Hunting Adversaries with "rastrea2r" and Machine Learning

Gabriel Infante-Lopez, @gainlo
Ismael Valenzuela, @aboutsecurity

SANS SOC Summit 2017 – Crystal City, VA
How do we investigate?

- Results of a survey conducted on 16 companies across US, LATAM and EUROPE
  - Up to weeks to make a conclusion.
  - 75% of the time spent on collecting data and 25% of the time spent on analyzing the data
  - 25% of alerts are never triaged and 1/4th of these return as severe or moderate incidents
- Talent shortage

- What resources do we use to drive our analysis? How do they help with scoping or adding context to the investigations?
SOC Capabilities: **Hunting**

SANS 2017 SOC Survey

“The adversaries defrauding and damaging organizations are human [...] **adaptive**, motivated and profitable.”

“Threat hunting with **automated data collection and correlation** improves **the speed** with which analysts can investigate and remediate unknown threats.”

---

**Hunting Means Looking for Something**

The lucky ones are those who are looking for something rather than waiting for it to find them. This involves collecting data (registry, process, file, user, etc.) and correlating it to find suspicious behavior in your enterprise environment. Every correlation and use case built into EDR, SIEM and incident response platforms or homegrown tools stems from an analyst investigating something that is a bit off.

Speeding up collection with **Rastrea2r**

Triage and Hunting for IOCs with ‘gusto’ and style 😊

**Rastrea2r** (pronounced *rastreador*):

- [https://github.com/aboutsecurity/rastrea2r](https://github.com/aboutsecurity/rastrea2r) (opensource!)
- Command line tool (coz command line is SEXY!)
- Python / Multiplatform (win32/64, linux and osx)
- Uses a REST API to report **YARA** scans
- Wrapper to sysinternal, system command and 3rd party tools
- Easy to integrate with McAfee ePO (but also distributable via SSCM, etc.)
- Packaged binaries available on GitHub
Current Functionality In **Rastrea2r** v0.7.1

And it works on Fridays too! (if you dare...)

- **yara-disk**: Yara scan for file/directory objects on disk
- **yara-mem**: Yara scan for running processes in memory
- **memdump**: Acquires a memory dump from the endpoint **Win only**
- **triage**: Collects triage information from the endpoint **Win only**
- **web-hist**: Collects and parses browsing history for all users in the system (ie, firefox, chrome and opera). **Win only** **NEW**
- **prefetch**: Collects and parses prefetch folder from the endpoint **Win only** **NEW**

Obtaining HELP:

```
  rastrea2r_win32_v0.7.1.exe –h
```
Integration with CyLR

- Collects forensics artifacts from NTFS in a fast and secure fashion.
- Bypasses Windows API (raw file collection process)
- Artifacts are collected in memory

By Alan Orlikoski and Jason Yegge

https://github.com/rough007/CyLR

The standard list of collected artifacts are:

"C:\Windows\System32\config"
"C:\Windows\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup"
"C:\Windows\Prefetch"
"C:\Windows\Tasks"
"C:\Windows\SchedLgU.Txt"
"C:\Windows\System32\winevt\logs"
"C:\Windows\System32\drivers\etc\hosts"
"C:\$MFT"
Triage + Hunting On Steroids = **Rastrea2r**

Forensically sound architecture and communication flows
Triaging with ‘rastrea2r’ from McAfee ePO console

Modularity:

• A specific task is created for each combination of command line switches
3rd Party Tools & Native Win Commands

[Image of a computer window showing various files and directories]
Rastrea2r In Action

Threat Name: W97M/Downloader (ED)
Event Generated Time: 2/27/16 2:26:57 AM EST
Threat Target Hostname: J-JM-05NW02-01
Threat Target IP Address: 172.22.32.22
Threat Target File Path: c:\Users\christina.xxxx\Downloads\Rick_Hansinger_Bill.doc\word/vbaProject.bin
Threat Target Username: SYSTEM
Analyzer Detection Method: (managed) Monthly Scheduled ODS
Rastrea2r In Action

1. **Memory capture:** rastrea2r_win32_v0.7.1.exe memdump
tools.myserver.com data.myserver.com

2. **Analyze** with Volatility inside SANS SIFT Workstation
3. **Volatility** plugins *shellbags, timeliner* and *mftparser* provide great output to begin creation of a timeline ([https://volatility-labs.blogspot.com/2013/05/movpii-23-creating-timelines-with.html](https://volatility-labs.blogspot.com/2013/05/movpii-23-creating-timelines-with.html))
4. Web browsing history provides us more context to where the file comes from. Searching for the output of this tool for activity on 1/27/2016 at around 22:51 UTC show the following events:
Evidence of **persistence**: sysinternals autoruns
Evidence of **persistence**

Sysinternals startup-list:

- Caption=hizee
- Command=hizee.lnk
- Description=hizee
- Location=Startup
- SettingID=
- User=DOMAIN\XXXXXXXXX
What are the shortcomings?

3 different analysts writing scripts to parse the output of rastrea2r:

• “Visual spoofing”
• Reputation of an IP address
• Process name entropy

Can the 3 analysts reuse each other’s scripts?

• No input/output standardization = no piping
• No agreement on how objects are manipulated
• No data traceability
From Collection to Mining
Use case example: Visual Spoofing

Visual spoofing are attack vectors that leverage characters from various languages that are visually identical to letters in another language.

As an analyst, I would like to detect all process names from an endpoint snapshot that contains some form of “visual spoofing” or critical process impersonation.
Visual Spoofing: Implementing a visual spoofer.

```python
def VisualSpoofing(str):
    # Check if str is mixed script
    if(is_mixed(str)) return []

    # Find strings in 'db' that are at 'dist' from 'str' using Liblevenshtein distance
    first_candidates = liblevenshtein(str, dist, db)

    # Fine tuning over 'first_candidates' using the Smith-Waterman local aligner
    last_candidates = sw_local_aligner(str, threshold, first_candidates)

    return last_candidates
```
Development Process

Sources: Different sources produce different content with different formats

Distribution: Using third-party function might be problematic.

- Stacking process: “Apply this process to snapshot containing files with VS only”

Sources:

- Endpoint Snapshot
- Substring parser
- [process name]
- Visual spoofing function
- Visual spoofing report

Collection of files → Substrings of interest → Strings with results

Parsing: Writing parsing is time consuming and ad-hoc

Traceability: “In which snapshot was this process?”
Problem Statement

• The landscape of community-driven ML for security.
  • no-common vocabulary,
  • lack of interoperability
  • lack of portability
• What is missing?
  • Connecting Collection to Mining
Solution: **OpenCNA** (Collect, Normalize & Analyze)

- JSON Definition of entities (file, user, system, etc)
  - close to CyBox when appropriate
- Python SDK
  - containing python wrappers for JSON
- Simple Infrastructure to pipe and report
- Sample Miners: Visual Spoofing Detector
Visual Spoofing: Implementing a visual spoofer.

```python
def VisualSpoofing(process):
    @process: Process
    @out Process with VisualSpoofing
    # Check if str is mixed script
    str = process.name
    out.print("process name: " + str)
    if(is_mixed(str)) return []
    # Find strings in ‘db’ that are at ‘dist’ from ‘str’ using Levenshtein distance
    first_candidates = liblevenshtein(str, dist, db)
    # Fine tuning over ‘first_candidates’ Smith-Waterman local aligner
    last_candidates = sw_local_aligner(str, threshold, first_candidates)
    return appendVS(process, new VS(last_candidates))
```
Pipelining Process

SnapShots → entity extractor → Sorter → miner → report → html → final report
Pipelining Process

Sources: Entity extractors for Rastrea2r.

Parsing: Logs to JSON

Distribution: Using Docker

Stacking process: Signature Definition

Traceability: Extending types

User defined entity extractor

SnapsShots Entities Sorter

report results final report html

1 2 3 4 5 miner miner miner miner report report report report
OpenCNA
OpenCNA (Collection & Normalization & Analysis) - Coming soon!
Updated just now

rastrea2r
Forked from aboutsecurity/rastrea2r
Collecting & Hunting for IOCs with gusto and style
Python 18 Updated an hour ago

InvestigationPlaybookSpec
Call to action

Checkout our public repos on GitHub: https://github.com/Foundstone/

Send us feedback!
- ismael_valenzuela@mcafee.com
- gabriel_infante-lopez@mcafee.com

Follow us on twitter for updates @Foundstone @aboutsecurity @gainlo