PANDAcap

A Framework for Streamlining Collection of Full-System Traces

Manolis Stamatogiannakis, Herbert Bos, and Paul Groth[†]

VUSec

UvA

INDE lab[†]

INtelligent Data Engineering

In this Talk

- Motivation for this work
- Overview of PANDAcap
- Case study: SSH honeypot and dataset
- Conclusion

Motivation

Full-system trace recording

- Log **all instructions** executed and **all data** used.
- Access to full system state – deep analysis.
- Decouples analysis from timing constraints.
- Analysis flexibility.

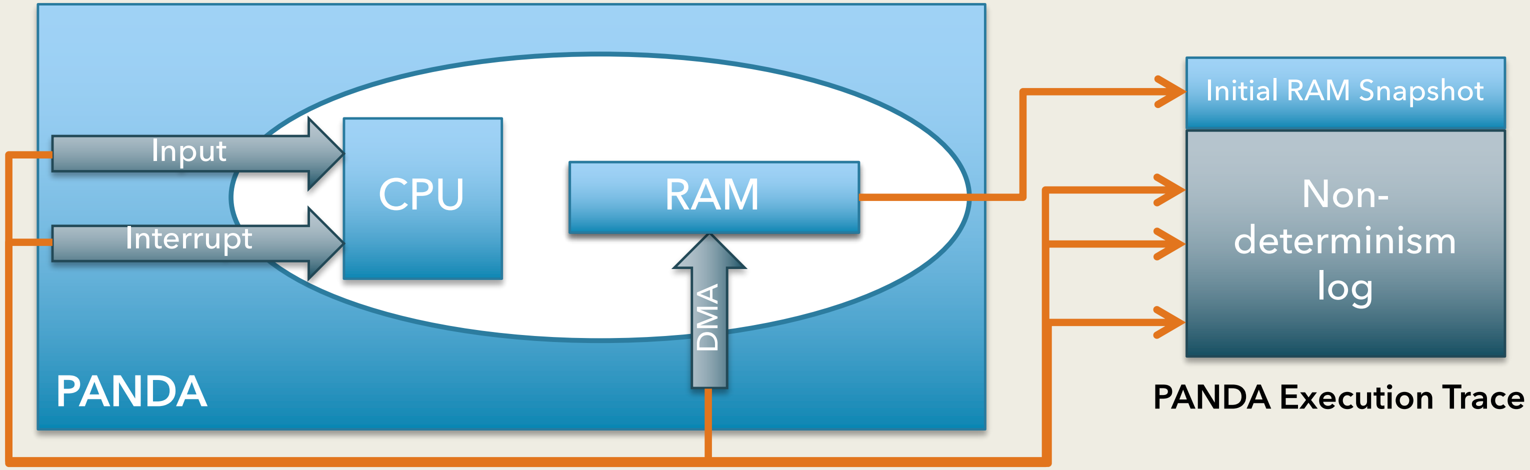
- Time consuming to setup.
- Very few full-system recording datasets available.

We aspire to lower the barrier for creating full-system recording datasets.

PANDA

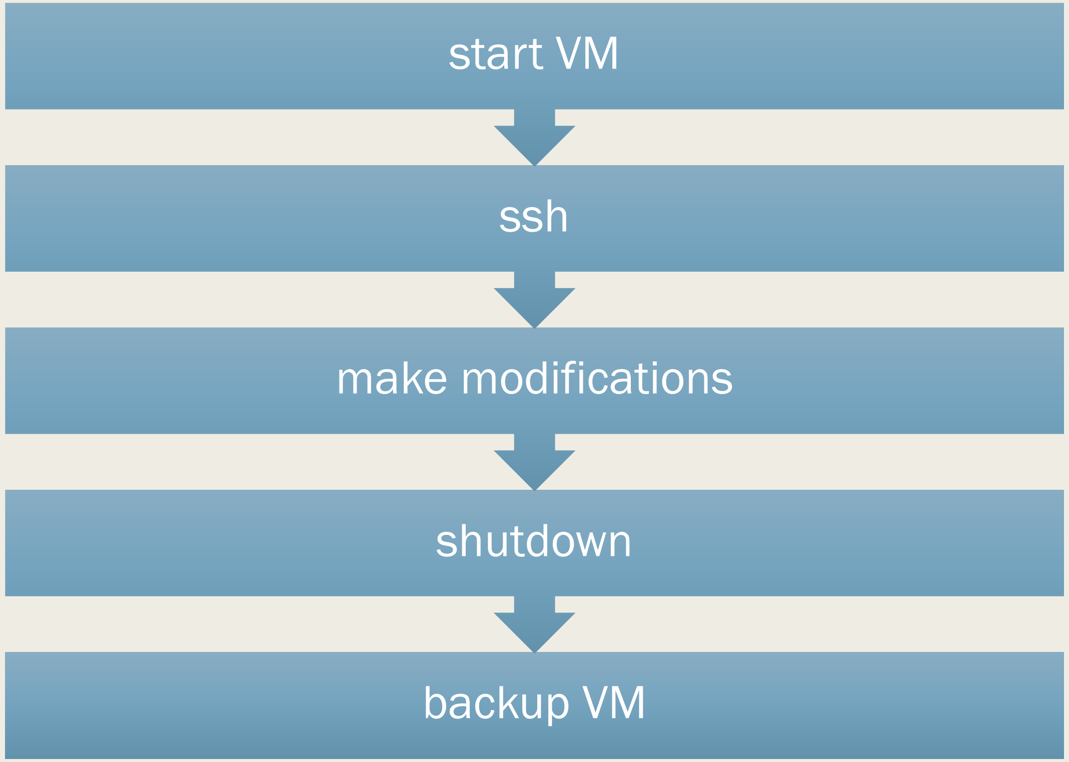
- Full System Record + Replay
- Based on QEMU
- Self-contained execution traces
- Analyses implemented as plugins

https://github.com/panda-re



(My) typical PANDA workflow

Prepare for recording



start VM

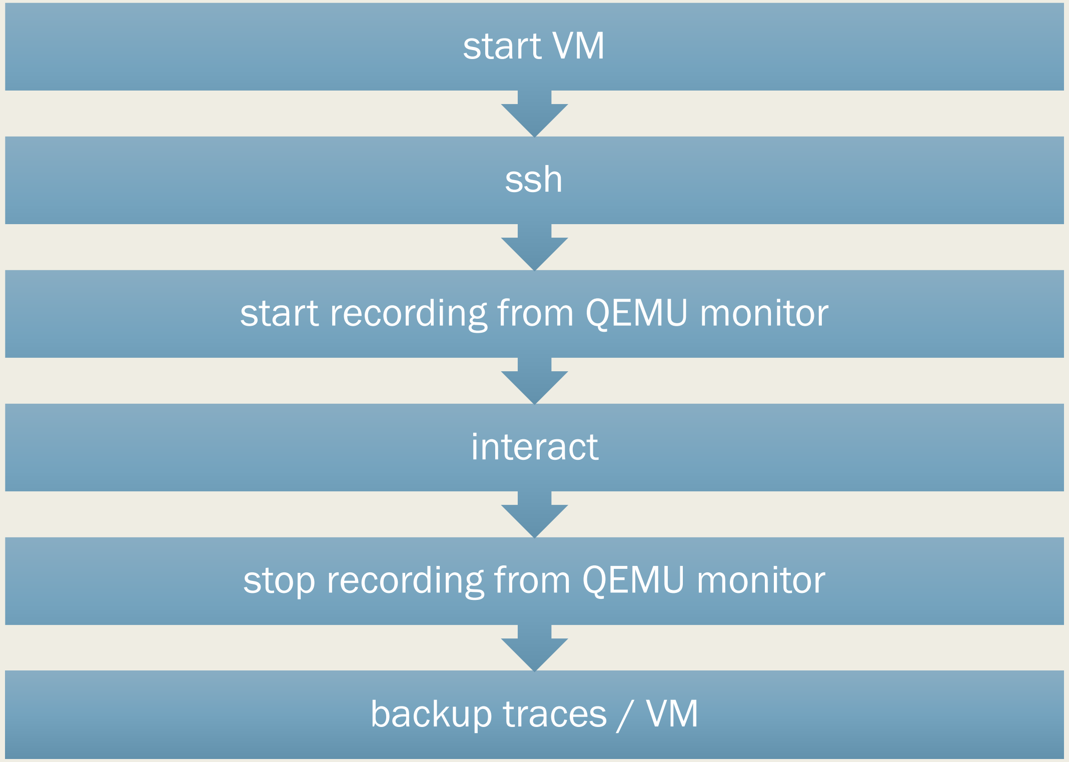
ssh

make modifications

shutdown

backup VM

Recording



start VM

ssh

start recording from QEMU monitor

interact

stop recording from QEMU monitor

backup traces / VM

Let’s create a PANDA dataset

- The regular PANDA workflow won’t cut it.
  - a lot of manual steps
  - error prone (due to the human factor)
- We need to automate things!

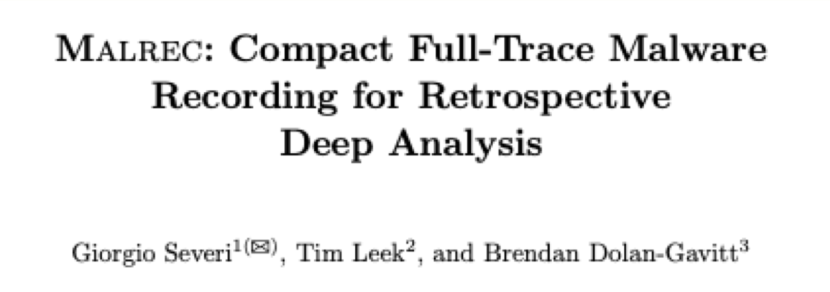
Workflow Automation Bottlenecks

- How can I start recording non-interactively?
  - Learn to work with QEMU Monitor Protocol.
- How can I start/stop recording at the right moment?
  - No elegant solution. Bummer!
- How do I move data in/out of the PANDA VM?
  - Deploy ssh keys + sftp?
- How do I replicate the same experiment with different inputs x100?
  - DIY scripting.
- How can I fully utilize my 12 core CPU?
  - …and more DIY scripting.

Now let’s put everything together

- Complicated!
- What was it again that I was doing?
- What do you mean I have to start over because I missed X?

MalRec (DIMVA 2018)



MALREC: Compact Full-Trace Malware Recording for Retrospective Deep Analysis

Giorgio Severi[1](✉), Tim Leek[2], and Brendan Dolan-Gavitt[3]

- Similar goal with us: create PANDA trace datasets
- Similar approach: off-the-shelf tools
- Purpose-built – not designed to be reusable.

  "This is not intended to work for anyone else out of the box, just to provide a starting point. You will undoubtedly have to make heavy local modifications."

- Last update in 2015 – tooling hasn't been modernized since.

Fast forward to 2020

- Containers are mainstream.
  - networking virtualization
  - storage virtualization
  - ease of deployment
- Some containers available for PANDA
  - geared towards testing builds
- Runtime customization of PANDA VMs still a DIY affair.

We can improve on this.

PANDAcap Overview

Enter PANDAcap

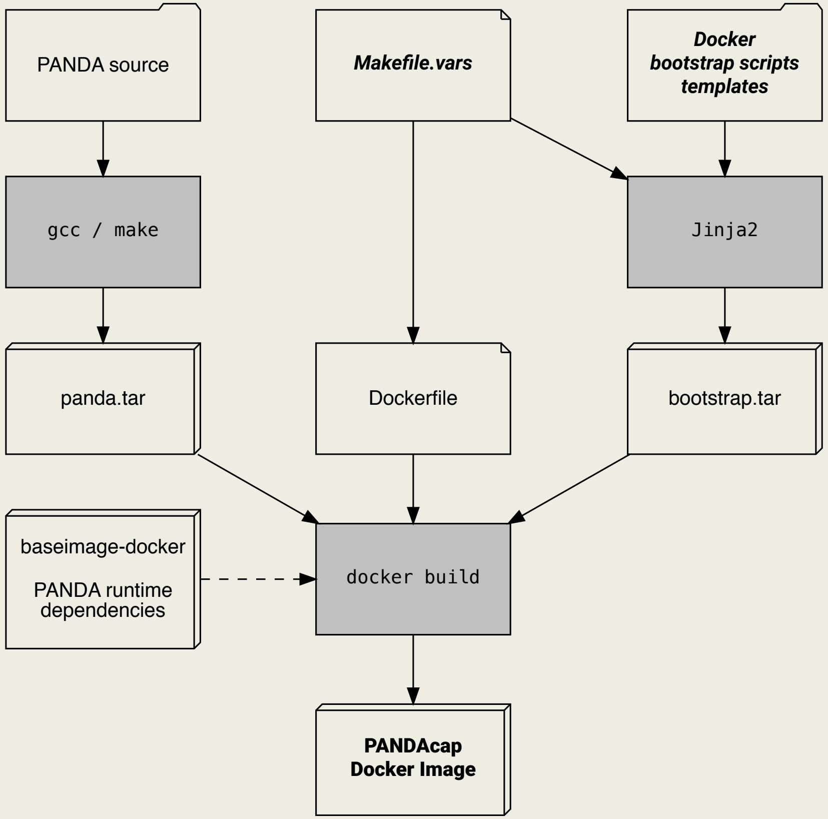
- Accurate start/stop of recording.
- Supports Docker – lean image.
- Streamlined VM bootstrapping.
  - rc.d-like initialization process
  - Jinja2 templating support
- Command line wrapper providing access to most commonly used features of Docker/PANDA.

The recctrl plugin

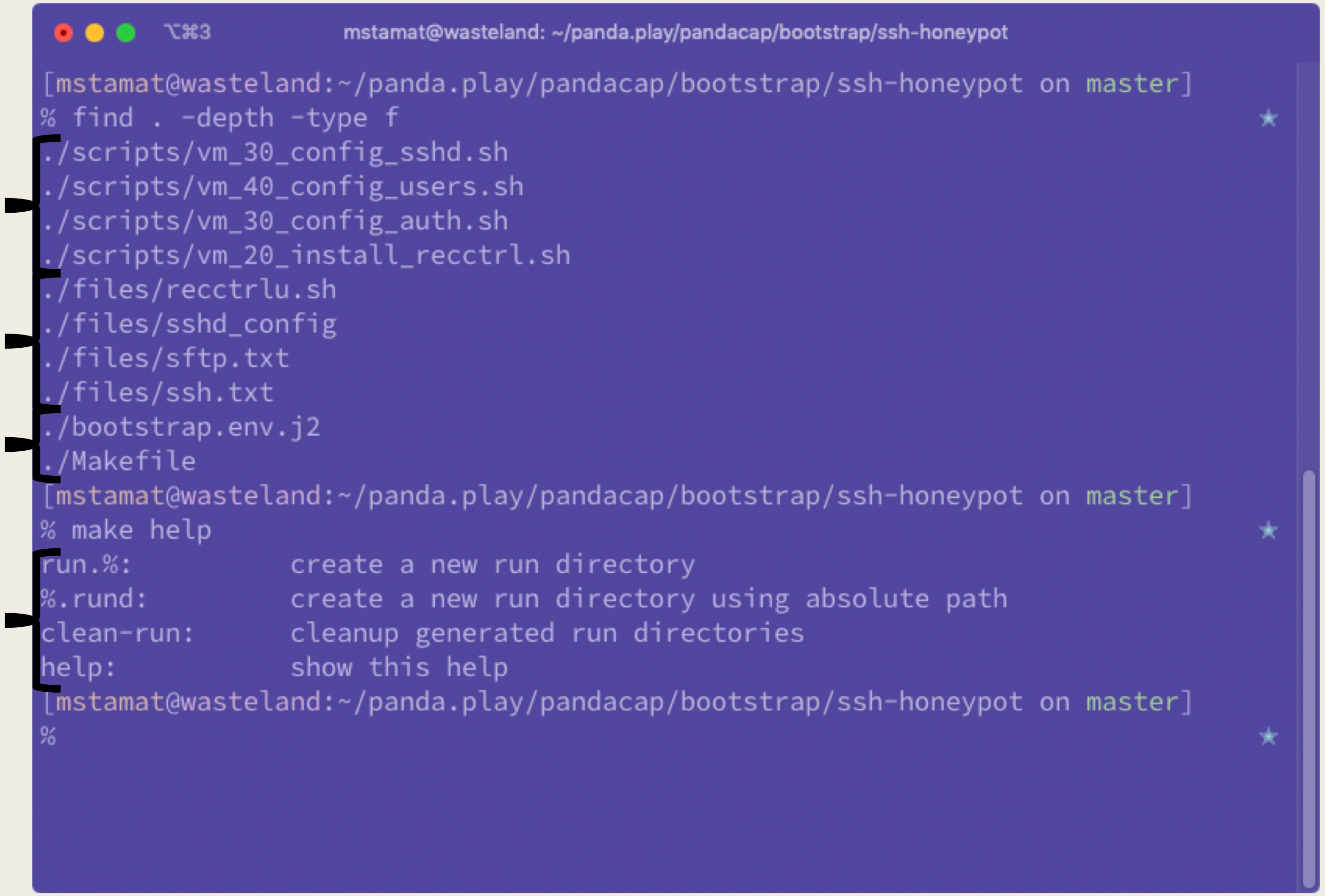
- Accurate start/stop of recording.
- Building block: PANDA_CB_GUEST_HYPERCALL.
- Support for sessions (semaphore-like).
- Support to specify the PANDA recording name from the guest.
- A timeout can be specified for limiting the length of the recording.
- Batteries included: recctrlu guest utility

Lean Docker Image

- Contains only runtime dependencies.
- Bootstrapping mechanism for Docker runtime environment.
- Shared configuration with VM runtime bootstrapping.
- Mountpoints affecting a run:
  - Docker runtime bootstrap directory
  - QCOW image for PANDA
  - Recording output directory
  - X11 server path



Runtime bootstrapping – layout



⌥⌘3
mstamat@wasteland: ~/panda.play/pandacap/bootstrap/ssh-honeypot

```
[mstamat@wasteland:~/panda.play/pandacap/bootstrap/ssh-honeypot on master]
% find . -depth -type f                                                 ★
./scripts/vm_30_config_sshd.sh
./scripts/vm_40_config_users.sh
./scripts/vm_30_config_auth.sh
./scripts/vm_20_install_recctrl.sh
./files/recctrlu.sh
./files/sshd_config
./files/sftp.txt
./files/ssh.txt
./bootstrap.env.j2
./Makefile
[mstamat@wasteland:~/panda.play/pandacap/bootstrap/ssh-honeypot on master]
% make help                                                             ★
run.%:          create a new run directory
%.rund:         create a new run directory using absolute path
clean-run:      cleanup generated run directories
help:           show this help
[mstamat@wasteland:~/panda.play/pandacap/bootstrap/ssh-honeypot on master]
%                                                                       ★
```

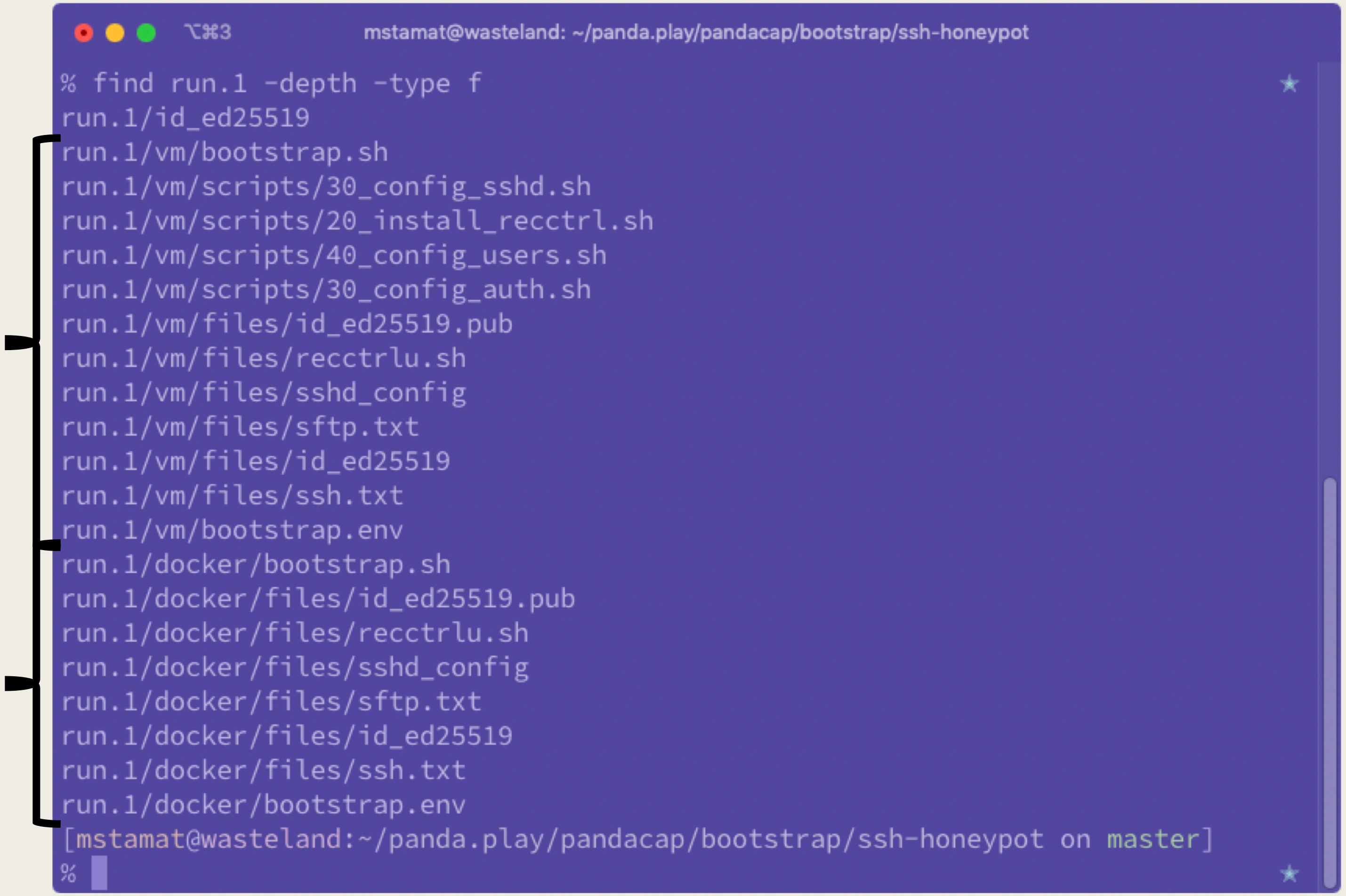
bootstrapping scripts

files used by the scripts

environment template / Makefile

Makefile targets

Runtime bootstrapping – output

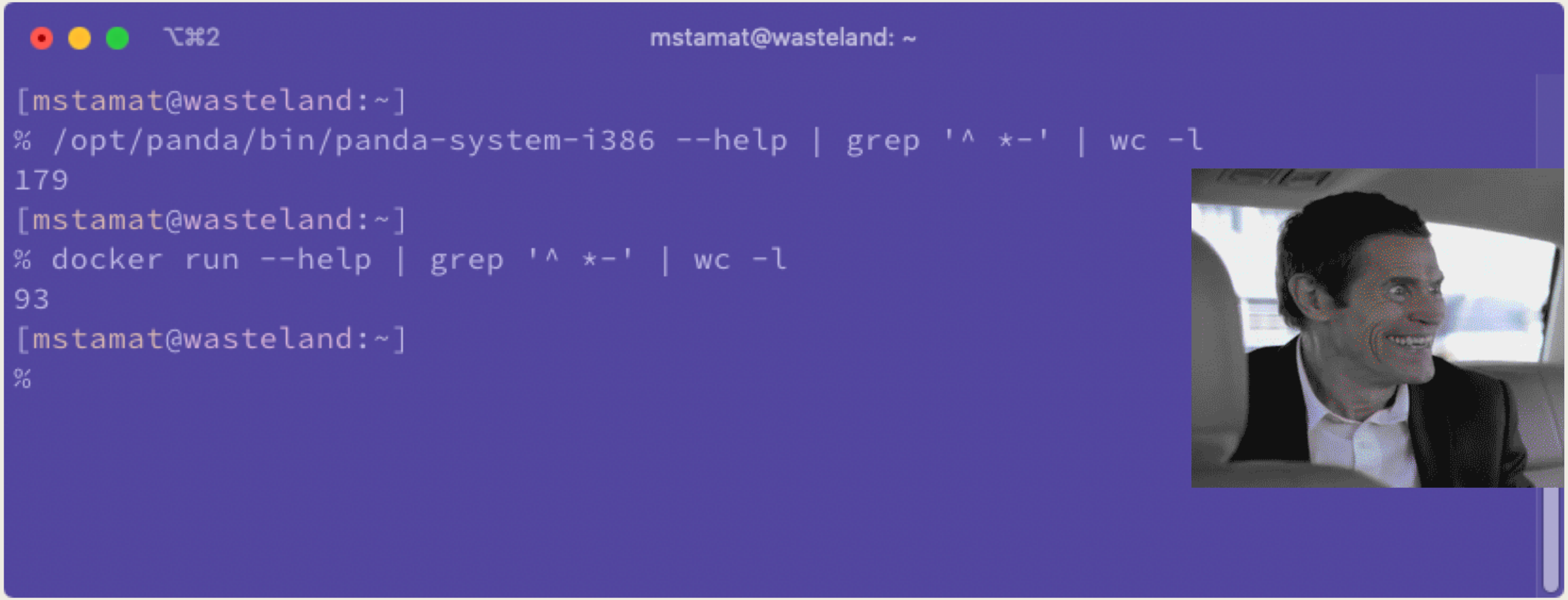

⌥⌘3
mstamat@wasteland: ~/panda.play/pandacap/bootstrap/ssh-honeypot

```
% find run.1 -depth -type f
run.1/id_ed25519
run.1/vm/bootstrap.sh
run.1/vm/scripts/30_config_sshd.sh
run.1/vm/scripts/20_install_recctrl.sh
run.1/vm/scripts/40_config_users.sh
run.1/vm/scripts/30_config_auth.sh
run.1/vm/files/id_ed25519.pub
run.1/vm/files/recctrlu.sh
run.1/vm/files/sshd_config
run.1/vm/files/sftp.txt
run.1/vm/files/id_ed25519
run.1/vm/files/ssh.txt
run.1/vm/bootstrap.env
run.1/docker/bootstrap.sh
run.1/docker/files/id_ed25519.pub
run.1/docker/files/recctrlu.sh
run.1/docker/files/sshd_config
run.1/docker/files/sftp.txt
run.1/docker/files/id_ed25519
run.1/docker/files/ssh.txt
run.1/docker/bootstrap.env
[mstamat@wasteland:~/panda.play/pandacap/bootstrap/ssh-honeypot on master]
%
```

VM runtime bootstrapping

Docker runtime bootstrapping

pandacap.py wrapper



```
[mstamat@wasteland:~]
% /opt/panda/bin/panda-system-i386 --help | grep '^ *-' | wc -l
179
[mstamat@wasteland:~]
% docker run --help | grep '^ *-' | wc -l
93
[mstamat@wasteland:~]
%
```

Most common PANDA/Docker options

PANDA

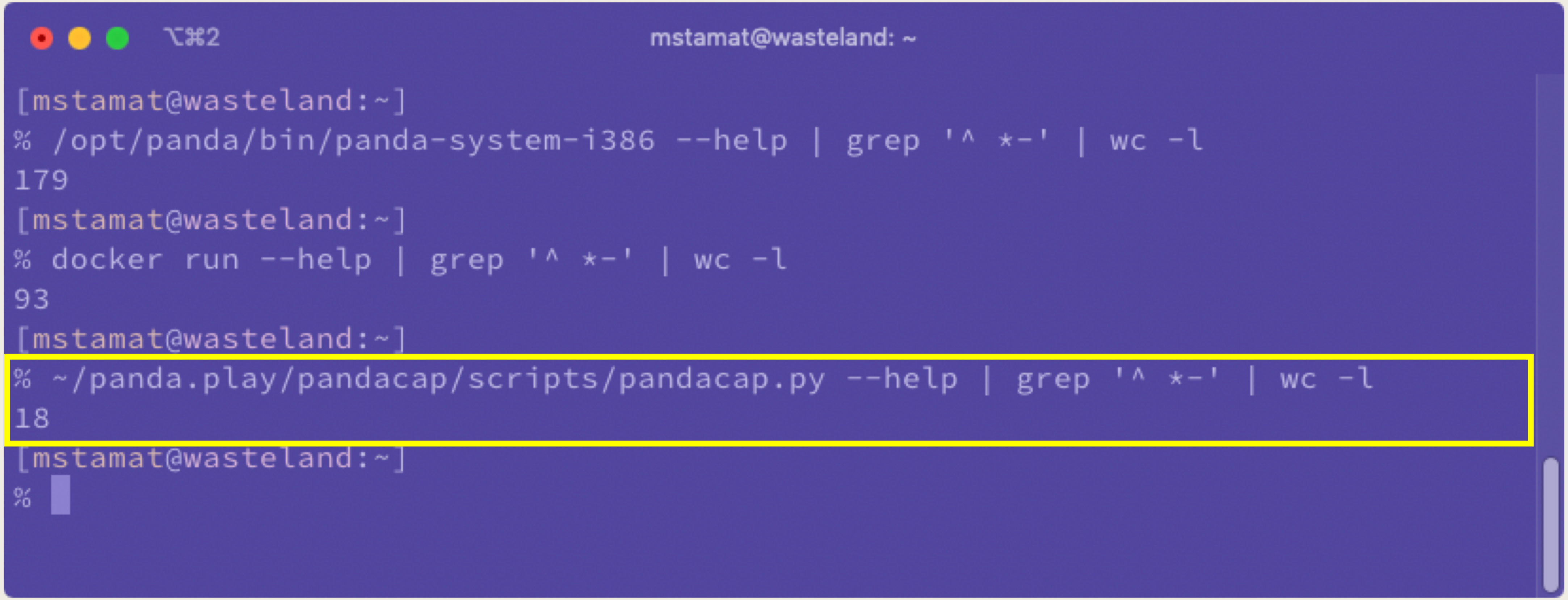
- Disk configuration.
- Network configuration and port forwarding.
- Creation of delta image.*
- Creation of bootstrap disk.*
- Memory/Arch configuration.
- Display configuration.

* Involves additional tools.

Docker

- Mount configuration.
- Network configuration and port forwarding.

pandacap.py wrapper



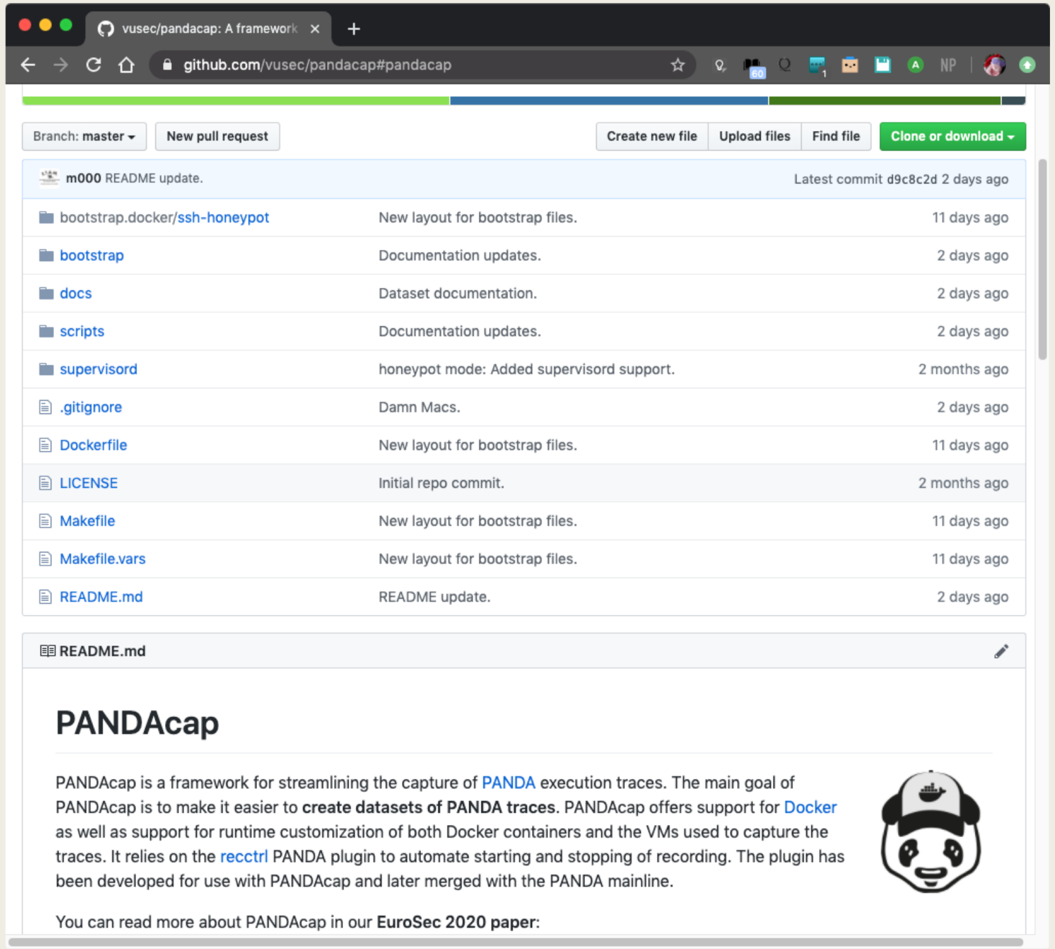
```
[mstamat@wasteland:~]
% /opt/panda/bin/panda-system-i386 --help | grep '^ *-' | wc -l
179
[mstamat@wasteland:~]
% docker run --help | grep '^ *-' | wc -l
93
[mstamat@wasteland:~]
% ~/panda.play/pandacap/scripts/pandacap.py --help | grep '^ *-' | wc -l
18
[mstamat@wasteland:~]
% 
```

pandacap.py wrapper

- All common options in one place.
- Takes care of:
  - Creation of bootstrap disk for the VM.
  - Initialization of a new delta image for the VM.
  - Proper escaping of commands.
- Output files/images are labeled so concurrent runs can be told apart.
- Does not mandate the use of Docker.
  - Can be used as a simple wrapper around PANDA.

PANDAcap source code

github.com/vusec/pandacap



m000 README update. Latest commit d9c8c2d 2 days ago

bootstrap.docker/ssh-honeypot	New layout for bootstrap files.	11 days ago
bootstrap	Documentation updates.	2 days ago
docs	Dataset documentation.	2 days ago
scripts	Documentation updates.	2 days ago
supervisord	honeypot mode: Added supervisord support.	2 months ago
.gitignore	Damn Macs.	2 days ago
Dockerfile	New layout for bootstrap files.	11 days ago
LICENSE	Initial repo commit.	2 months ago
Makefile	New layout for bootstrap files.	11 days ago
Makefile.vars	New layout for bootstrap files.	11 days ago
README.md	README update.	2 days ago

README.md

PANDAcap

PANDAcap is a framework for streamlining the capture of PANDA execution traces. The main goal of PANDAcap is to make it easier to **create datasets of PANDA traces**. PANDAcap offers support for Docker as well as support for runtime customization of both Docker containers and the VMs used to capture the traces. It relies on the recctrl PANDA plugin to automate starting and stopping of recording. The plugin has been developed for use with PANDAcap and later merged with the PANDA mainline.

You can read more about PANDAcap in our **EuroSec 2020 paper**:

Case Study: SSH Honeypot

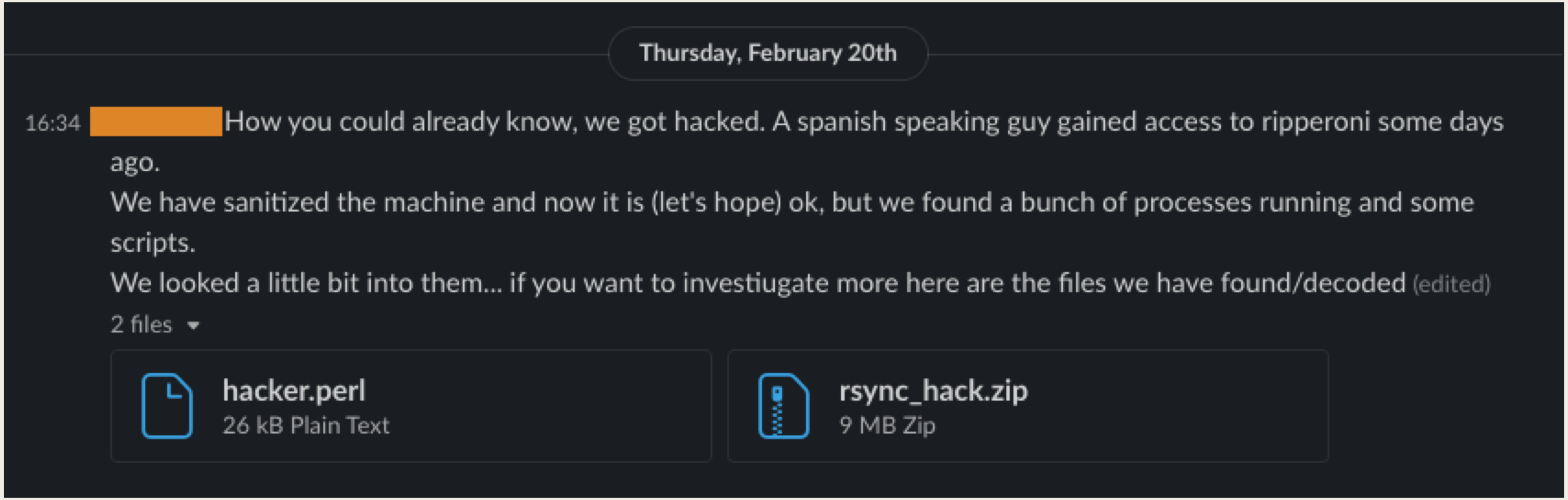
and dataset

PANDAcap Case Study: ssh honeypot

- Brute-force ssh attacks are still popular.
- In their 2016 survey of existing honeypot software, Nawrocki et al. mention no honeypot based on full system Record and Replay.
  https://arxiv.org/abs/1608.06249
- Full system Record and Replay offers significant advantages:
  - Flexibility of analysis.
  - Captures all transient effects on the system.

- Common misconception: Analyzing an ssh intrusion is trivial.

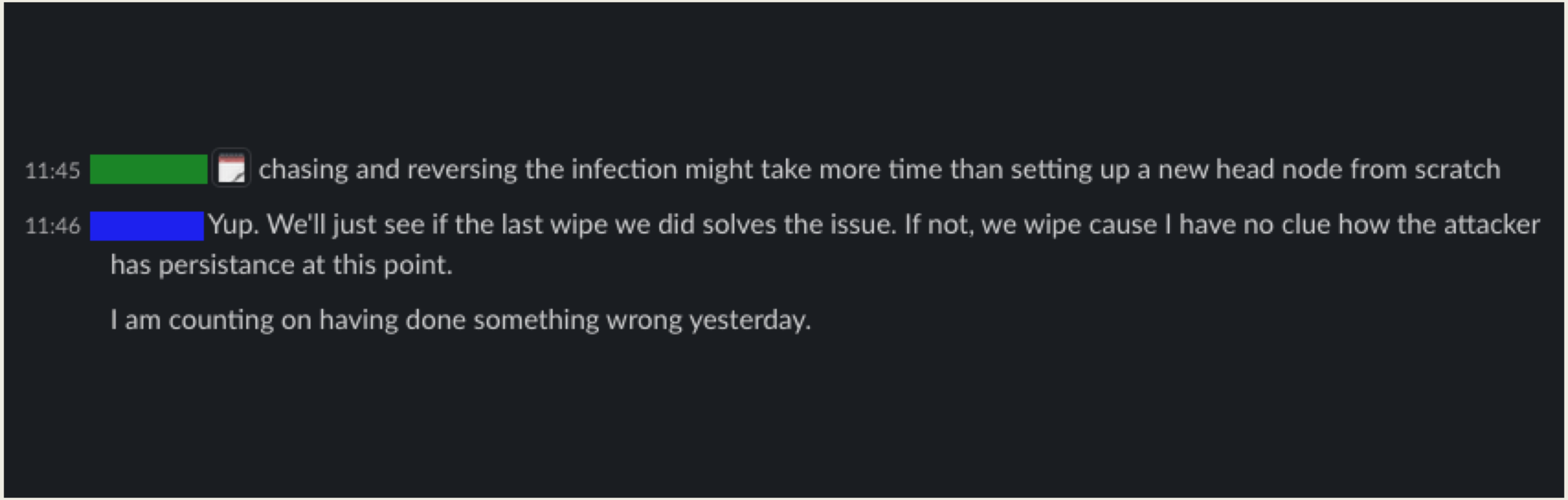
In a Slack channel somewhere…



Thursday, February 20th

16:34 [redacted] How you could already know, we got hacked. A spanish speaking guy gained access to ripperoni some days ago.
We have sanitized the machine and now it is (let's hope) ok, but we found a bunch of processes running and some scripts.
We looked a little bit into them... if you want to investiugate more here are the files we have found/decoded (edited)

In a Slack channel somewhere…

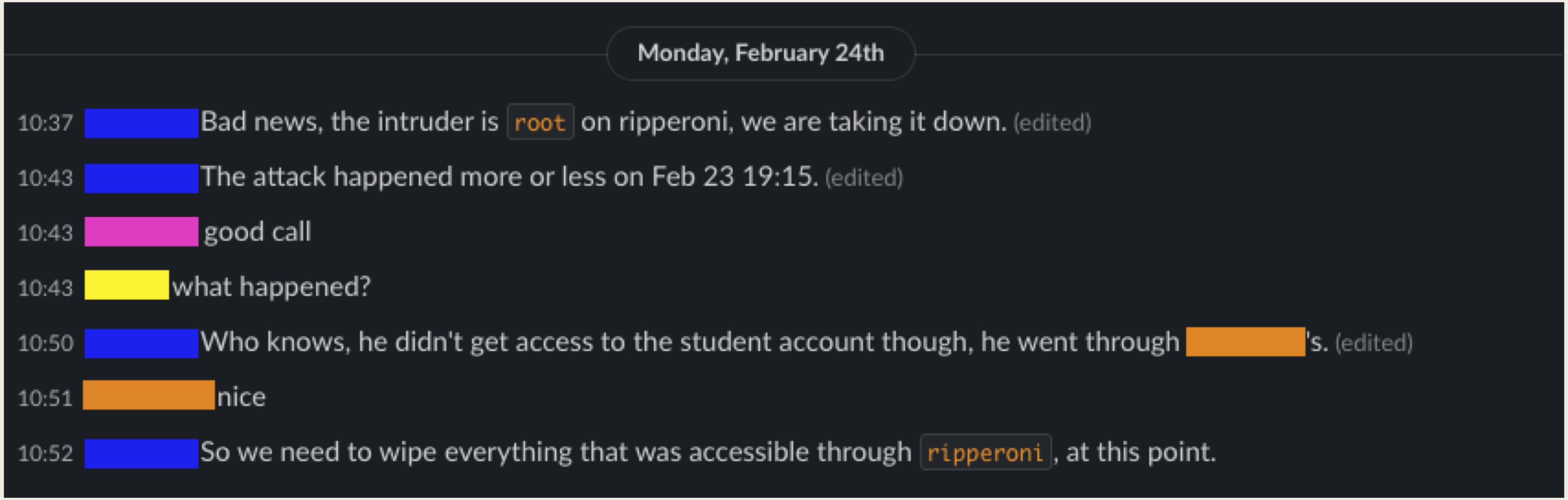


11:45 [redacted] chasing and reversing the infection might take more time than setting up a new head node from scratch

11:46 [redacted] Yup. We'll just see if the last wipe we did solves the issue. If not, we wipe cause I have no clue how the attacker has persistance at this point.

I am counting on having done something wrong yesterday.

In a Slack channel somewhere…



Monday, February 24th

10:37 [redacted] Bad news, the intruder is `root` on ripperoni, we are taking it down. (edited)

10:43 [redacted] The attack happened more or less on Feb 23 19:15. (edited)

10:43 [redacted] good call

10:43 [redacted] what happened?

10:50 [redacted] Who knows, he didn't get access to the student account though, he went through [redacted]'s. (edited)

10:51 [redacted] nice

10:52 [redacted] So we need to wipe everything that was accessible through `ripperoni`, at this point.

Aftermath

- No point of entry was determined.
- Unsure how privilege escalation was achieved.
- Partial recovery of the hacker's tools.
- Partial log of communications.
- Failed to cleanup the machine properly.

- **Post-mortem analysis is hard, even for experts.**
- PANDA system-tracing can provide answers!

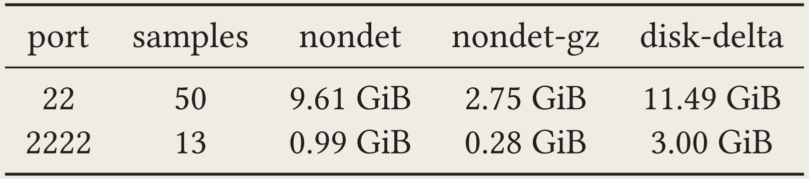
Honeypot analysis with PANDA

- Privilege escalation → exact trace of system calls that led e.g. to a privileged execve
- Hacker tools → ability to fully reconstruct them from the non-determinism log, even if they have been “shredded”
- Communication logs → pcap files + access to unencrypted network stack buffers
- Cleaning up the system → produce a detailed provenance log for all the files that were modified, identify potentially malicious modifications

PANDAcap honeypot dataset

- Ran the experiment for ~3 days on a single IP address.
- Traces limited to 30′.
- Out of 3 ports used, only 2 were visited.
- Collected 63 traces in total.
- Compressed size (including disk deltas) ~23Gb.

Table 1: Collected samples per *ssh* port. No attempts to gain access to the VM listening on port 2200 were made.



port	samples	nondet	nondet-gz	disk-delta
22	50	9.61 GiB	2.75 GiB	11.49 GiB
2222	13	0.99 GiB	0.28 GiB	3.00 GiB

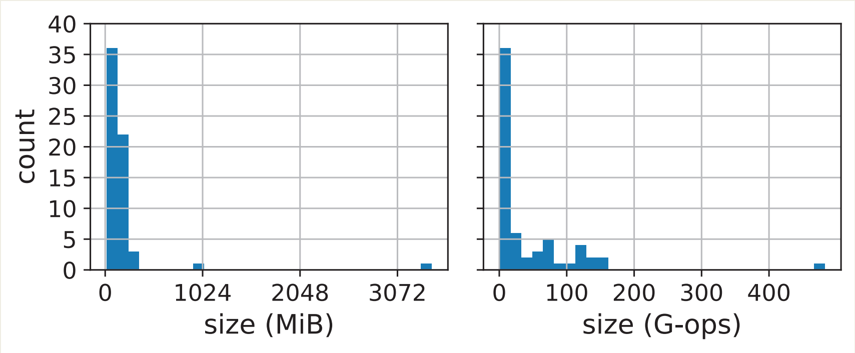


Figure 2: Trace size and instruction count distributions.

PANDAcap honeypot dataset

- Quick qualitative analysis revealed a variance of behaviours.
- Different roles:
  - SSH scanning vs. HTTP/S communication
- Different “return” patterns:
  - 2 logins was the most common case
  - 68 logins was the most common
  - only 2 instances of full log wiping

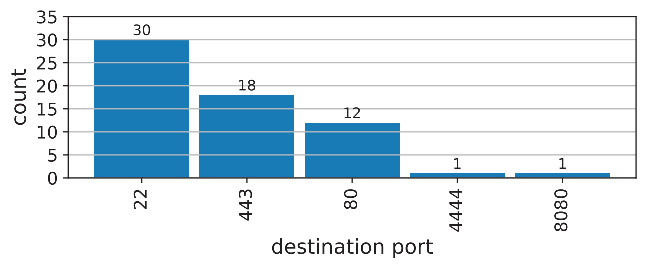


Figure 3: Top target ports for outgoing connections. In one trace, there were no outgoing connections.

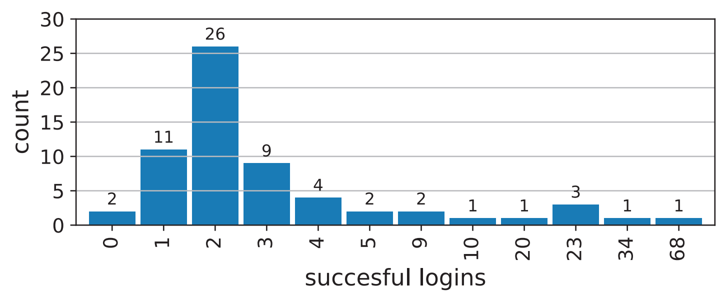
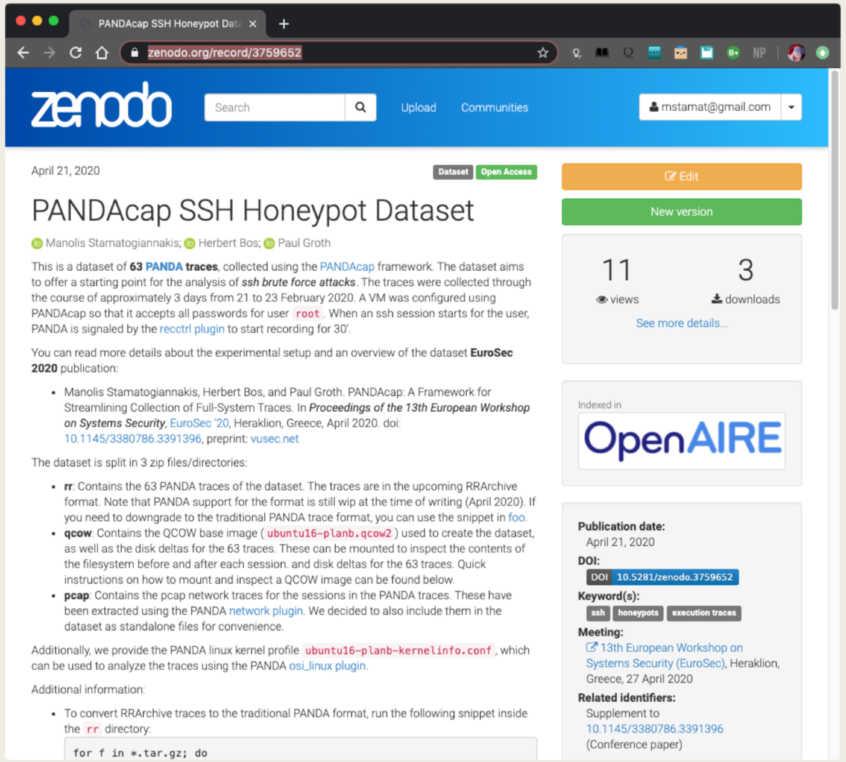


Figure 4: Succesful logins attempts in auth.log.

PANDAcap honeypot dataset availability

zenodo.org (CERN)



April 21, 2020

PANDAcap SSH Honeypot Dataset

Manolis Stamatogiannakis; Herbert Bos; Paul Groth

This is a dataset of **63 PANDA traces**, collected using the PANDAcap framework. The dataset aims to offer a starting point for the analysis of ***ssh brute force attacks***. The traces were collected through the course of approximately 3 days from 21 to 23 February 2020. A VM was configured using PANDAcap so that it accepts all passwords for user `root`. When an ssh session starts for the user, PANDA is signaled by the recctrl plugin to start recording for 30'.

You can read more details about the experimental setup and an overview of the dataset **EuroSec 2020** publication:

- Manolis Stamatogiannakis, Herbert Bos, and Paul Groth. PANDAcap: A Framework for Streamlining Collection of Full-System Traces. In ***Proceedings of the 13th European Workshop on Systems Security***, EuroSec '20, Heraklion, Greece, April 2020. doi: 10.1145/3380786.3391396, preprint: vusec.net

The dataset is split in 3 zip files/directories:

- **rr**: Contains the 63 PANDA traces of the dataset. The traces are in the upcoming RRArchive format. Note that PANDA support for the format is still wip at the time of writing (April 2020). If you need to downgrade to the traditional PANDA trace format, you can use the snippet in foo.
- **qcow**: Contains the QCOW base image (`ubuntu16-planb.qcow2`) used to create the dataset, as well as the disk deltas for the 63 traces. These can be mounted to inspect the contents of the filesystem before and after each session. and disk deltas for the 63 traces. Quick instructions on how to mount and inspect a QCOW image can be found below.
- **pcap**: Contains the pcap network traces for the sessions in the PANDA traces. These have been extracted using the PANDA network plugin. We decided to also include them in the dataset as standalone files for convenience.

Additionally, we provide the PANDA linux kernel profile `ubuntu16-planb-kernelinfo.conf`, which can be used to analyze the traces using the PANDA osi_linux plugin.

Additional information:

- To convert RRArchive traces to the traditional PANDA format, run the following snippet inside the `rr` directory:

```
for f in *.tar.gz; do
```

11 views

3 downloads

See more details...

Indexed in OpenAIRE

Publication date:
April 21, 2020

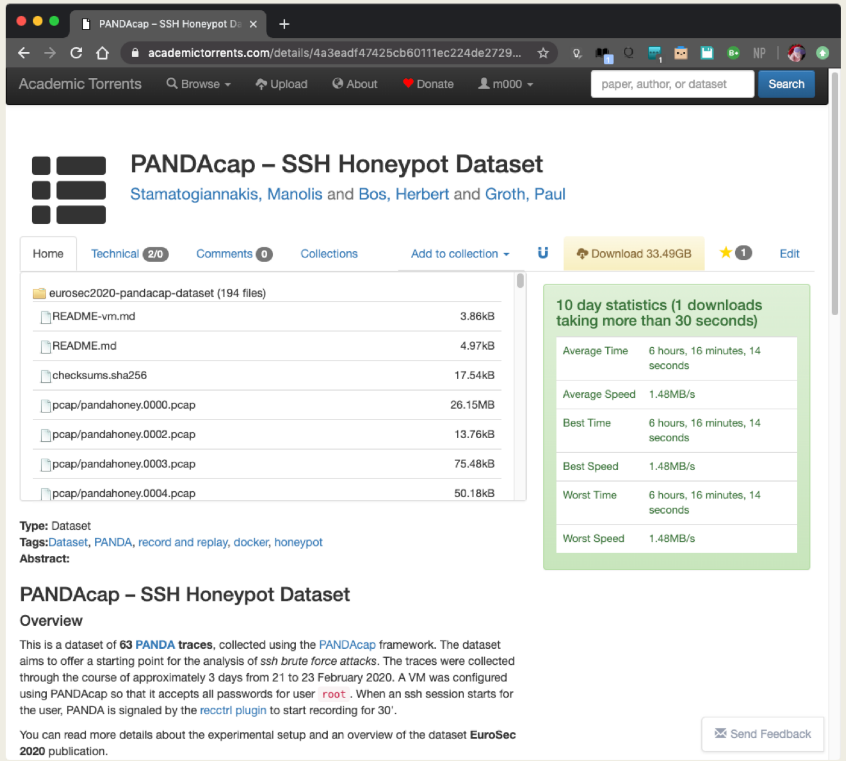
DOI:
DOI 10.5281/zenodo.3759652

Keyword(s):
ssh honeypots execution traces

Meeting:
13th European Workshop on Systems Security (EuroSec), Heraklion, Greece, 27 April 2020

Related identifiers:
Supplement to
10.1145/3380786.3391396
(Conference paper)

academictorrents.com



PANDAcap – SSH Honeypot Dataset

Stamatogiannakis, Manolis and Bos, Herbert and Groth, Paul

eurosec2020-pandacap-dataset (194 files)

README-vm.md	3.86kB
README.md	4.97kB
checksums.sha256	17.54kB
pcap/pandahoney.0000.pcap	26.15MB
pcap/pandahoney.0002.pcap	13.76kB
pcap/pandahoney.0003.pcap	75.48kB
pcap/pandahoney.0004.pcap	50.18kB

10 day statistics (1 downloads taking more than 30 seconds)

Average Time	6 hours, 16 minutes, 14 seconds
Average Speed	1.48MB/s
Best Time	6 hours, 16 minutes, 14 seconds
Best Speed	1.48MB/s
Worst Time	6 hours, 16 minutes, 14 seconds
Worst Speed	1.48MB/s

Type: Dataset
Tags: Dataset, PANDA, record and replay, docker, honeypot
Abstract:

PANDAcap – SSH Honeypot Dataset

Overview

This is a dataset of **63 PANDA traces**, collected using the PANDAcap framework. The dataset aims to offer a starting point for the analysis of *ssh brute force attacks*. The traces were collected through the course of approximately 3 days from 21 to 23 February 2020. A VM was configured using PANDAcap so that it accepts all passwords for user `root`. When an ssh session starts for the user, PANDA is signaled by the recctrl plugin to start recording for 30'.

You can read more details about the experimental setup and an overview of the dataset **EuroSec 2020** publication.

Conclusion

Conclusion

- PANDAcap:
  - easier creation of PANDA trace datasets
  - Docker support
  - streamlined bootstrapping
  - Apache 2.0 license
- PANDAcap SSH honeypot dataset:
  - 63 samples
  - CC 4.0 license

More Information

Code & dataset

github.com/vusec/pandacap

Twitter

#PANDAcap #eurosec2020

@vusec VUSec

@inde_lab_ams INDE lab INtelligent Data Engineering