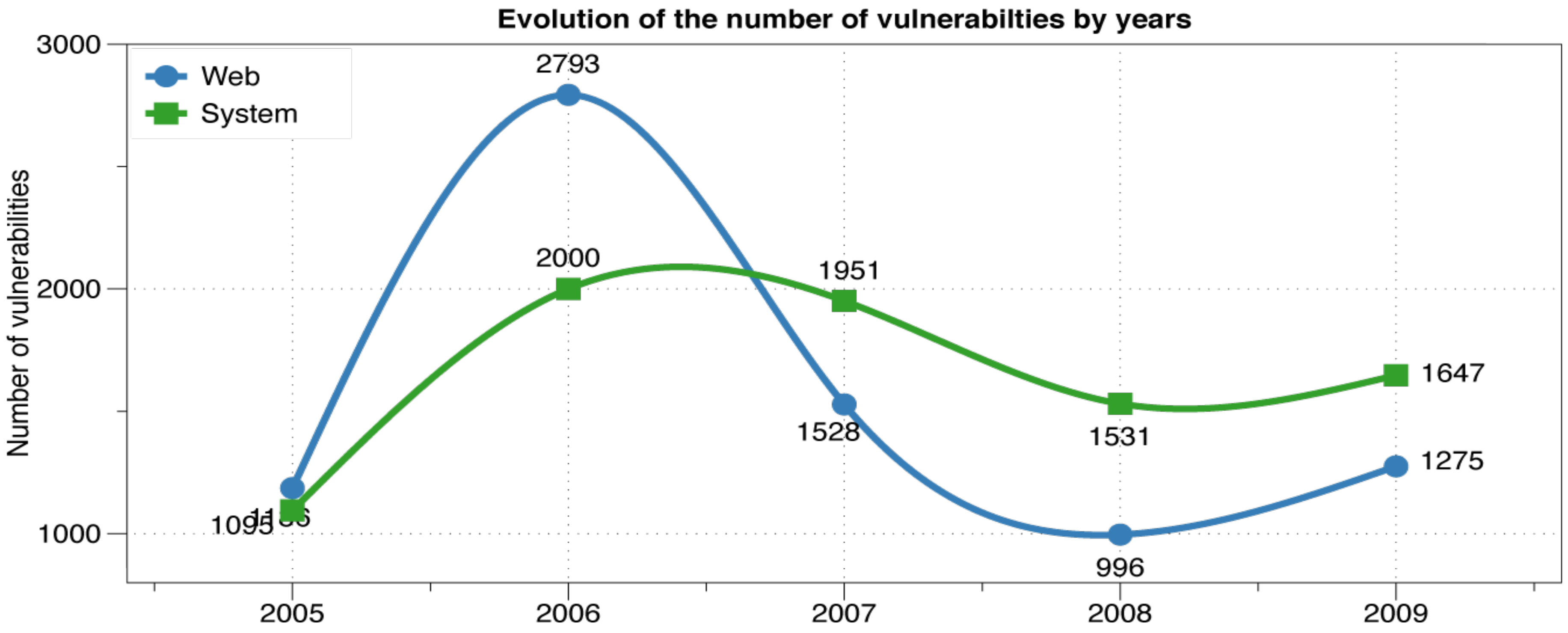
State of The Art:
Automated Black Box
Web Application Vulnerability Testing

Jason Bau Elie Bursztein
Divij Gupta John Mitchell

Stanford Computer Security Lab

Background

- Web Application Vulnerability Protection
  - High incidence vulnerabilities (XSS, SQLI, …)
  - Required for standards compliance (e.g. PCI)

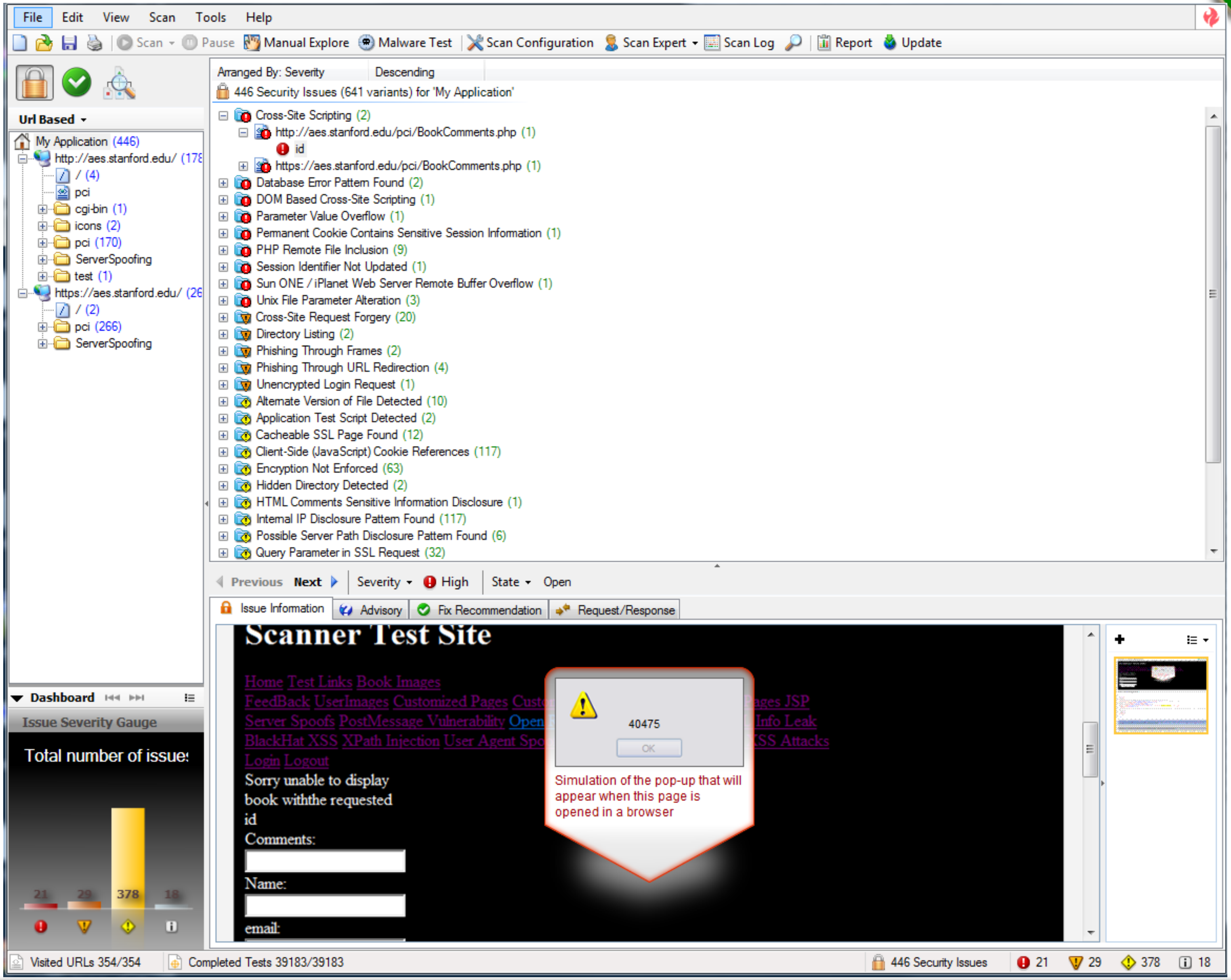


Data from VUPEN

Security Tools for Web Apps

- Vulnerability Detection Techniques:
  - Manual vs. Automated
  - White-Box vs. Black-Box
  - Code review, Static analysis, Pen tester
  - **Automated Black Box Testing**
    - Cheaper? Less intrusive to workflow?
    - Accepted method of PCI compliance

Scanner 1


File Edit View Scan Tools Help
Scan Pause Manual Explore Malware Test Scan Configuration Scan Expert Scan Log Report Update
Url Based
Previous Next Severity High State Open
Issue Information Advisory Fix Recommendation Request/Response
Dashboard
Issue Severity Gauge
Total number of issue:
21 29 378 18
OK

Arranged By: Severity Descending

446 Security Issues (641 variants) for 'My Application'

- My Application (446)
  - http://aes.stanford.edu/ (178
    - / (4)
    - pci
    - cgi-bin (1)
    - icons (2)
    - pci (170)
    - ServerSpoofing
    - test (1)
  - https://aes.stanford.edu/ (26
    - / (2)
    - pci (266)
    - ServerSpoofing

- Cross-Site Scripting (2)
  - http://aes.stanford.edu/pci/BookComments.php (1)
    - id
  - https://aes.stanford.edu/pci/BookComments.php (1)
- Database Error Pattern Found (2)
- DOM Based Cross-Site Scripting (1)
- Parameter Value Overflow (1)
- Permanent Cookie Contains Sensitive Session Information (1)
- PHP Remote File Inclusion (9)
- Session Identifier Not Updated (1)
- Sun ONE / iPlanet Web Server Remote Buffer Overflow (1)
- Unix File Parameter Alteration (3)
- Cross-Site Request Forgery (20)
- Directory Listing (2)
- Phishing Through Frames (2)
- Phishing Through URL Redirection (4)
- Unencrypted Login Request (1)
- Alternate Version of File Detected (10)
- Application Test Script Detected (2)
- Cacheable SSL Page Found (12)
- Client-Side (JavaScript) Cookie References (117)
- Encryption Not Enforced (63)
- Hidden Directory Detected (2)
- HTML Comments Sensitive Information Disclosure (1)
- Internal IP Disclosure Pattern Found (117)
- Possible Server Path Disclosure Pattern Found (6)
- Query Parameter in SSL Request (32)

Scanner Test Site

Home Test Links Book Images
FeedBack UserImages Customized Pages Custo… …ages JSP
Server Spoofs PostMessage Vulnerability Open… …Info Leak
BlackHat XSS XPath Injection User Agent Spo… …SS Attacks
Login Logout

Sorry unable to display book withthe requested id

Comments:

Name:

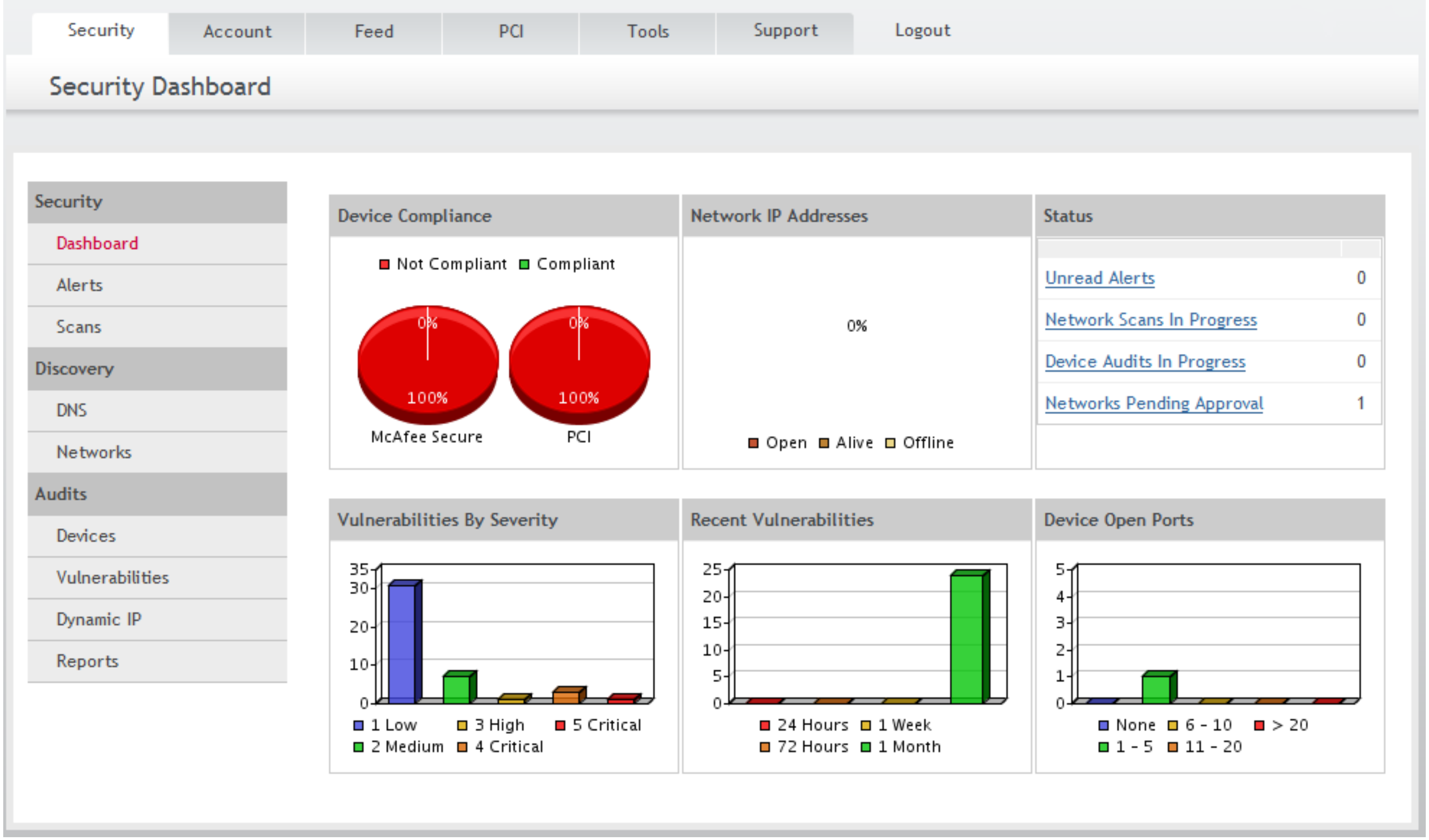
email:

40475

Simulation of the pop-up that will appear when this page is opened in a browser

Visited URLs 354/354 Completed Tests 39183/39183 446 Security Issues 21 29 378 18

Scanner 2



Unread Alerts	0
Network Scans In Progress	0
Device Audits In Progress	0
Networks Pending Approval	1

Goals of Study

- What vulnerabilities are tested by scanners?
- How representative are scanner tests of in-the-wild vulnerabilities?
- What can the user expect from scanner?
- What is difficult for the scanner to detect?

Non-Goals

- Not a product ranking
- Not a benchmark of particular tools

Outline

- Vulnerability categories tested by scanners
- How prevalent are these in the wild?
- Common application results
- Custom testbed design
- Custom testbed results
  - Coverage
  - Detection
  - False Positives

Survey of Leading Products

Local Installation

acunetix

IBM®

RAPID7

N-Stalker
THE WEB SECURITY SPECIALISTS

hp

CENZIC

Service

McAfee SECURE
TESTED 02-FEB

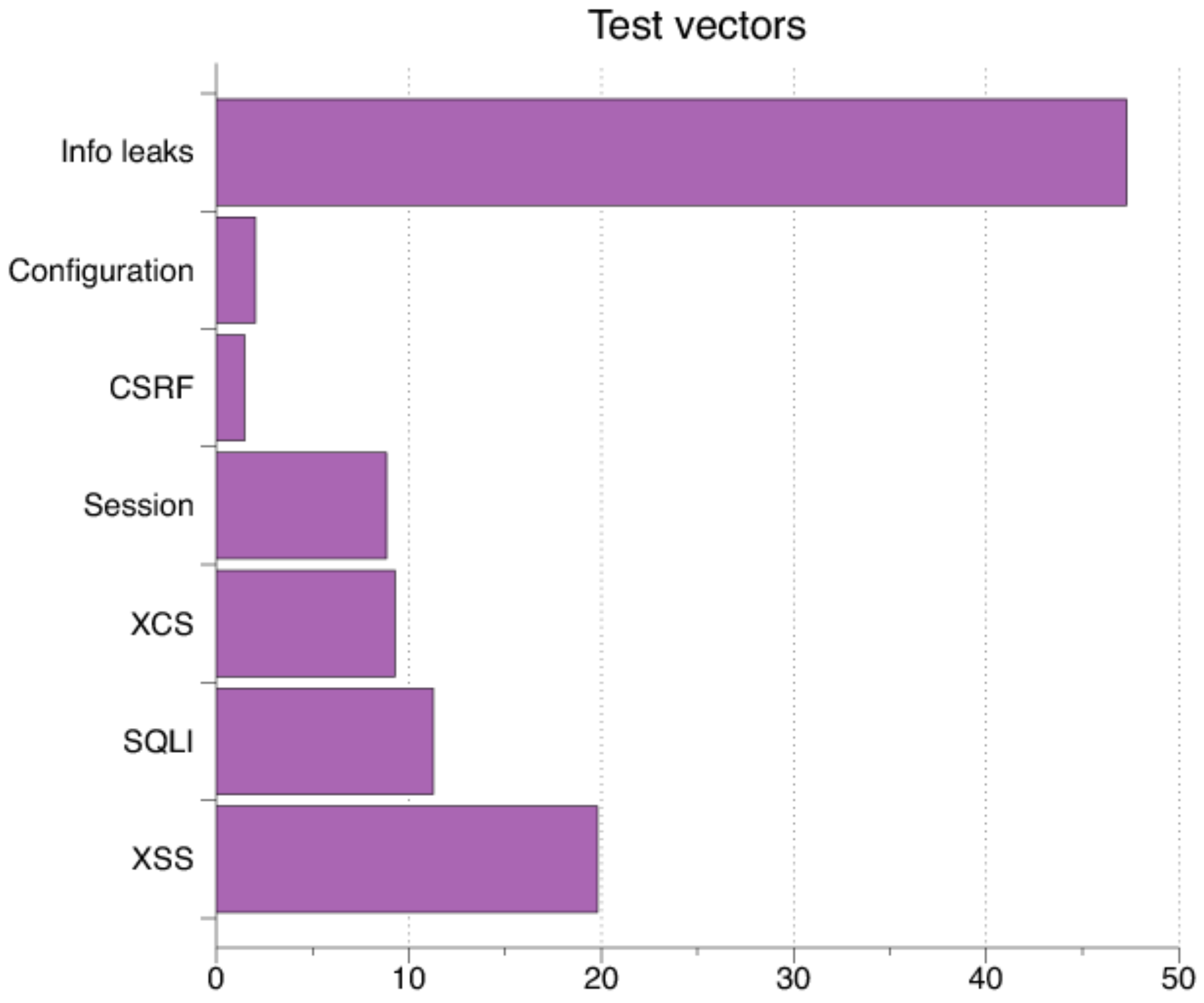
QUALYS®

>$100K total retail price

Vulnerability Categories From Scanners

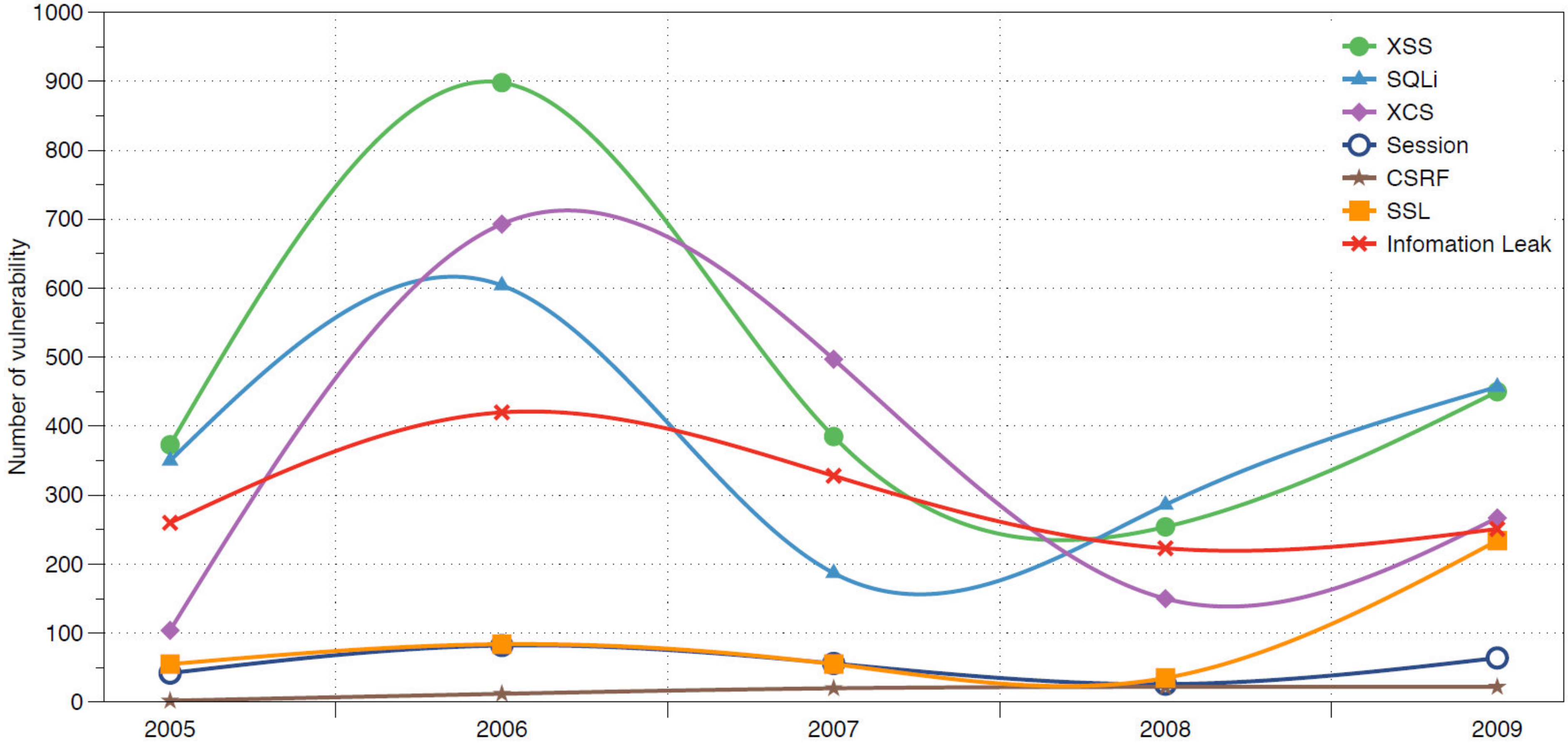
Category	Example Vulnerabilities
Cross Site Scripting	XSS
SQL Injection	SQLI
Cross Channel Scripting (Other forms of injection)	Arbitrary File Upload Remote File Inclusion OS command Injection
Session Management	Session Fixation and Prediction Authentication Bypass
Cross-Site Request Forgery	CSRF
SSL/Server Config	Self-Signed Cert, HTTP Trace
Info Leakage	Temp file access, path traversal Error message disclosure

Test Vectors By Category



Test Vector Percentage Distribution

Reported Vulnerabilities "In the Wild"



Data from VUPEN

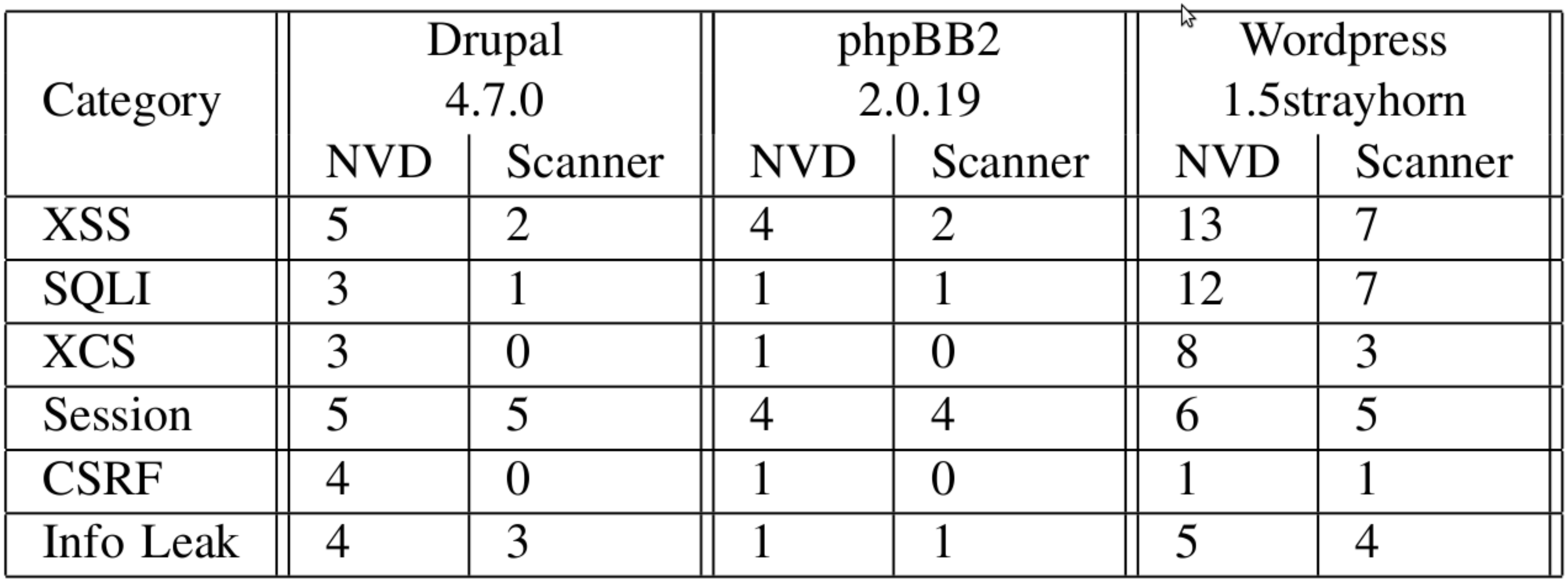
Scanners vs. In-the-Wild

- Top 4 for both:
  - XSS
  - SQLI
  - XCS
  - Info Leak

- Scanners have many more info leak vectors
  - Easier to write?

Detecting Known Vulnerabilities

Vulnerabilities for
previous versions of Drupal, phpBB2, and WordPress



Category	Drupal 4.7.0		phpBB2 2.0.19		Wordpress 1.5strayhorn	
	NVD	Scanner	NVD	Scanner	NVD	Scanner
XSS	5	2	4	2	13	7
SQLI	3	1	1	1	12	7
XCS	3	0	1	0	8	3
Session	5	5	4	4	6	5
CSRF	4	0	1	0	1	1
Info Leak	4	3	1	1	5	4

Good: Info leak, Session (Anecdote from re-test)
Decent: XSS/SQLI
Poor: XCS, CSRF (low vector count?)

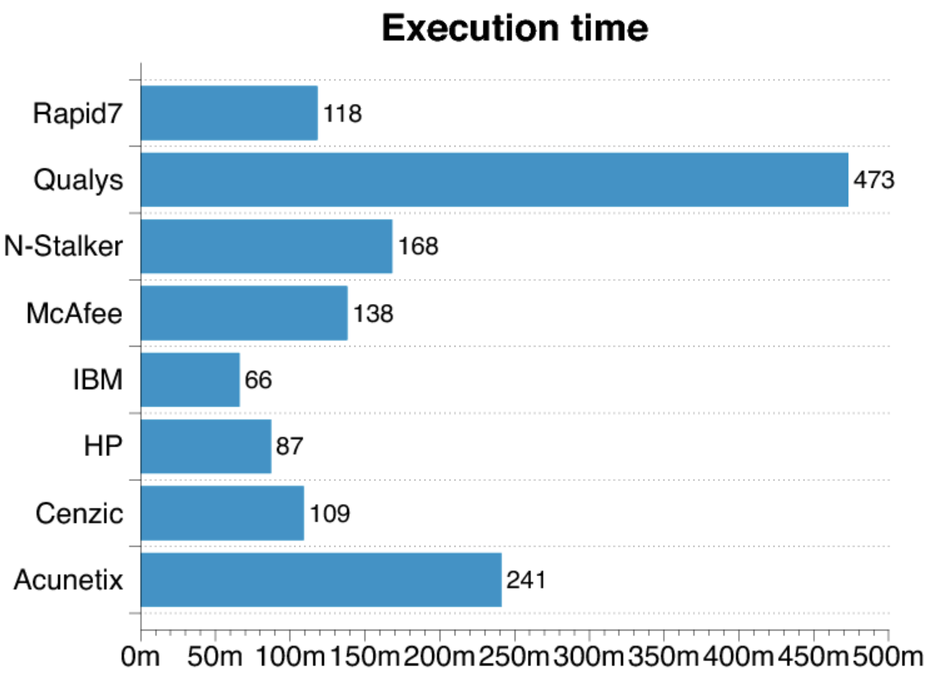
Custom Testbed for Scanners

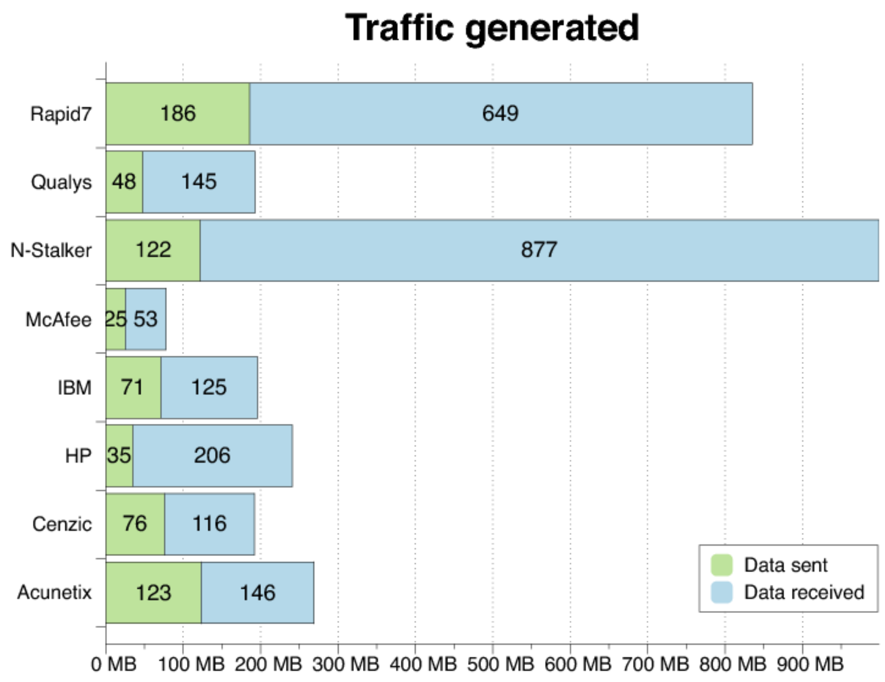
- Vulnerabilities covering
  - OWASP Top 10
  - WASC Web Security Threat Classifications
- NIST and WASC scanner selection criteria
  - Test all of NIST recommendations
  - Test 37 of 41 capabilities listed by WASC

Our Custom Testbed

- Linux + Apache + MySQL + PHP (LAMP)
- Measure Performance
    - Test Duration / Network Traffic
- Measure Coverage
    - Links coded in various technologies (Flash, SilverLight, ...)
    - Can scanner follow link?
- Measure Vulnerability Detection Rate
    - XSS (Type 1, Type 2, Advanced)
    - SQLI (Type 1, Type 2)
    - Cross Channel Scripting
    - CSRF
    - Session Management
    - Server/Crypto Config
    - Information Leak
    - Malware

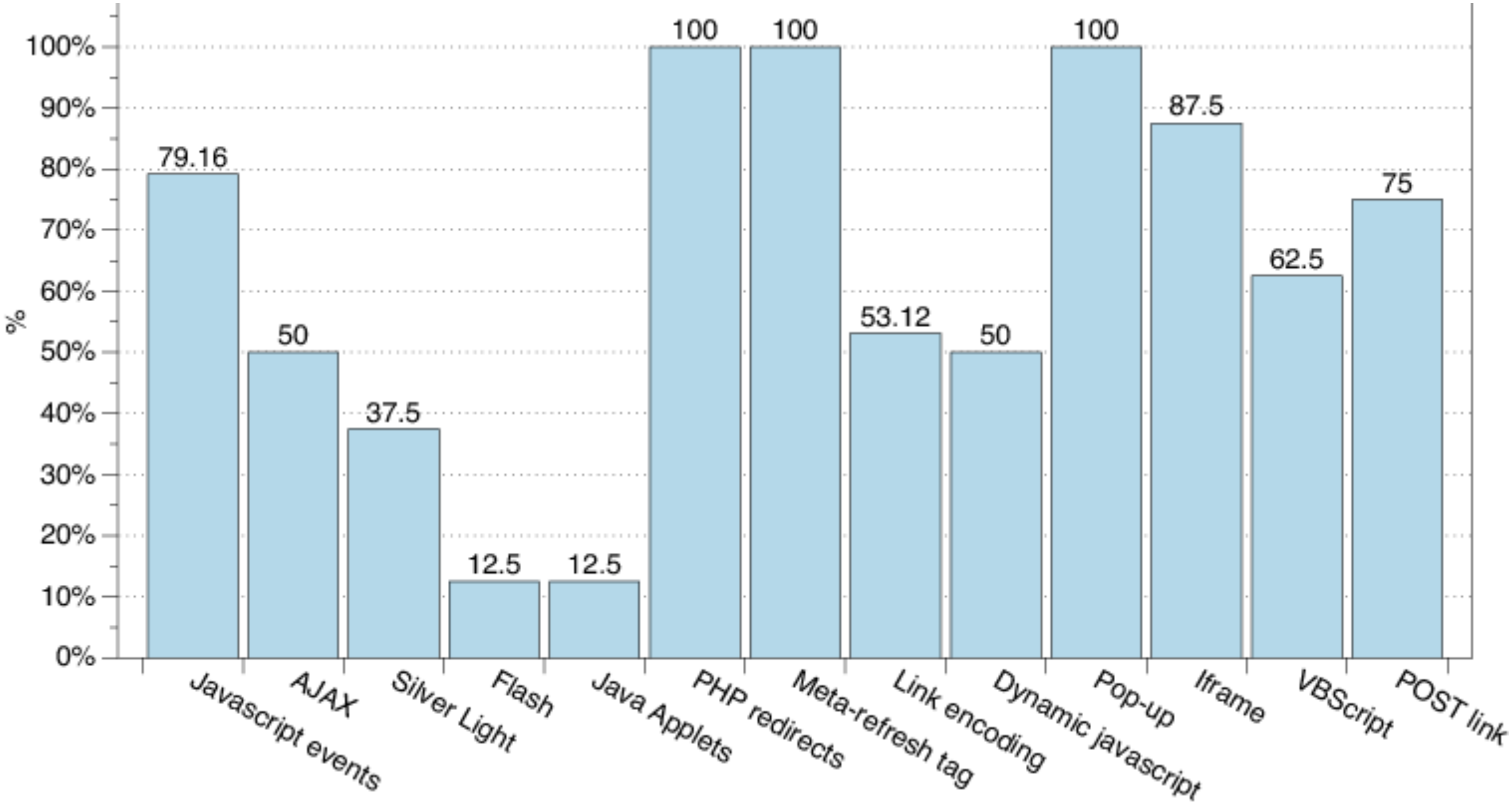
Scanner Performance





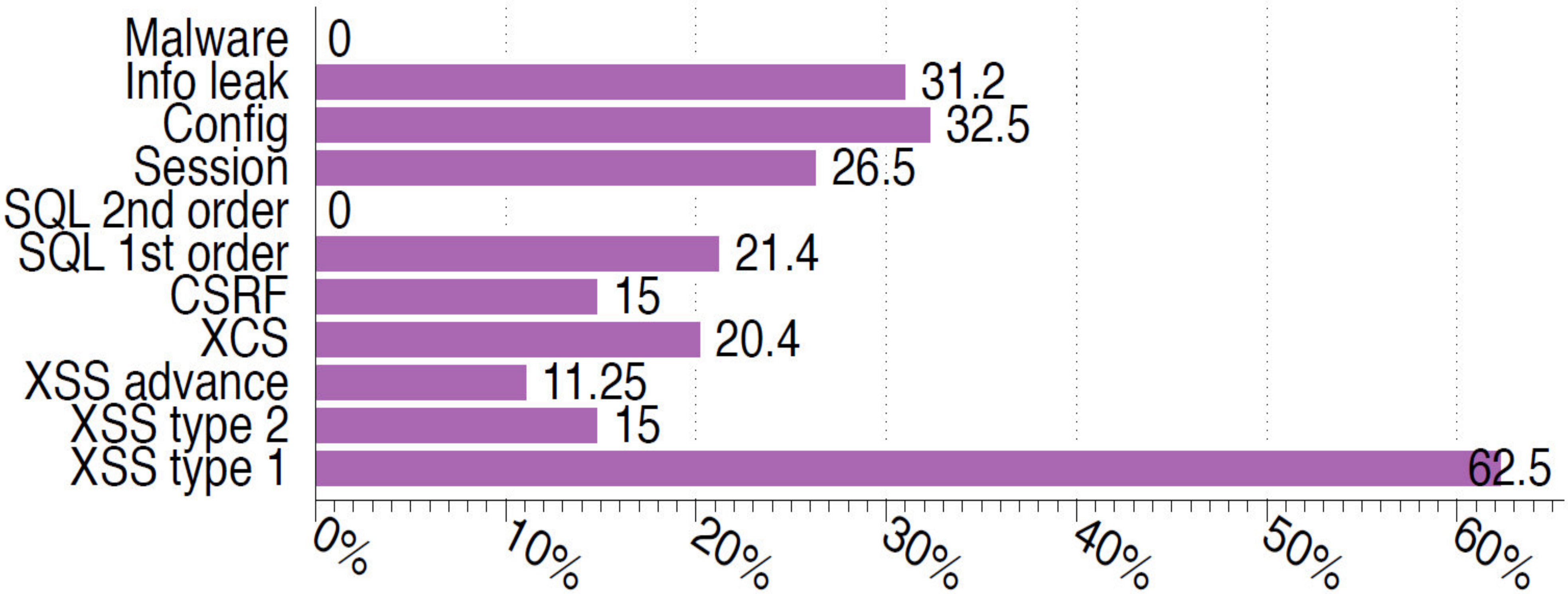
Performance did not correlate well with vulnerability detection

Scanner Page Coverage



% Successful Link Traversals By Technology, Averaged over all Scanners

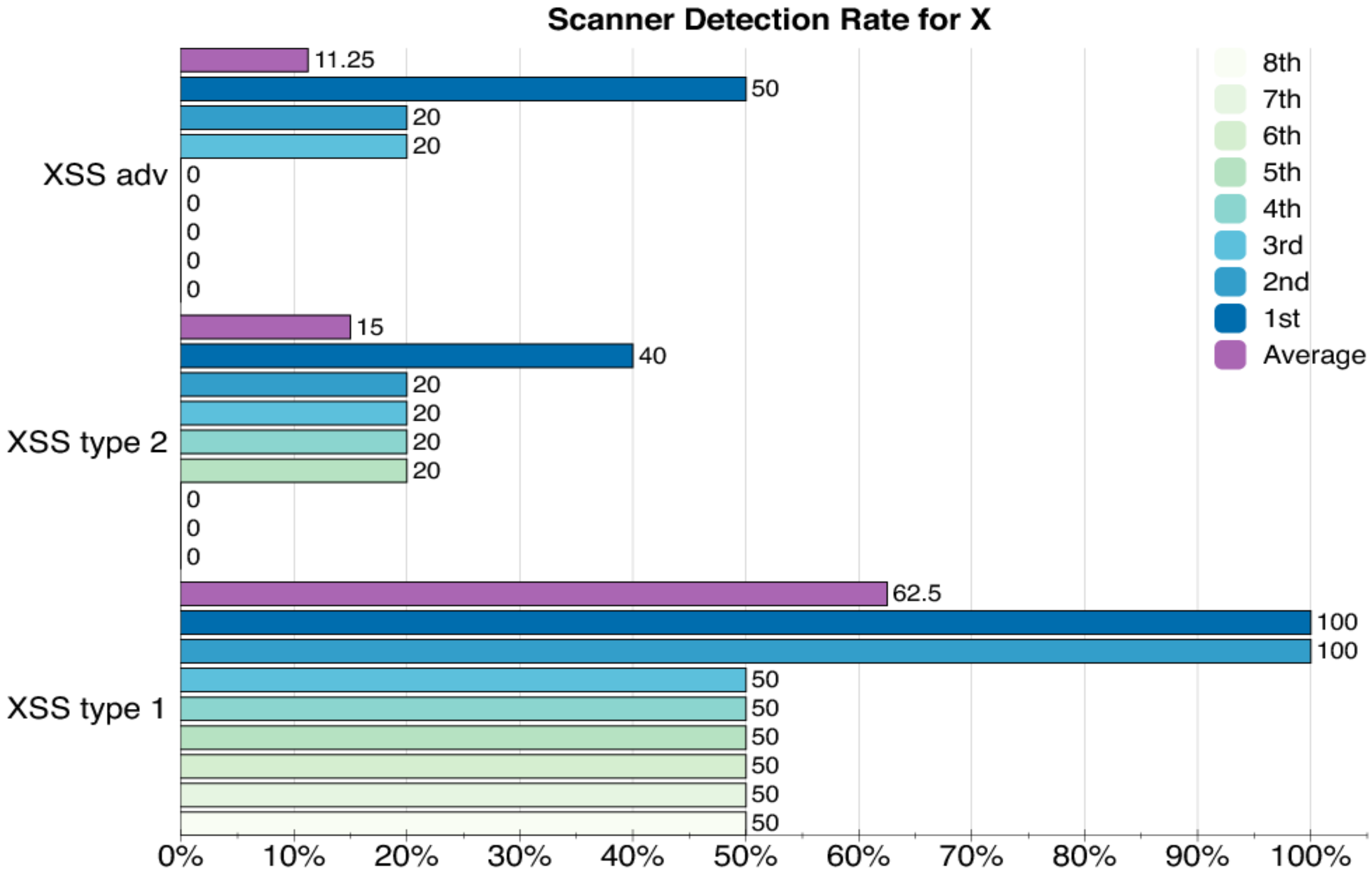
Vulnerability Detection Rate



XSS Testbed

- Type 1: Textbook "Reflected" Vulnerability
  - User input → page w/o sanitization
- Type 2: Textbook Stored Vulnerability
  - User input → DB → Served Page
  - Some viewable only by different user
- Advanced (all reflected)
  - Novel Tags: e.g. <object>, <prompt>
  - Novel Channels:
    - URL → $_SERVER['HTTP_SELF']
    - Filename → error msg

XSS Results



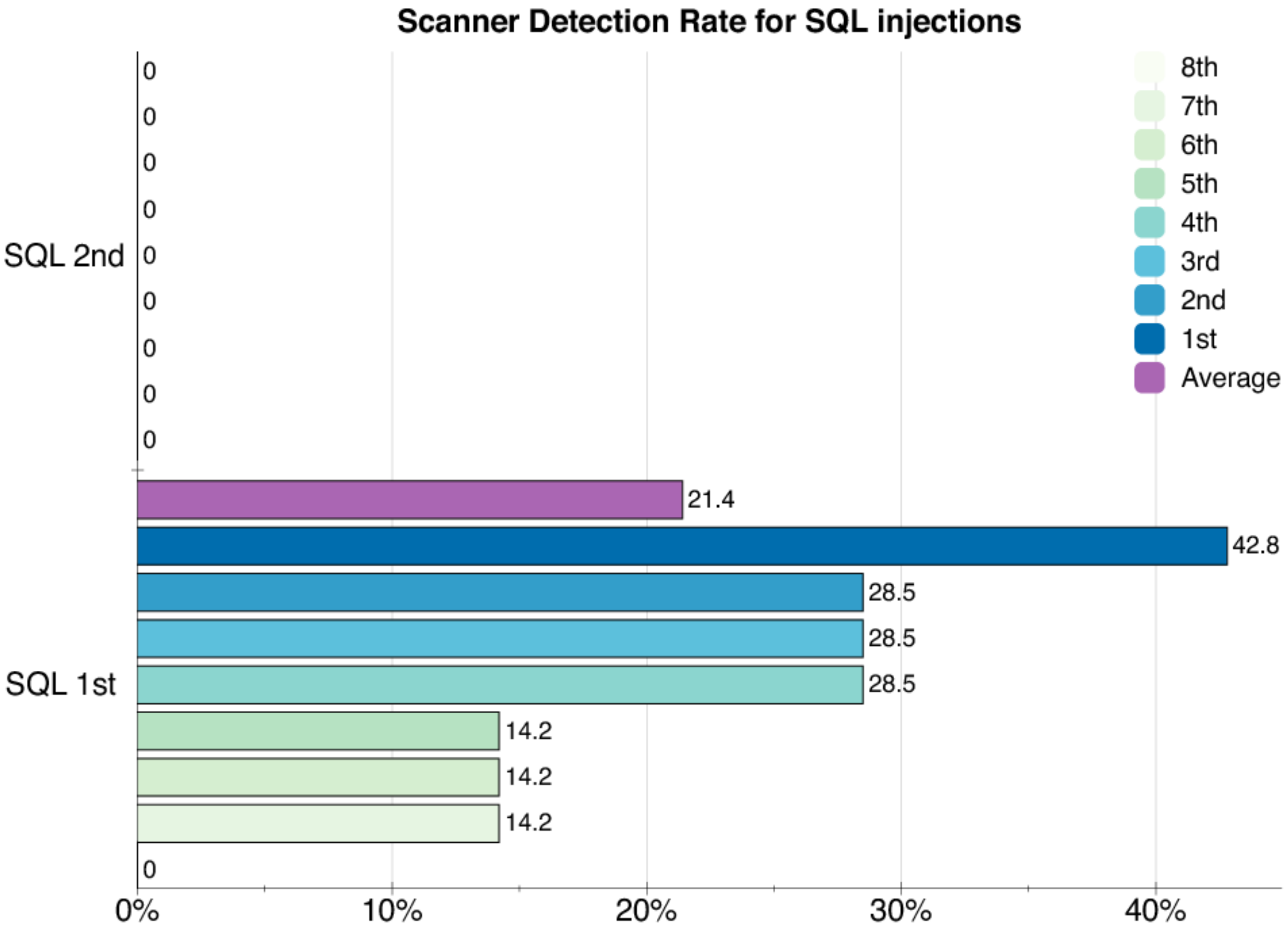
Anecdote about Type 2 "alert"

SQLI Testbed

- Type 1: User input → SQLI on page generation
  - Basic: ‘ ; --
  - Advanced: “, LIKE, UNION

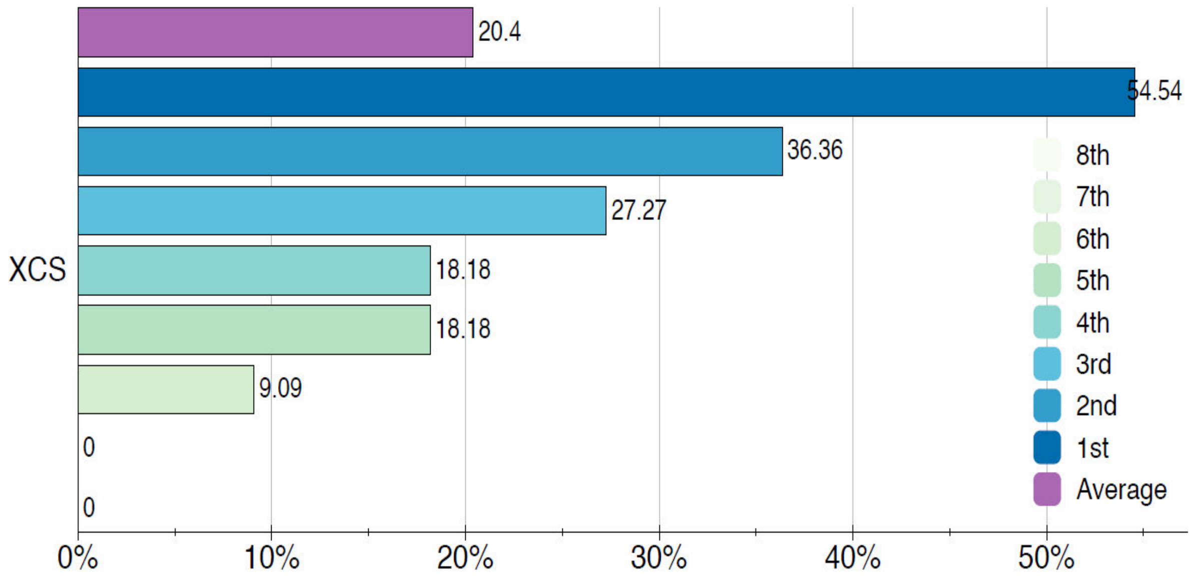
- Type 2: Input → DB → SQL Query
  - Only basic cases
  - Unsanitized form input (username) → DB
    - Later used in SQL query

SQLI Results



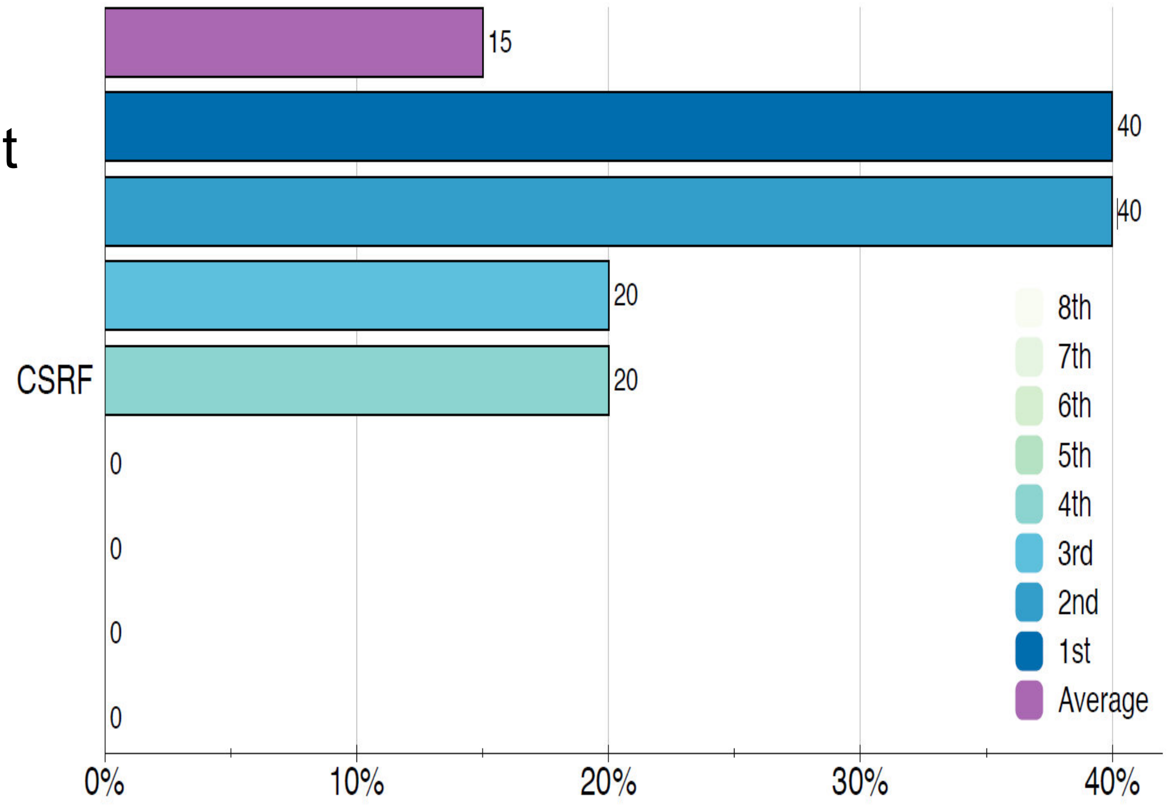
XCS Results

- “Other forms of Injection” by attacker
- Manipulates server or client browser
- Tests:
  - XPATH injection
  - Malicious File Upload
  - Cross-Frame Scripting
  - File Includes
  - Open Redirects
  - Header Injection
  - Flash Parameter
  - SMTP Injection



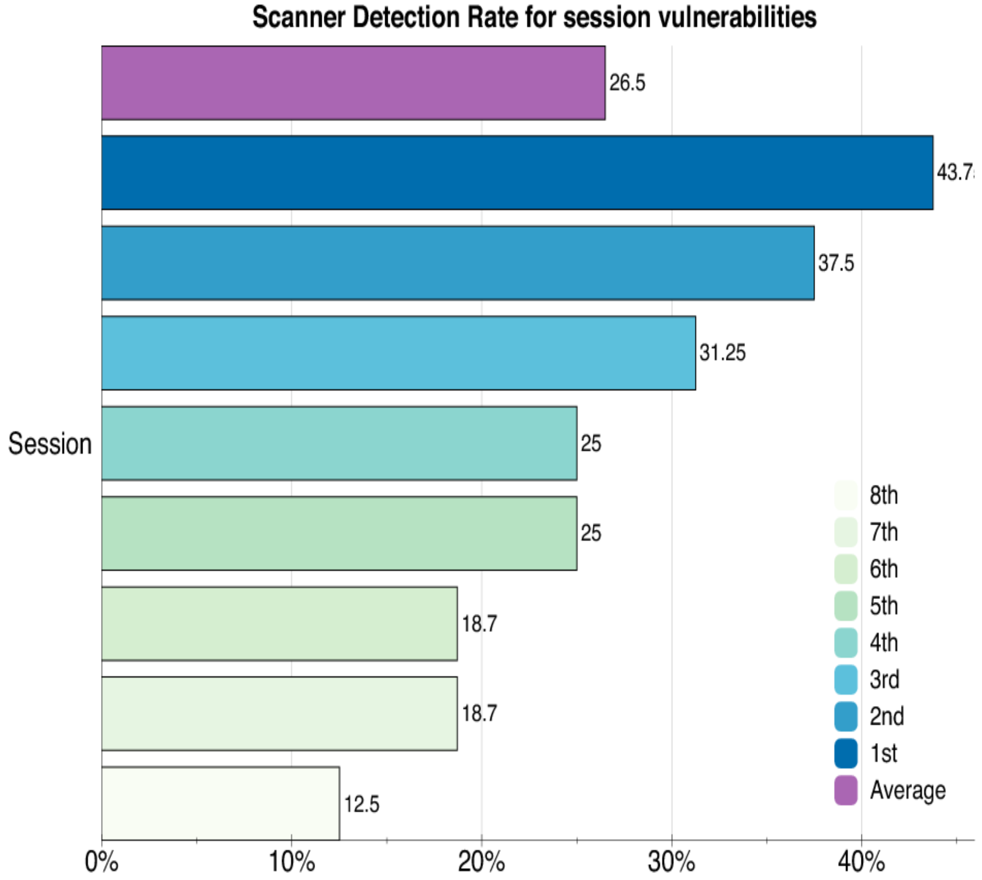
CSRF Results

- Post-login forms
  - w/o hidden random token
  - with weak [0,9] token
  - with same token each time
- JSON Hijacking
  - Sensitive AJAX request
  - No session id sent
- Anecdote:
  Not checked on purpose



Session Management

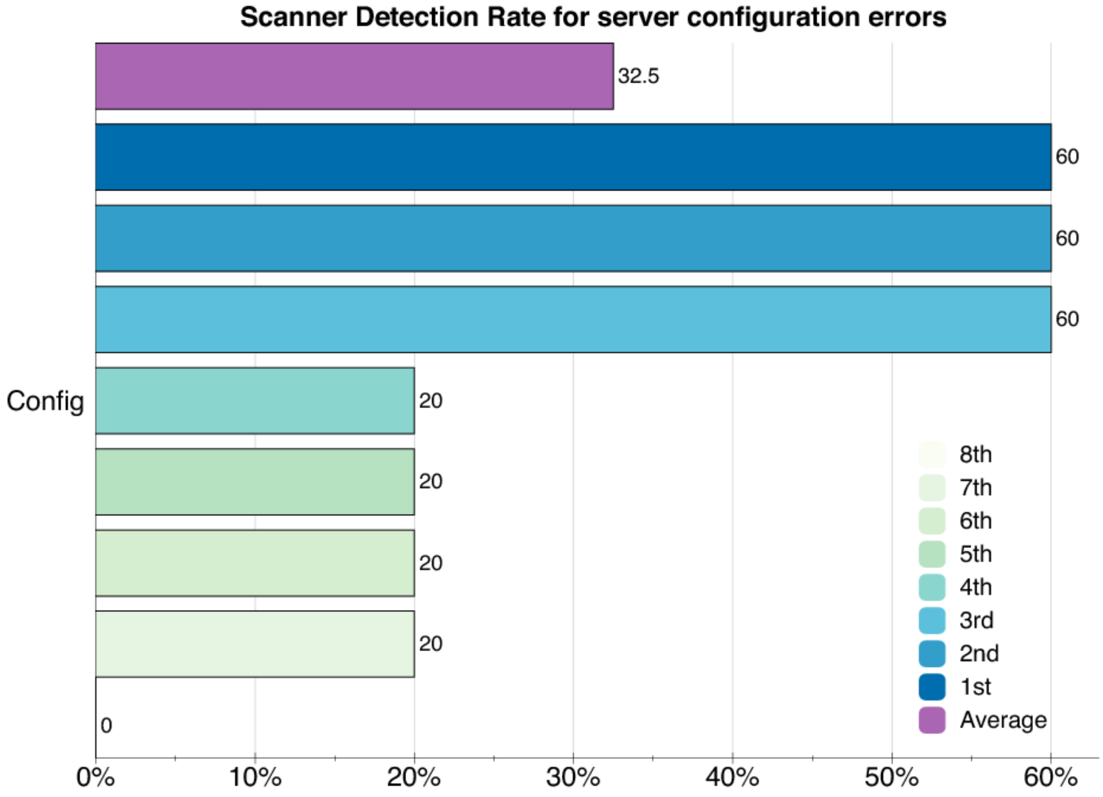
- Login / form errors
  - Login form not https
  - Reg. credentials in clear
  - Autocomplete pwd field
  - Weak pwds and pwd recovery question
  - Weak reg. page CAPTCHA
- Cookie errors
  - Not HttpOnly
  - Auth tokens not https
  - Persistent Auth token value MD5 (pwd)
  - Logout fails to clear cookie
  - Path restriction to '/'



Server/Crypto Mis-Config

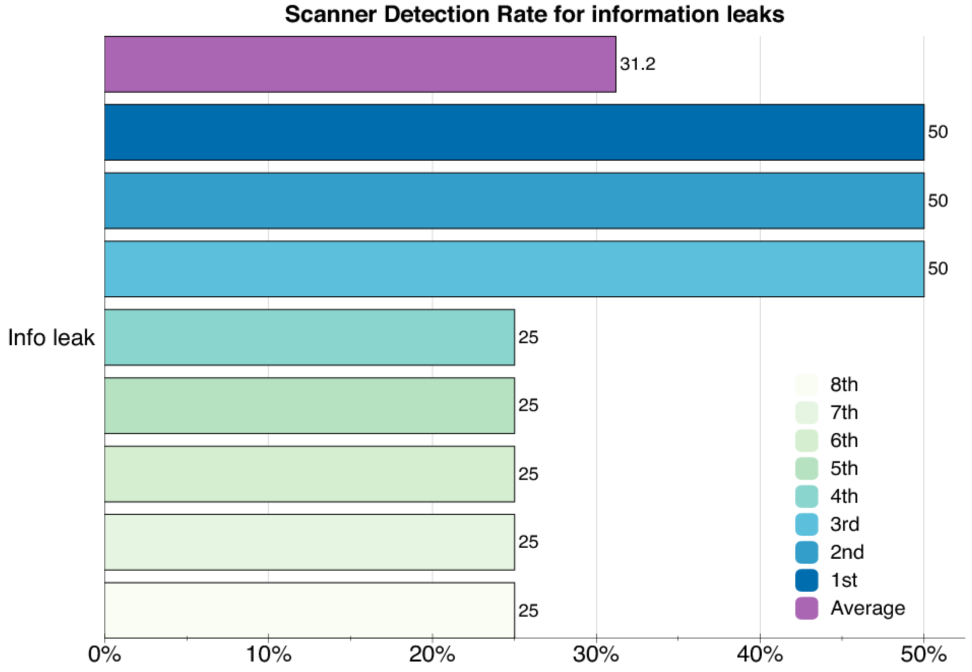
- Server Mis-Config:
  - HTTP Trace enabled
  - PHP settings allowing code includes
  - PHP img parsed as code

- Crypto Mis-Config
  - Self Signed Cert
  - Weak SSL Cipher



Info Leak

- SQL error message
- Username existence
- Backup files
- Comment/Path Disclosure
- Path Traversal
    - Inclusion of /etc/secret.txt



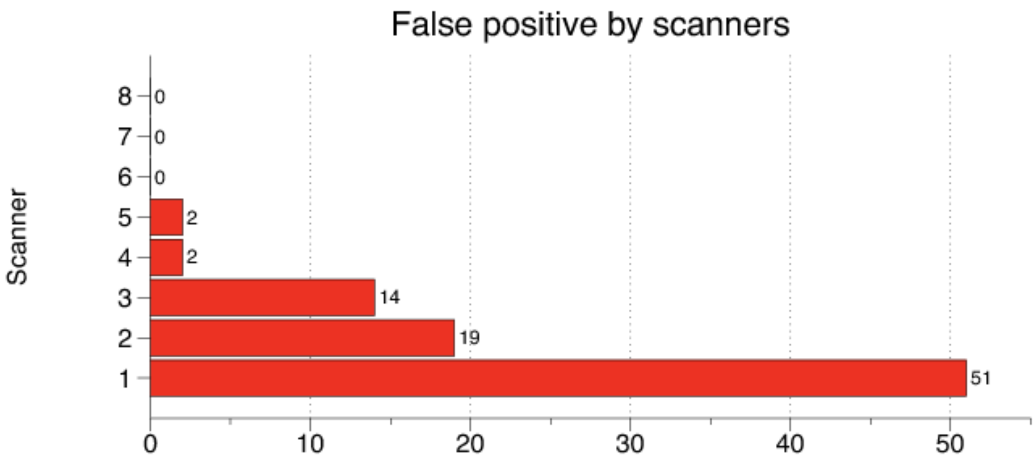
Malware Presence

- JavaScript key-logger on login page
- Malicious graphic uploaded by user
  - Directly reference-able
- No Scanner Detected
  - Because not part of PCI compliance?

False Positives

- Testbed Traps
  - alert()s as site behavior (not part of injection)
    - Scanners avoided
  - Benign (comment) region within <script> tags
    - Tripped 2 scanners (reported 1 and 13 times)

- On a testbed of ~90 confirmed vulnerabilities



- Low FP rates due to high vulnerability density in testbed?

False Positive Observations

- Scanners exist in all these categories:
  - High Detection Rate, Low False Positive Rate
  - Low Detection Rate, High False Positive Rate
  - Low Detection Rate, Low False Positive Rate

- False positive rate not indicative of detection rate

Conclusions

- No scanner was top 3 performer across all categories
- Scanners relatively good at detecting
  - Historical vulnerabilities
  - Textbook XSS and SQLI
  - Info Leak, Session, and Server/Crypto Mis-config
    - Easier test vectors to write/interpret
- Can improve
  - Understanding of active content such as Flash, SL
  - CSRF, Malware, XCS
    - Low test vector count → Not vendor focus?
  - Advanced (novel) forms of XSS, SQLI
    - Faster reactive process
  - Stored forms of XSS, SQLI (acknowledged by a CTO)
    - Better DB modeling

Thank You

Jason Bau
jbau@stanford.edu