Copyright 2015 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This material has been approved for public release and unlimited distribution except as restricted below.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

Carnegie Mellon®, CERT® and FloCon® are registered marks of Carnegie Mellon University.

DM-0002067
Definition

“Indicator expansion is a process of using one or more data sources to obtain more indicators of malicious activity by identifying those related to currently known indicators.”

~ Some guy named: Jono Spring 2013
Generic Situation

1. Our host communicates with known bad IP address
2. Host gets infected
3. Host communicates with a different IP for:
   • Command and control
   • Exfiltration

Let’s try and find these second-level IP addresses
• They’re bad
What we need to do

1. Detect our host communication with black list IP
2. Keep a list of these hosts
3. Track the IPs where these hosts send traffic
4. Count how many hosts contact each IP
5. Alert if some number of hosts contact an IP
6. Record those IPs in alerts and/or IP Sets
Disclaimer

This algorithm is generic

Threshold values in the example are just examples, they are not to be used

This is not being run anywhere

Illuminates a way Analysis Pipeline can implement existing analysis ideas
Needs / Decisions

• Need: Accepted malicious IP list
  • SiLK IPSet: badIPs.set will contain these IPs
• Need: White list of IPs where our hosts often communicate with
  • SiLK IPSet: safePopularIPs.set will contain these IPs
• Decision: Track our hosts for 1 day
• Decision: Use 50 hosts contacting second level IP as the threshold, within a 36 hour time window
• Decision: Dump list of second level IPs in both an alert and IPSet file every 6 hours
Analysis Pipeline overview

- Version 4.4.1 publicly released:
  - tools.netsa.cert.org/analysis-pipeline
- Streaming analysis of SiLK records
- Filters
- Internal Filters – “scratch paper”
- Evaluations / Statistics
  - Can bin state based on value of specified field
- Configuration file tells Pipeline what to do
  - Simple config files accomplishes our entire scenerio
Mechanics of Flow Collection

- **YAF**
  - rwflowpack
  - rwsender

- Pipeline
  - flow copy
  - directory

- SiLK
  - Data store
Steps 1 & 2 – Detect and Track

FILTER badTraffic
  DIP IN LIST “badIPs.set”
END FILTER
INTERNAL FILTER trackInfectedHosts
  FILTER badTraffic
  SIP infectedHosts 1 DAY
END INTERNAL FILTER
Step 3 watch where infected hosts go

FILTER nonWhiteListPostInfected
    SIP IN LIST infectedHosts
    DIP NOT IN LIST safePopularIPs.set
END FILTER
Step 4 & 5: Count Hosts Per IP and Alert

EVALUATION secondLevelPopularIPs
  FILTER nonWhiteListPostInfected
  FOREACH DIP
  OUTPUT TIMEOUT 1 DAY
  OUTPUT LIST DIP secondLevelIPs
  <alerting options…not discussed>
  CHECK THRESHOLD
    DISTINCT SIP > 50
    TIME WINDOW 36 HOURS
  END CHECK
END EVALUATION
Step 6: Report Expanded Indicators

LIST CONFIGURATION secondLevelIPs
  UPDATE 6 HOURS
  SEED “latestSecondLevelIPs.set”
  OVERWRITE ON UPDATE
END LIST CONFIGURATION
Full Configuration – not so hard

FILTER badTraffic
  DIP IN LIST “badIPs.set”
END FILTER
INTERNAL FILTER trackInfectedHosts
  FILTER badTraffic
  SIP infectedHosts 1 DAY
END INTERNAL FILTER
FILTER nonWhiteListPostInfected
  SIP IN LIST infectedHosts
  DIP NOT IN LIST safePopularIPs.set
END FILTER

EVALUATION secondLevelPopularIPs
  FILTER nonWhiteListPostInfected
  FOREACH DIP
  OUTPUT TIMEOUT 1 DAY
  OUTPUT LIST DIP secondLevelIPs
  <alerting options…not discussed>
  CHECK THRESHOLD
  DISTINCT SIP > 50
  TIME WINDOW 36 HOURS
END CHECK
LIST CONFIGURATION secondLevelIPs
  UPDATE 6 HOURS
  SEED “latestSecondLevelIPs.set”
  OVERWRITE ON UPDATE
END LIST CONFIGURATION
Questions/comments?

druef@cert.org
netsa-help@cert.org