networks Using Kernel-Level System Information

Adrien Cosson, Amit Kumar Sikder, Leonardo Babun, Z. Berkay Celik, Patrick McDaniel, A. Selcuk Uluagac
Authors

- **Adrien Cosson** - Penn State University, Graduated with Master’s
- **Amit Kumar Sikder** - Florida International University, Postdoctoral Fellow
- **Leonardo Babun** - Florida International University, Received Doctoral Degree
- **Z. Berkay Celik** - Purdue University, Assistant Professor
- **Patrick McDaniel** - Penn State University, Professor, Director of the Institute for Networking and Security Research (INSR)
- **A. Selcuk Uluagac** - Florida International University, Assistant Professor

IoTDI 2021 - ACM/IEEE Conference on Internet of Things Design and Implementation, 2021
IoT & Cyber Attacks

- IoT devices becoming more common
- Influenced by economics and speed to market
- Devices are resource-constrained
- Developers don’t have direct access to the hardware to integrate security measures
- Attacks
  - Node-level
  - Network-level
  - Application-level
- Mirai Botnet: launched a series of DDoS attacks
Intrusion Detection

- Intrusion detection detects a system for malicious behavior
  - Architectures
    - Network-based IDS (NIDS): monitor the state of an entire network
    - Host-based IDS (HIDS): run on a specific host and search for malware operating inside of it through the use of system-level and process-level information
  - Approaches
    - Signature-based: compares the collected data pattern to a list signatures of known threats
    - Anomaly-based: builds an internal representation of the system compared to an expected baseline state
    - Specification-based: has set of baseline and threshold values and compares to the current situation
Sentinel Overview

- The idea of using low-level host data for intrusion detection is not new, but it hasn't been implemented for IoT environments.
- Sentinel uses a Linux-based kernel module (SKM) to collect low-level host data which is used to detect node and network level attacks.
- The heavy work of analyzing the data using ML is offloaded to the hub to differentiate between benign and malicious attacks.
Sentinel Architecture

- Uses Linux, which has high market share for IoT devices (43%)
- SKM is lightweight and easily implemented on other OS platforms
- File-based view of kernel data structures provides an easier interface for developers
- SKM is low overhead and needs less computing power
- Uses commonly found pub-sub protocol (MQTT) to make information accessible to the hub
  - Naming convention example: home/mqtt_lock/available
Sentinel Features

- Configurable polling rates: low-high, dynamic polling rate
- PostgreSQL database collects data and allows for concurrent access
- ML-based detection techniques used: Naïve Bayes, Rule-Based, Regression Model, Neural Network, Tree-Based Classifiers
- IDS collects data from the database, trains the ML model, learns benign device behavior, pushes a notification to the user interface via the hub in case of a malicious attack
Sentinel Framework

- **Polling Application**: System-level and process-level metrics
- **Kernel Module**: IoT Device (Node)
- **MQTT Broker**: Data Collection Application
- **Remote Detection Module**: Local Detection Module
- **PostgreSQL Database**: IoT Hub
Using Mirai Effects to Test Sentinel

- **Network scan/pivoting**
  - Attack 1: the attacked device continuously scans a server to find other devices

- **Exfiltration**
  - Attack 2: send large UDP packets to a server that discards them

- **C&C Keep-alive**
  - Attack 3: periodically ping an infected device that responds with an empty payload

- **Black/Grey Hole Attack**: disrupt the network by compromising a device
  - Attack 4: server floods network with large message
  - Attack 5: send out random messages to simulate the partial packet drops
Evaluation Setup & Methodology

- 2 IoT Platforms: Home Assistant and WebThings
- Binary Classification
  - The datasets contain samples recorded every second over a time window and are labeled if there is an attack or not
  - 7 performance metrics: True Positive Rate (TPR), False Negative Rate (FNR), True Negative Rate (TNR), False Positive Rate (FPR), Accuracy, F-score, and Average Computation Time (Avg. CT)
- Multi-Class Classification
  - 5 Attacks + No Attack
  - For each device/attack/framework combination, run each device for 20 min. of traces for attack scenarios and record metrics

Figure 3: Floor plan of the experimental testbed
Impacts

- Model Parameters
- Platform Configurations
- Power Consumption
- Polling Rate
Results - Binary and Multi-Class Classification

DT & RF have highest accuracies

<table>
<thead>
<tr>
<th>ML Algorithm</th>
<th>TPR</th>
<th>FNR</th>
<th>TNR</th>
<th>FPR</th>
<th>Acc.</th>
<th>F-Score</th>
<th>Avg. CT (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>0.8</td>
<td>0.2</td>
<td>0.94</td>
<td>0.06</td>
<td>0.87</td>
<td>0.864</td>
<td>21.6</td>
</tr>
<tr>
<td>PART</td>
<td>0.85</td>
<td>0.15</td>
<td>0.94</td>
<td>0.06</td>
<td>0.895</td>
<td>0.892</td>
<td>24.5</td>
</tr>
<tr>
<td>LR</td>
<td>0.91</td>
<td>0.09</td>
<td>0.9</td>
<td>0.1</td>
<td>0.905</td>
<td>0.905</td>
<td>34</td>
</tr>
<tr>
<td>MP</td>
<td>0.89</td>
<td>0.11</td>
<td>0.95</td>
<td>0.05</td>
<td>0.92</td>
<td>0.919</td>
<td>68.5</td>
</tr>
<tr>
<td>DT</td>
<td>0.95</td>
<td>0.05</td>
<td>0.97</td>
<td>0.03</td>
<td>0.96</td>
<td>0.959</td>
<td>35.6</td>
</tr>
<tr>
<td>RF</td>
<td>0.95</td>
<td>0.05</td>
<td>0.98</td>
<td>0.02</td>
<td>0.965</td>
<td>0.964</td>
<td>87.9</td>
</tr>
<tr>
<td>LMT</td>
<td>0.94</td>
<td>0.06</td>
<td>0.92</td>
<td>0.08</td>
<td>0.93</td>
<td>0.92</td>
<td>102.5</td>
</tr>
</tbody>
</table>

RF has high CT

97% average accuracy of detecting attack

96% average accuracy of detecting attack

Table 3: Performance of Sentinel in binary classification.

Table 4: Confusion matrix for WebThings multi-class classification.

Table 5: Confusion matrix for Home Assistant multi-class classification.
Results - Model Parameters

- DT: accuracy increases with the number of tree depths
- RF: accuracy increases with number of trees, but computation time increases significantly with number of trees
- Accuracy is insignificant compared to the computation time

Figure 5: Impact of model parameter in Sentinel: (a) tree depth vs accuracy using decision tree, (b) number of tree vs accuracy using random forest, and (c) number of tree vs computation time using random forest.
Results - Platform Configurations

- Accuracy drops as sampling rate increases
- Sentinel can effectively run on a low core-count IoT device

Figure 6: Detection Accuracy for (a) different polling rate (1s and 10s), (b) different computation power (1 and 4 cores).
Results - Power Consumption

- As polling frequency decreases, the power consumption overhead incurred decreases
- Inactive devices have large overhead because of sleep mode
- Can correlate the running processes to reduce overhead by reducing the polling rate

![Power Overhead Graph](image)

**Figure 7:** Power overhead caused by Sentinel for various polling periods, expressed as absolute and relative values
Results - Polling Rate

- Accuracy and power consumption are proportional for different polling rates
- Small tradeoff between accuracy and power consumption

Figure 8: Fixed polling vs dynamic polling in Sentinel
Positive Points

- Low-Cost
- Lightweight Framework
- Scalable for different configurations

Negative Points

- Device Malfunctions
- Attackers could falsify SKM data
- Any user on device can access the exposed data
Discussion

- How secure is the system?
- What are important features for the customer that Sentinel should have in terms of security?
- Is ~95% accuracy good enough?
- Are there any other metrics that could be considered, in addition to low-level system information?