Imperial College London

446H – Applied Network Security

2. Intrusion detection

Dr Sergio Maffeis
Department of Computing
Course web page: https://446h.cybersec.fun

Project 2

Imperial College London

- **Intrusion detection based on Web logs**
- Goal: develop tools to analyse web logs and identify attempted or successful attacks against web applications
- 3 phases
  1. signature based detection (25%)
  2. anomaly detection (50%)
  3. comparison of approaches via analysis of real logs (25%)
- You are given 2 months of HTTPS and error logs from the DOC wwwhomes server
  - Respect confidentiality: use only for project, delete after
  - Respect privacy: privacy concerns are out of scope of the project, focus only on security
  - You may or may not want to use the error logs

Budget your effort

Imperial College London

- Each group member should aim to spend about 12 hours on the project
- It will not be possible to look for any conceivable kind of attack. Pick your choices, but include at least some form of detection of SQL injection attempts.
- You are very likely to report (many) false positives
  - Tool configuration should allow some form of control of the FP/FN trade off
  - Make sure your tools are tuned to find something in the given log data

Phase 1

Imperial College London

- Signature based command line tool
  - inputs
    - a file containing signatures
    - a configuration file
    - a directory containing web logs
  - output
    - log containing only suspicious log entries
    - annotated with a severity score and an indication of the reason why the entry is considered suspicious
  - write your own signatures and/or use relevant Snort or ModSecurity rules

Phase 2

Imperial College London

- Anomaly-detection based command line tool
  - inputs
    - configuration file
    - directory containing web logs
  - output: same as phase 1
- Note: you are not provided with data guaranteed to contain only benign behaviour.
- Advice: build different models for different applications
  - requests under different home directories
  - requests for resources of different types

Phase 3

Imperial College London

- Run both tools on the given logs
  - analyse and compare the results
  - produce a joint log with the top 10 entries that you consider the most severe and representative
    - irrespective of the severity score that was assigned automatically
  - Write up with attack analysis in the report

Requirements

Imperial College London

- Tools should be command line and work on a standard DOC Linux distribution
  - minimal effort for installation and usage
- Tools can be implemented in any language compatible with point above
  - Bash, Python, Node, C, Java
  - ELK, Splunk: out of scope
- Your tools should run successfully and not crash on the provided logs.
  - They will be tested on analogous but different logs from the same set of web applications.
- Performance is not a main concern, but execution on the full dataset should not take more than a few hours.
- Tools should produce annotated output logs
  - sub-log of input log (only suspicious entries)
  - same format as input logs plus 2 fields: a numerical severity score and a reason why

Report

Imperial College London

- 3-4 pages long
- Write a section for each of the 2 tools, describing clearly and concisely:
  - what specific issues does your tool aim to identify, and how;
  - how to use the tool and interpret its output;
  - what are the key technical ideas and/or challenges in your implementation;
  - what would you do next if you had more time.
- Write one evaluation section describing phase 3
  - provide a commentary on your top-10 entries.

Discussion

Imperial College London

- What attacks can we see from HTTPS web logs?
- Some examples
  - attempts to deliver a typical web application attack, such as
    - SQL injection, XSS, Path traversal, File inclusion, Forceful browsing, etc
  - attempts to identify the presence of known vulnerable web applications
  - suspicious TLS-level activity
  - accesses from malicious IP
  - suspicious UserAgent strings
  - anomalous frequency of requests
  - attempts to brute-force authentication.