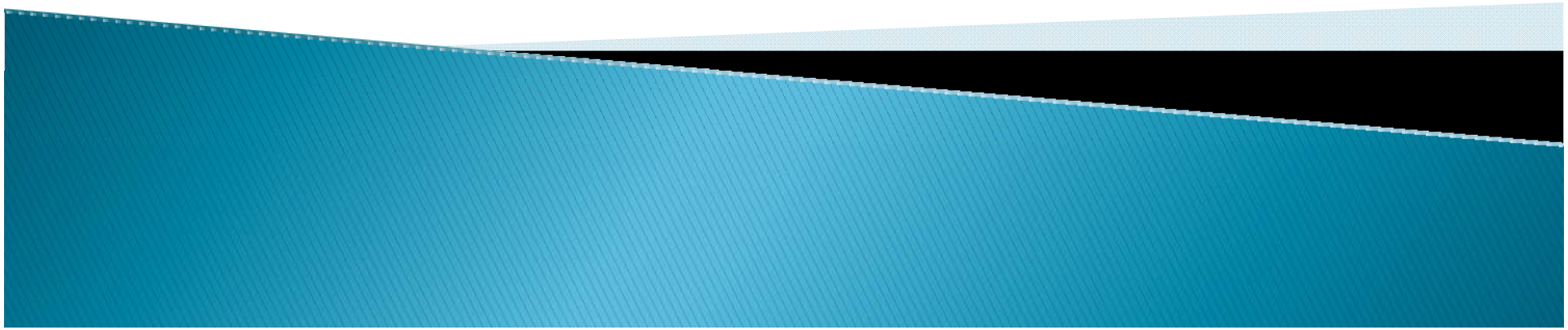
Why Johnny Can't Pentest: An Analysis of Black-box Web Vulnerability Scanners

Adam Doupé, Marco Cova and Giovanni Vigna

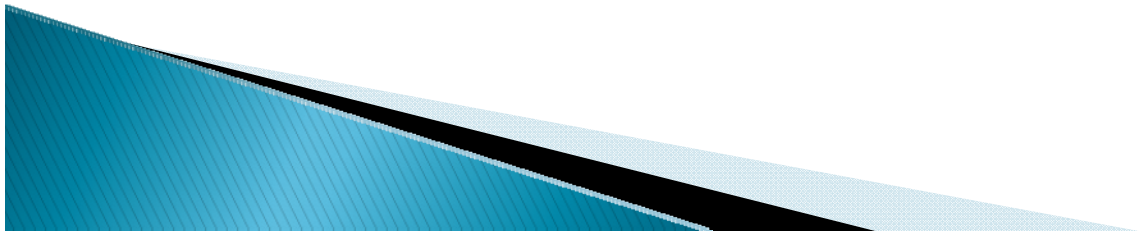
University of California, Santa Barbara

DIMVA 2010 - 7/8/10



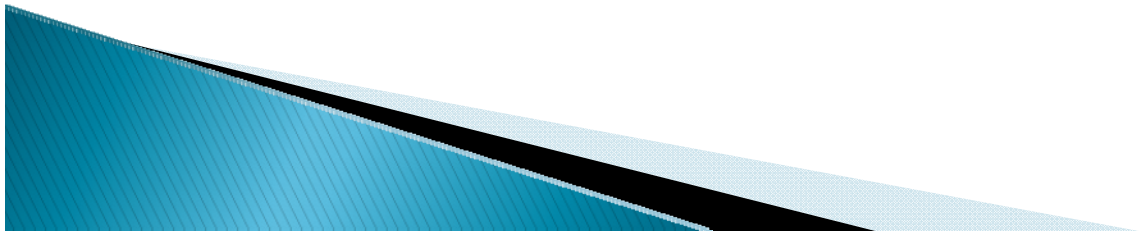
Outline

- Introduction to black box web vulnerability scanners
- Design of custom vulnerable website – WackoPicko
- Results
- Analysis

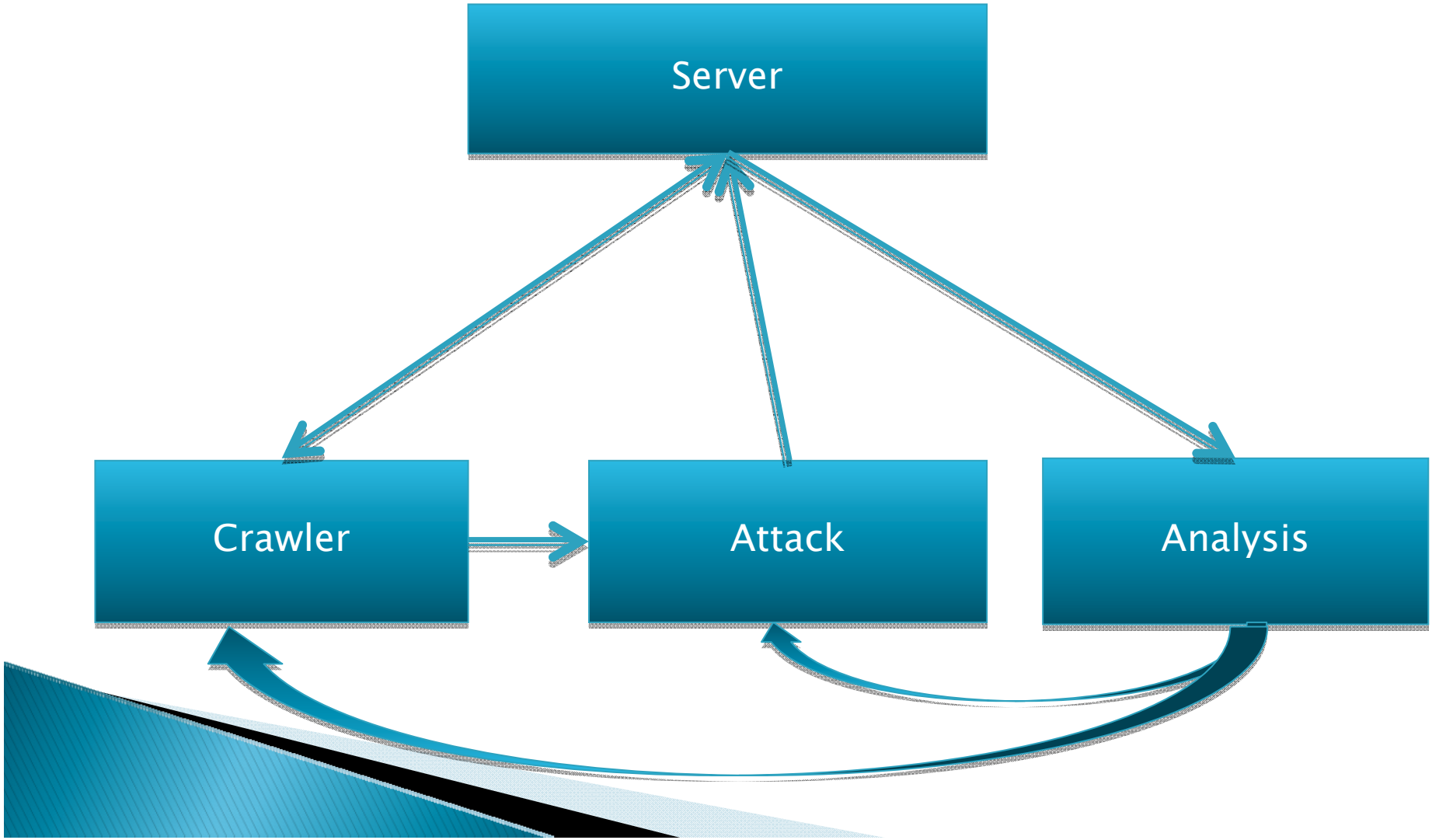


Contributions

- Describe the design of a testing web application
- Identify a number of challenges that scanners need to overcome when testing modern web applications
- Test the performance of eleven real-world scanners and identify areas that need further work

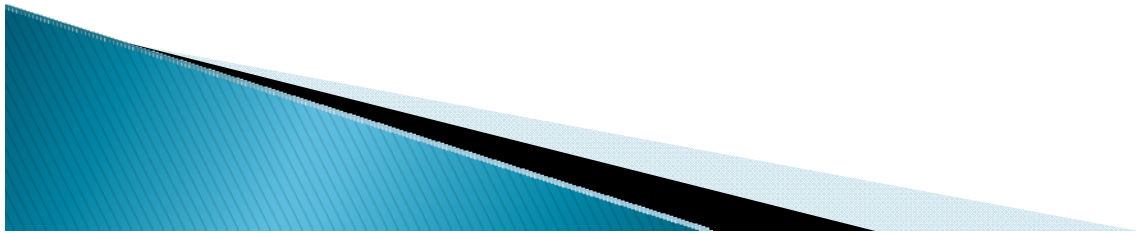


Web Application Vulnerability Scanners



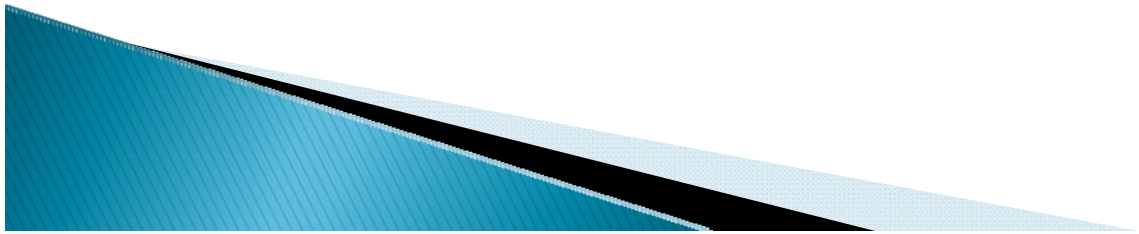
Vulnerable Web Application – WackoPicko: Design

- Authentication
- Upload Pictures
- Comment on Pictures
- “Purchase” Pictures
- Tag Search
- Guestbook
- Admin Area



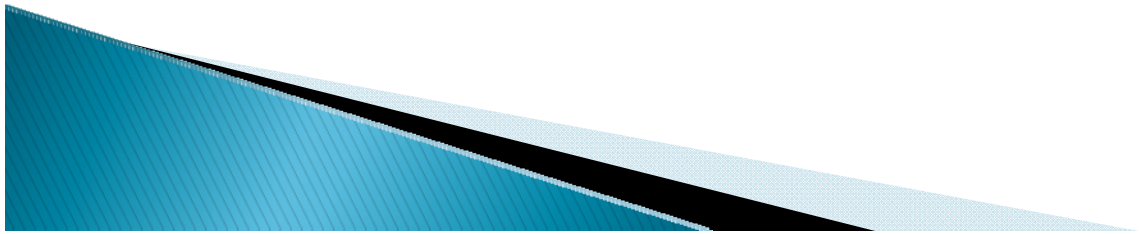
Vulnerable Web Application – WackoPicko: Publicly Accessible

- XSS
  - Reflected, Stored, and Reflected behind JavaScript
- Session ID
- Weak Password
- Reflected SQL Injection
- Command Line Injection
- File Inclusion
- File Exposure
- Parameter Manipulation



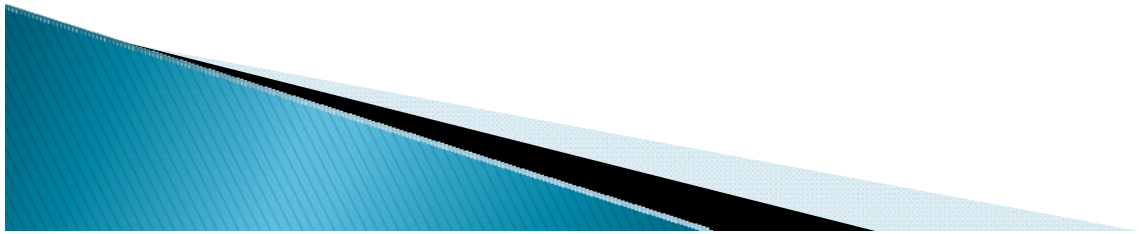
Vulnerable Web Application – WackoPicko: Authentication

- Reflected XSS behind Flash
- Stored SQL Injection
- Directory Traversal
- Multi-step Stored XSS
- Forceful Browsing
- Logic Flaw



Crawling Challenges

- HTML Parsing
- Multi-Step Process / State
- Infinite Website
- Authentication
- Client-side Code
  - Web Input Vector Extractor Teaser (WIVET)

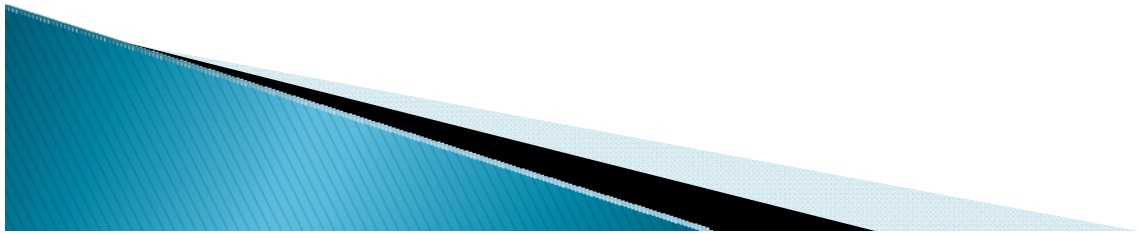


Scanners

Name	Price
Acunetix	$4,995 – $6,350
AppScan	$12,550 – $32,500
Burp	£125 ($190.82)
Grendel-Scan	Open source
Hailstorm	$10,000
Milescan	$495 – $1,495
N-Stalker	$899 – $6,299
NTOSpider	$10,000
Paros	Open source
w3af	Open source
Webinspect	$6,000 – $30,000

Experiment

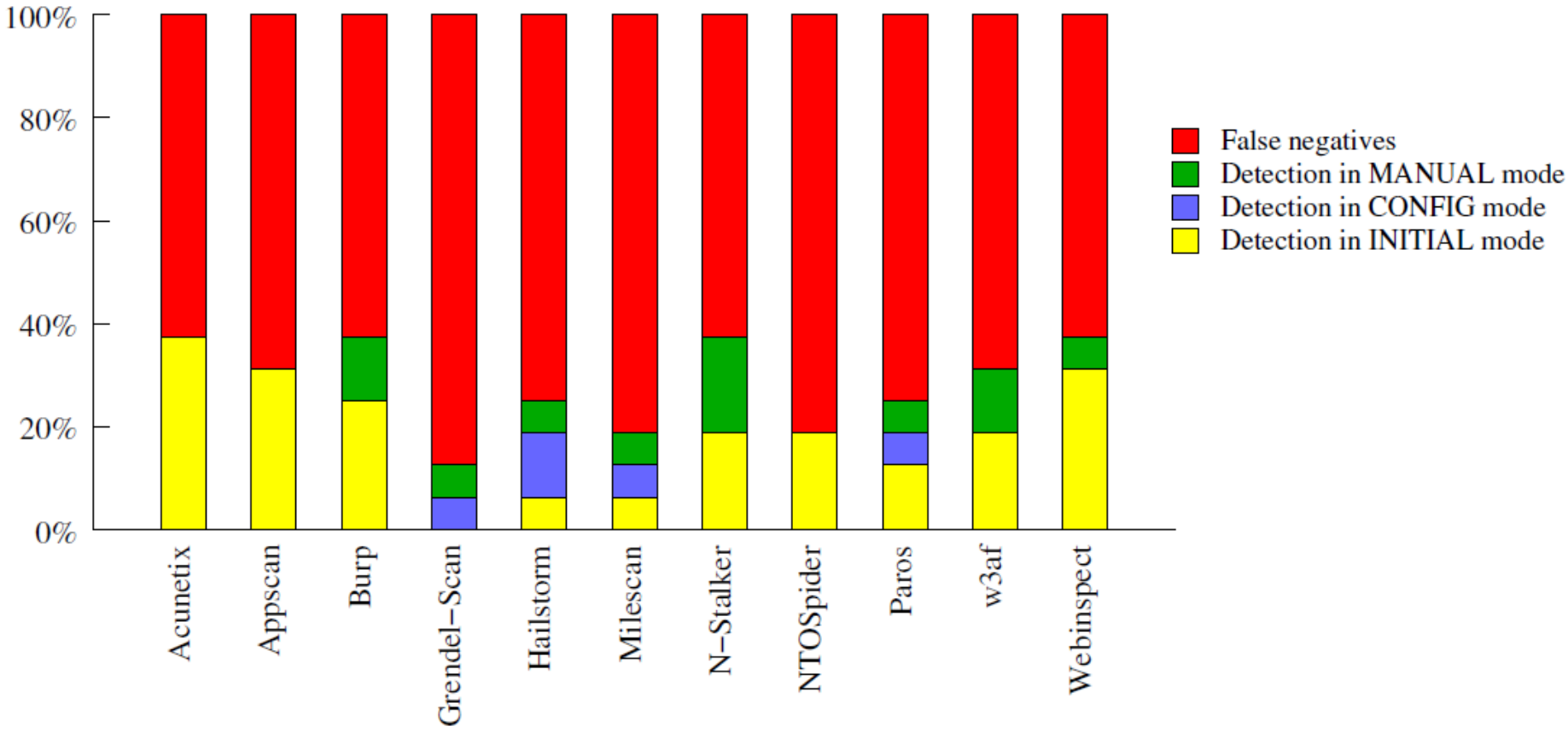
- Each scanner run four times:
  - WackoPicko
    - Initial – No configuration (point and click)
    - Config – Given valid Username/Password
    - Manual – Used proxy to thoroughly browse site.
  - WIVET – Testing JavaScript capabilities
- Limitations



Results

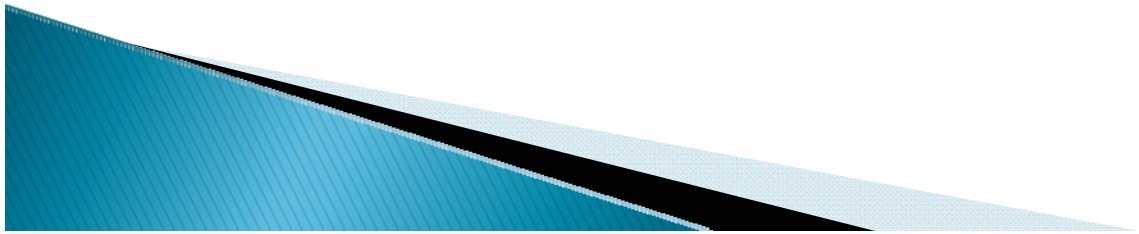
Name	Reflected XSS	Stored XSS	SQL Injection	Command line Injection	File Inclusion	File Exposure	XSS via JavaScript	XSS via Flash
Acunetix	Initial	Initial	Initial		Initial	Initial	Initial	
AppScan	Initial	Initial	Initial		Initial	Initial		
Burp	Initial	Manual	Initial	Initial		Initial		Manual
Grendel-Scan	Manual		Config					
Hailstorm	Initial	Config	Config					Manual
Milescan	Initial	Manual	Config					
N-Stalker	Initial	Manual	Manual			Initial	Initial	Manual
NTOSpider	Initial	Initial	Initial					
Paros	Initial	Initial	Config					Manual
w3af	Initial	Manual	Initial		Initial			Manual
Webinspect	Initial	Initial	Initial		Initial		Initial	Manual

Results


100%
80%
60%
40%
20%
0%
Acunetix
Appscan
Burp
Grendel−Scan
Hailstorm
Milescan
N−Stalker
NTOSpider
Paros
w3af
Webinspect
False negatives
Detection in MANUAL mode
Detection in CONFIG mode
Detection in INITIAL mode

Missed Vulnerabilities

- Missed by all scanners
  - Session ID
  - Weak Password
  - Parameter Manipulation
  - Forceful Browsing
  - Logic Flaw
- Will discuss later
  - Stored SQL Injection
  - Directory Traversal
  - Stored XSS Behind Login

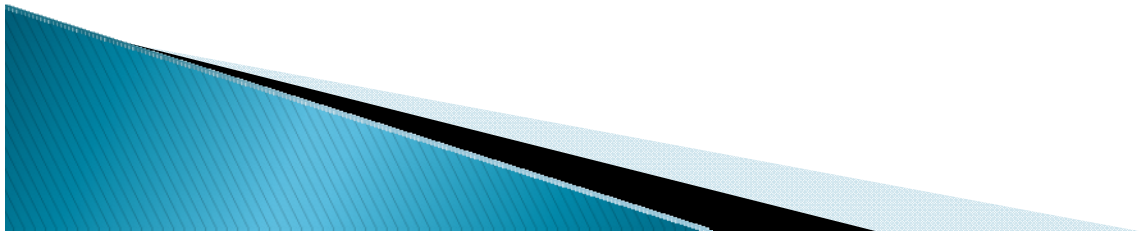


False Positives

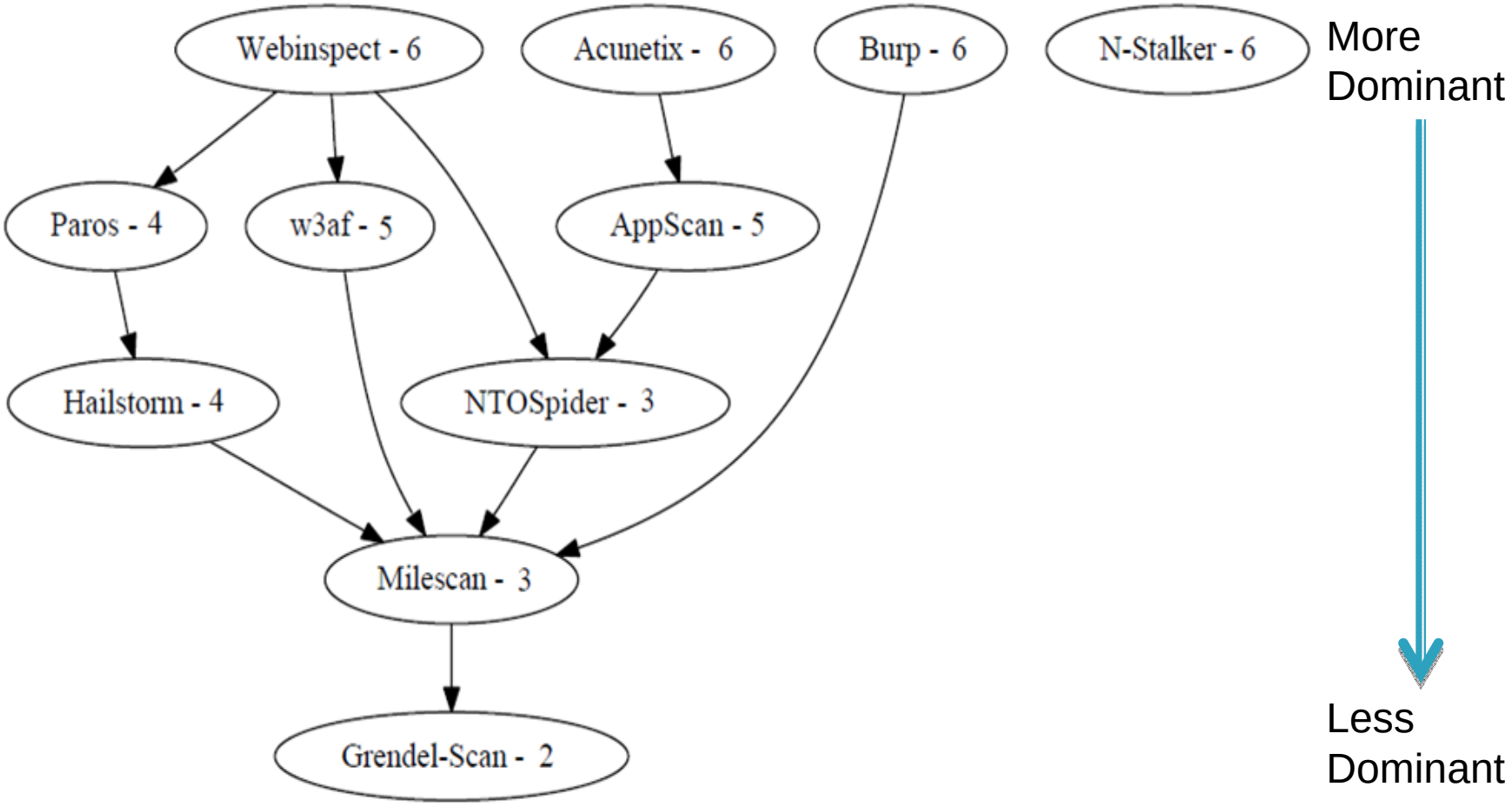
- Ranged from 0 to 200+
  - Average was ~25
- Why?
  - Server Path Disclosure
- “Actual” False Positives
  - Hailstorm
    - XSS, 2 Code Injection
  - NTOSpider
    - 3 XSS
  - w3af
    - PHP eval() Injection

Measuring and Comparing Detection Capabilities

- Strictly Dominates

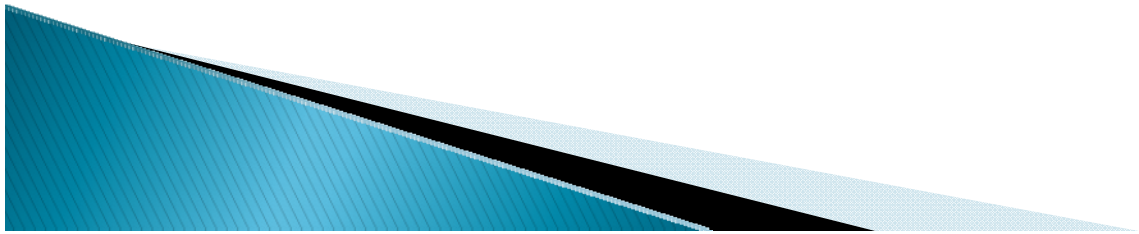


Dominates Graph



Attack and Analysis Capabilities

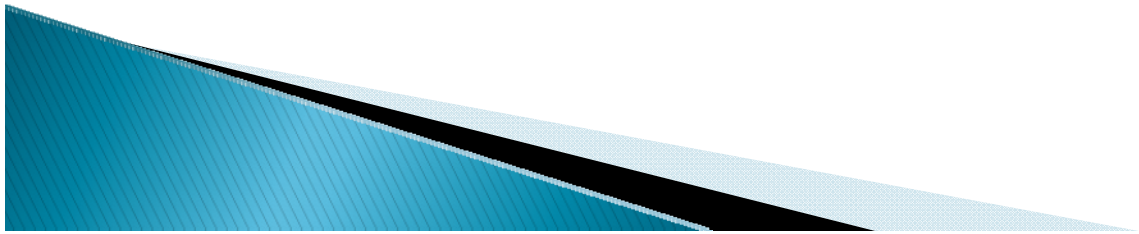
- Default values
- XSS attacks
- Command-line Injection
- SQL Injection
- File Exposure
- Remote Code Execution



Crawling Capabilities

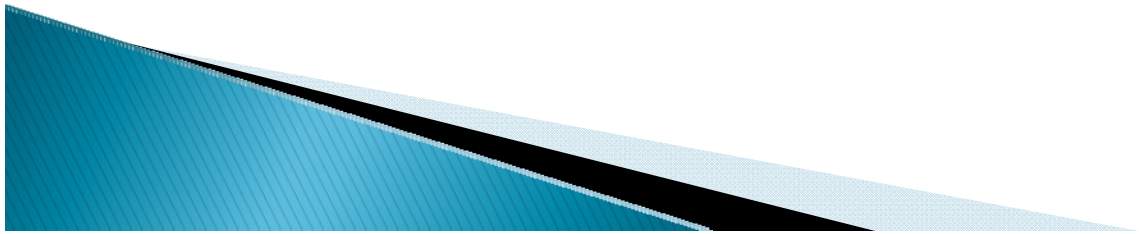
- Number of Accesses
  - Range from ~50 per page to ~3,000 per page
  - Hailstorm accessed vulnerable pages that required an account on INITIAL scan!

- HTML
  - Burp and N-Stalker
    - <TEXTAREA>
  - Milescan and Grendel-Scan
    - POST
  - Hailstorm
    - No-Injection
  - w3af
    - No Default



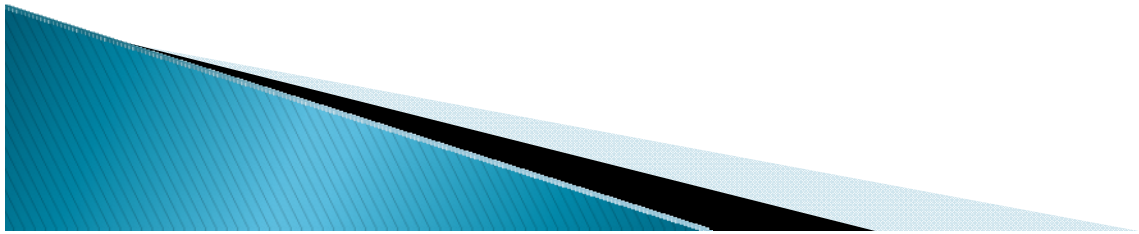
Crawling Capabilities

- Uploading a Picture
  - 2 Scanners uploaded without help
  - 3 Scanners unable to upload one!



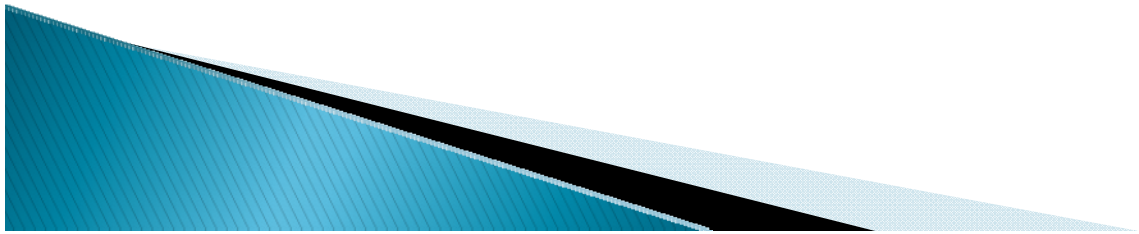
Crawling Capabilities – Client-side Code

- WIVET
  - 3 Scanners couldn't complete
    - Paros and Burp – <base>
    - N-Stalker – Frame?
  - Dynamic JavaScript
    - Webinspect, Acunetix, NTOSpider, Hailstorm
  - JavaScript library
  - No Flash



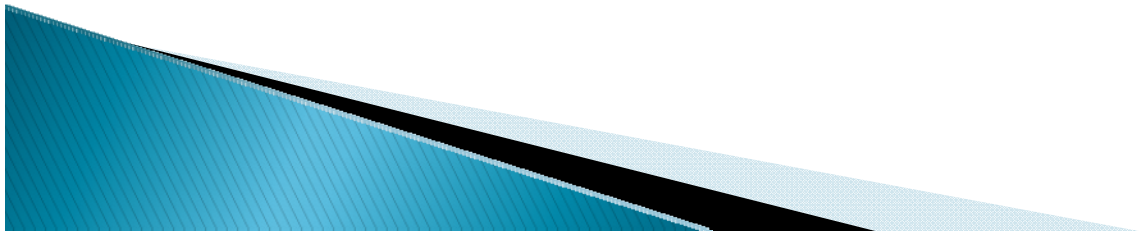
Crawling Capabilities – Authentication

- Created an account successfully
  - 4 Scanners
    - Hailstorm
    - N-Stalker
    - NTOSpider
    - WebInspect



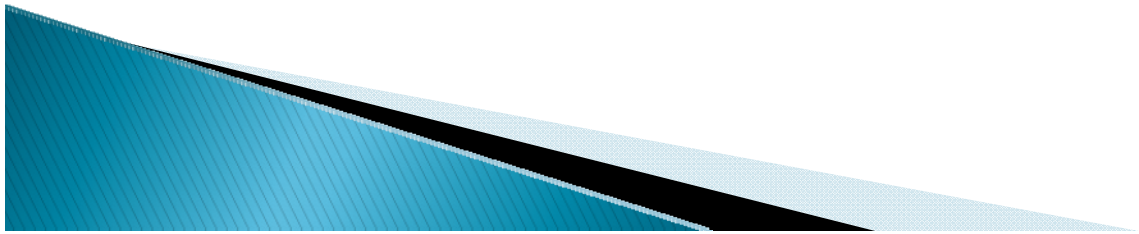
Lessons Learned: Want to make your own benchmark?

- Incorporate lots of logging in the application
- Two versions of the site
  - No vulnerabilities
  - All vulnerabilities
- Script running the tests
- Include:
  - File upload forms
  - AJAX
  - Several JavaScript UI Libraries

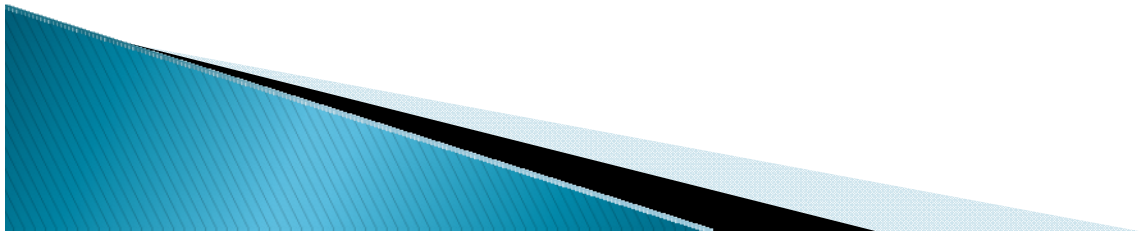


Conclusions

- Ability to crawl as important as detection
- Many vulnerabilities cannot be detected
- Cost not directly proportional to functionality



Questions?



Thanks!