OWASP Web Honeypot Project - Application Honeypot Threat Intelligence

• adrian.winckles@owasp.org
Bio – Adrian Winckles

• Director of Cyber Security, Networking & Big Data Research Group, Anglia Ruskin University, Cambridge.
• OWASP Activities
  – OWASP Cambridge Chapter Leader,
  – OWASP Europe Board Member
  – Project Leader – OWASP Web Honeypot Project
  – Project Leader – OWASP Application Security Curriculum Project
• Chair Cambridge Cluster of the UK Cyber Security Forum.
• Vice Chair of the BCS Cyber Forensics Special Interest Group.
Introduction to Honeypots

• A computer system setup to detect or lure attacks.

• Honeypot types:
  – Production (detect)
  – Research (lure)

• Honeypot interaction types:
  – Low - emulated services, limited to no emulated login capability (low risk).
  – Medium - emulated services, emulated login, emulated commands.
  – High - Actual services, system logins, and commands (very risky).
A production honeypot has no legitimate business purpose and should never see any traffic, unless...

– Something is misconfigured on the network
– Someone is malicious on the network

Honeypot logs are low volume and high value
Why OWASP Web Honeypots (Part 1)?

• Sector focus is on HTTP(S) today
• According to CAIDA, (Center for Applied Internet Data Analysis) web is ~85% of total internet traffic.
• 92% of vulnerabilities now in the application (NIST/Gartner)
Why Web Honeypots?
Why OWASP Web Honeypots (Part 2)?

• Focus is on HTTP(S) today
• According to CAIDA, (Center for Applied Internet Data Analysis) web is ~85% of total internet traffic.
• 92% of vulnerabilities now in the application (NIST/Gartner)
• Web architecture is complicated
• It also means complicated attacks are acceptable
• Attacks that will only work on 0.01% of users are valuable
The Web is Complicated
Why OWASP Web Honeypots (Part 3)?

- Focus is on HTTP(S) Today
- Special care needs to be taken here
- According to CAIDA, (Center for Applied Internet Data Analysis) web is ~85% of total internet traffic
- As a result web architecture is complicated
- It also means complicated attacks are acceptable
- Attacks that will only work on 0.01% of users are valuable
- Diversity of attacks is high as well (number of variations)
  - Attacker on server / Attacker on client
  - Attacker on client via server
  - Attacker on server via server
  - Attacker on intermediary
What do we want to capture?

• Think about using existing tools so that you can catch automated web attack tools that are scanning IP network ranges looking for web ports.

• Instead of developing and deploying an entirely new honeypot web server or application, we can easily reuse the existing legitimate web server platform’s organisations are already running.
Consider the WAF - Web Application Firewall

- WAFs Come in multiple different forms
The WAF as a Honeypot or Probe?

• WAFs Come in multiple different forms
  • Can be placed in several places on the network
    • Inline
    • Out-of-line
    • Load balancer mirror port
    • On the web server
• Different Technologies
  • Signatures
  • Heuristics
• Often driven by PCI requirements, as it’s an approved security control
• What is the difference between an IDS versus WAF?
ModSecurity - An Open Source Web Application Firewall

• Probably the most popular WAF
  – Designed in 2002
  – Currently on version 2.9.1 with version 3.0 in the works

• Designed to be open and supports the OWASP Core Rule Set
  – First developed in 2009
  – An OWASP project meant to provide free generic rules to ModSecurity users
  – CRS v3.0 now deployed
ModSecurity’s Apache Request Cycle Hooks
What is the OWASP Core Rule Set (CRS)?

- A generic, plug-n-play set of WAF rules
- Choose your mode of operation
  - Standard vs. Anomaly Scoring
- Detection Categories:
  - Protocol Validation
  - Malicious Client Identification
  - Generic Attack Signatures
  - Known Vulnerabilities Signatures
  - Trojan/Backdoor Access
  - Outbound Data Leakage
  - Anti-Virus and DoS utility scripts
CRS Traditional Detection Mode – *Birth of a Honeypot Probe*

- IDS/IPS mode with “self-contained” rules
- Like HTTP itself – the rules are stateless
  - **No intelligence is shared between rules**
  - If a rule triggers, it will execute a disruptive/logging action
- Easier for the new user to understand
- Not optimal from a rules management perspective (handling false positives/exceptions)
- Not optimal from a security perspective
  - Not every site has the same risk tolerance
  - Lower severity alerts are largely ignored
Event Logging - *Standard vs. Correlated Events*

- **Standard mode**
  - Rules log event data to both the Apache error_log and the ModSecurity Audit log can be relayed using mlogc http/json

- **Correlated mode**
  - Basic rules are considered reference events and do not directly log to the Apache error_log
  - Correlation rules in the logging phase analyze inbound/outbound events and generate special events
    - `modsecurity_crs_60_correlation.conf`
Modsecurity Log Collector (mlogc) – Event Logging

AuditConsole

Event Processor

Database

Event Browsing

Apache

ModSecurity

E-Mail Notification

Event Deletion

Script Execution (planned)
Project Aims & Objectives

• The OWASP Honeypot Project provides:
  – Real-time, detailed Web Application Attack Data
  – Threat Reports to the community

• What do we need
  – Volunteers to run honeypots/probes in their network
  – Contributor’s to the project
Automated Web Attacks using OWASP ZAP

Project Test Bed

VM Based WAF Probes

WASC Honeypot Sensor

WASC Honeypot Sensor

WASC Honeypot Sensor

Audit Console (Apache Webserver)

Audit data passed to PHP script and logged to MySQL

Audit data

-mlogc HTTP audit log data
Distributed Probes Model
"Security is lax on this side."
Ongoing & Future Work

• Setup Proof of Concept to understand how Mod Security baed Honeypot/Probe interacts with a receiving console (develop a VM and/or Docker based test solution to store logs from multiple probes) **DONE**

• Evaluate console options to visualise threat data received from ModSecurity Honeypots/probes in ModSecurity Audit Console, WAF-FLE, Fluent and bespoke scripts for single and multiple probes. **Ongoing**

• Develop a mechanism to convert from stored MySQL to JSON format.

• Provide a mechanism to convert ModSecurity mlogc audit log output into JSON format.

• Provide a mechanism to convert mlogc audit log output directly into ELK (ElasticSearch/Logstash/Kibana) to visualise the data.
Ongoing & Future Work (cont’d)

• Provide a mechanism to forward honest output into threat intelligence format such as STIX using something like the MISP project (https://www.misp-project.org) to share Threat data coming from the Honeypots making it easy to export/import data from formats such as STIX and TAXII., may require use of concurrent logs in a format that MISP can deal with.

• Consider new alternatives for log transfer including the use of MLOGC-NG or other possible approaches.

• Develop a new VM based honeypot/robe based on CRS v3.0.

• Develop new alternative small footprint honeypot/probe formats utilising Docker & Raspberry Pi.

• Develop machine learning approach to automatically be able to update the rule set being used by the probe based on cyber threat intelligence received.
Any Questions?