Scalable Application Assessments in the Enterprise

Tom Parker / Lars Ewe
Securicon, LLC / Cenzic

AppSec DC
November 13th 2009

The OWASP Foundation
http://www.owasp.org

Agenda

- The trouble with Application Assessments and Scale
- Pros & Cons of Application Automation
- Automation & Compliance
- Non Web-Based Application Assessments

Why Fully Manual Assessments Don't Scale (well)

- It's (generally) a people problem
  - (Good) Application Testers do not grow on trees
- Often requires a background in Application Development
  - To understand how given functionality may be implemented
  - In order to foresee mistakes that may have been made
  - And therefore find, and exploit (where applicable) them
- Money Can (in part) Fix this
  - Many large organizations invest heavily in app testing talent
  - But it still doesn't scale!

Throwing Money at the Problem

- Conservative Salary of Application Tester Talent ~ 140k
- Throw in SG&A .. 180k

- And even then..
  - In a large enterprise, most applications will be lucky if they get looked at more than once a year.

Transactional Application Assessments and Automation

- Two schools of thought in the security industry, which generally prevent good, readily scalable application assessments:
  - Automation is all we need, we can pretend to be security experts by relying on automation and undercut our competition.
  - Automation is the devil, it's generic, misses findings and makes us look like skr1pt k1dd13s.

- Both schools have some legitimate origins, however things have changed significantly in the assessment automation space in the past five years.

Why Too Much Automation can be Bad

- Automation is mostly bad when the people running it rely on it too heavily, without knowing too much about the application and environment.

- A degree of understandings for the application being tested, and the tests being conducted remain of high value in order to:
  - Place risk based context around issues identified
  - Weed out false positives
  - Ensure that automated tests are appropriate for environment under test (application technology & supporting architecture)

However

- Automation is your friend.
- It helps you be efficient.
- Finds the easy stuff..
  - Which leaves you more time for the fun stuff!

Evolution Of VA Automation

- It all started out with manual testing, assisted by some basic tools, like intercepting proxies, simple fuzzers, etc.
- Out of that evolved more special purpose test suites, like Burp Suite, Paros, WebScarab, etc.
- In parallel more automated, commercial scanners emerged: IBM AppScan, HP WebInspect, and Cenzic Hailstorm
- For the longest time most automation aspects of VA were strictly signature based
- That changed with the introduction of behavioral based VA automation
  - Try to maintain app state
  - Observe app behavior, rather than static response content

  ➔ More accurate results; less false positives & negatives

Solution Requirements (Basic)

- Ability to map / analyze the target application
  - Learn about application structure and behavior
  - Technology fingerprinting (e.g. AJAX / Flash)
  - Identify session management, login/logout & authentication mechanisms (incl. change password & register functionality), etc.
  - Automatic detection of data-driven variations of pages (e.g. each book on Amazon)
  - Etc.

Solution Requirements (Basic) – contd.

- Ability to traverse / crawl the target application
  - Automated crawling, recorded (proxy & ideally also gesture based – e.g.), and manual crawling, combination thereof
  - Manage session identifiers, login/logout & authentication mechanisms (incl. change password & register functionality), etc.
  - Ability to train forms: random date-ranges, random values, unique (one-time) values (e.g. unique email address or passwords), etc.
  - Web 2.0: Perform mouse events, JavaScript links, Flash menus, etc.
  - Ability to define white lists, black lists, depth vs. breadth first spidering, max. # of pages / depth / time, "uniqueness rules", etc.

Solution Requirements (Basic) – contd.

- Ability to attack / assess the target application
  - Automated attack vectors, updated regularly (think AV defs)
  - Configurable: Attack throttling, attack order, injection values, control injection targets (headers, cookies, parameters) through back & white lists, field-at-a-time vs. parallel attacks, support various encodings, etc.
  - Web 2.0: JSON, Flash, AMF, etc.
  - Customizable: Ability to define custom attacks (ideally based on out-of-the-box attacks), custom injection values, etc.
  - Low false positives & negatives
- Ability to generate reports
  - Configurable, customizable, support for various formats
  - Various out-of-box compliance reports (PCI, HIPAA, OWASP, etc.)

Solution Requirements (Enterprise)

- Ability to scale
  - Run many assessments in parallel
  - Support variety of different deployment topologies
- Support best practice workflows, allow for company wide collaboration
  - Integrate with 3rd party systems (defect tracking, LDAP, etc.)
  - Role based access and solution views
  - Email event notifications
  - Access and event logging
- Manage company wide application portfolio / risk management
  - Auto-discover apps
  - Assess them and compare them by risk
  - Manage thousands of apps (scalability)
  - Automatically retest regularly ➔ Trending

VA Automation And Web 2.0

- Spidering is more complex than just processing ANCHOR HREF's; various events need to be simulated (e.g. mouseover, keydown, keyup, onclick, onfocus, onblur, etc.)
- Timer events and dynamic DOM changes need to be observed
- Use of non-standard data formats for both requests and responses make injection and detection hard to automate; need to support JSON, XML, serialized data, etc.
- Page changes after XHR requests can sometimes be delayed
- In short, you need to have browser like behavior (JavaScript engine, DOM & event management, etc.)

Application Assessments & Compliance

- Most Compliance Tests Small Subset of Universe of Possible Application flaws.
- Automation is great for anything that requires checks of a fixed sub-set of tests.
- Finding validation remains CRITICAL.

Non Web Based Applications

- Thick Clients
- Proprietary Server Components

Automation Solutions

- Much like Web Application Testing Technologies, Automation in the Non-Web Space has Advanced Significantly.

- Source Code Analysis Tools
  - Fortify SCA, Ounce Labs et al
- Static Analysis & Binary Disassembles / Decompilers
  - IDA Pro, Hex-Rays
- Fuzzing Frameworks
  - Mu Dynamics, Peach Fuzz, Codenomicon Defensics
  - Most Cover Both Network Protocols and File Format Fuzzing

Proprietary Network Protocol Case Study

- Most 'Proprietary' Protocols are really not that proprietary.
- Efficient Assessment of Applications based on Proprietary Network Protocols often a question of selecting the correct automation tool.
- Mu Dynamics – Mu Studio Tool:
  - Automates Analysis of IP Based Protocols
  - Creation of Protocol "Mutation Routines"
  - Establishes inter-protocol field relationships

Automation & Reporting

- Reporting interfaces have come a long way
- Speeds up often arduous process of documenting findings – still leaving scope for details customizations.
- Helps standardize finding class descriptions, data point references and reporting formats.

Summing it Up – Automation & Cost

- Internal Application Team – No Automation
  - Five Sr. Testers + Manager – $950,000.00
- Internal Application Team – Automation
  - Two Mid-Level Testers + Manager
  - Copy of SCA Tool
  - Common Application
  - $500,000
- But – it isn't just about cost

Summing it Up - Continued

- More Regular Assessments of Applications
- Easy Integration into Organizational SDLC
- Test Harness Integration (Quality Center etc)
  - Does anyone like filing tickets anyway?

Questions?