AppSensor

Colin Watson
colin.watson(at)owasp.org

OWASP Training
Dublin

11th March 2011

3. AppSensor project

Category: Protection

Type: Documentation (& Tool)

Status: Beta

A framework for detecting and responding to attacks from within the application – application layer intrusion detection and prevention

Background

- Established Summer 2008
- AppSensor book, developer guide and planning workbook
- Presented at multiple conferences
- Team:
  - Michael Coates
  - John Melton
  - Colin Watson
- OWASP Live CD & OWASP Broken Web Apps

Resources

- Source in Google Code, demo WAR

  http://code.google.com/p/appsensor/

- Recent video presentations by Michael Coates
  - Real Time Application Defenses - The Reality of AppSensor & ESAPI
    http://vimeo.com/15726323
  - Automated Application Defenses to Thwart Advanced Attackers
    http://michael-coates.blogspot.com/2010/06/online-presentation-thursday-automated.html

- Live demo implementation
  http://michael-coates.blogspot.com/2011/02/live-demo-of-attack-aware-application.html

The threat: advanced attackers

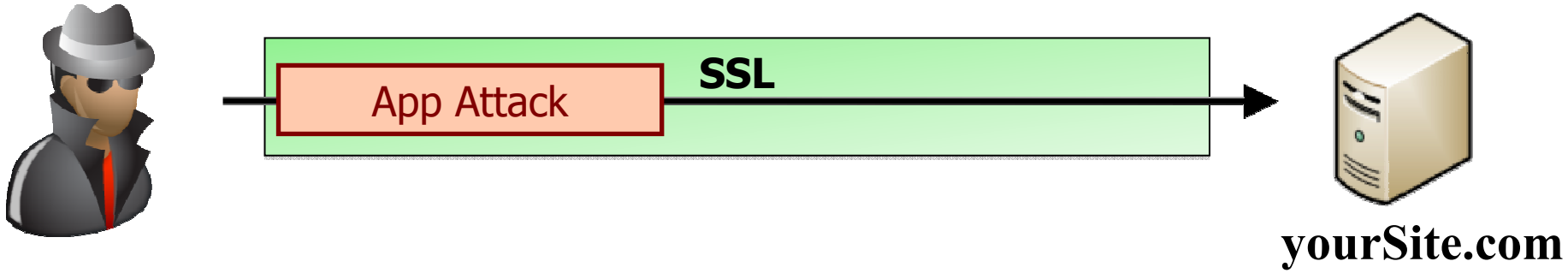
- Skilled
- Financially Motivated
- Organized
- Patient and Persistent
- In Possession of Your Source Code
- Outside & Inside Your Company

Application defence failures

- "We use SSL"
- "We use firewalls"
- "We use deep packet inspection"
- "We installed a web application firewall"

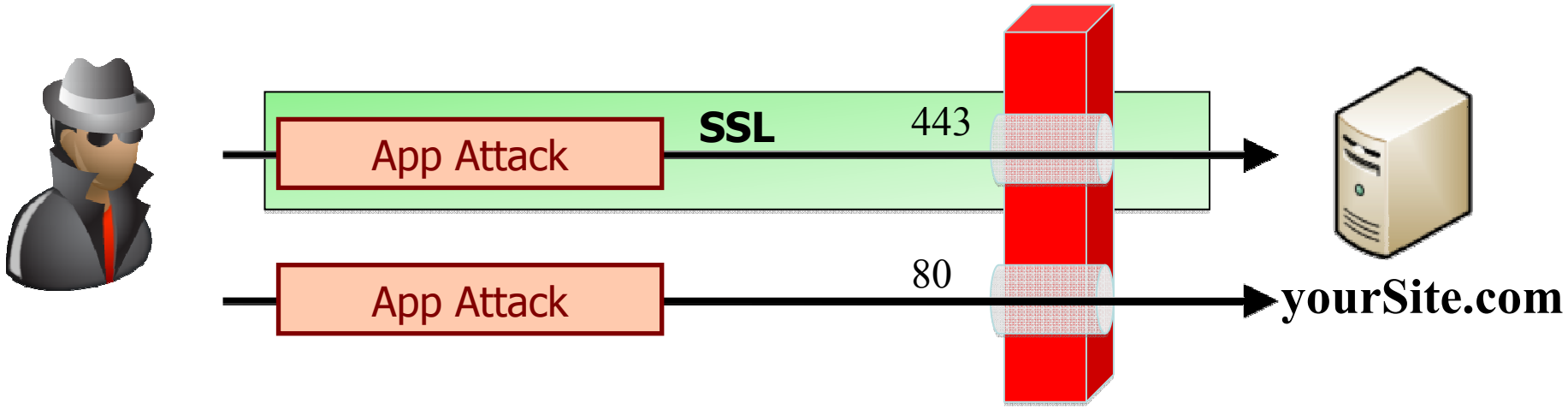
"We use SSL"

- SSL Protects Transmitted Traffic
- No Guarantee or Inspection of Data
- Zero Impact to Attackers
- Provides Zero Protection to Site Against Attackers



"We use firewalls"

- Purpose of Firewall: Allow or Deny Access via Port
- Necessity of Working Web App: Allowed Access via 80 or 443
- Result: Firewall is an Open Door



"We use deep packet inspection"

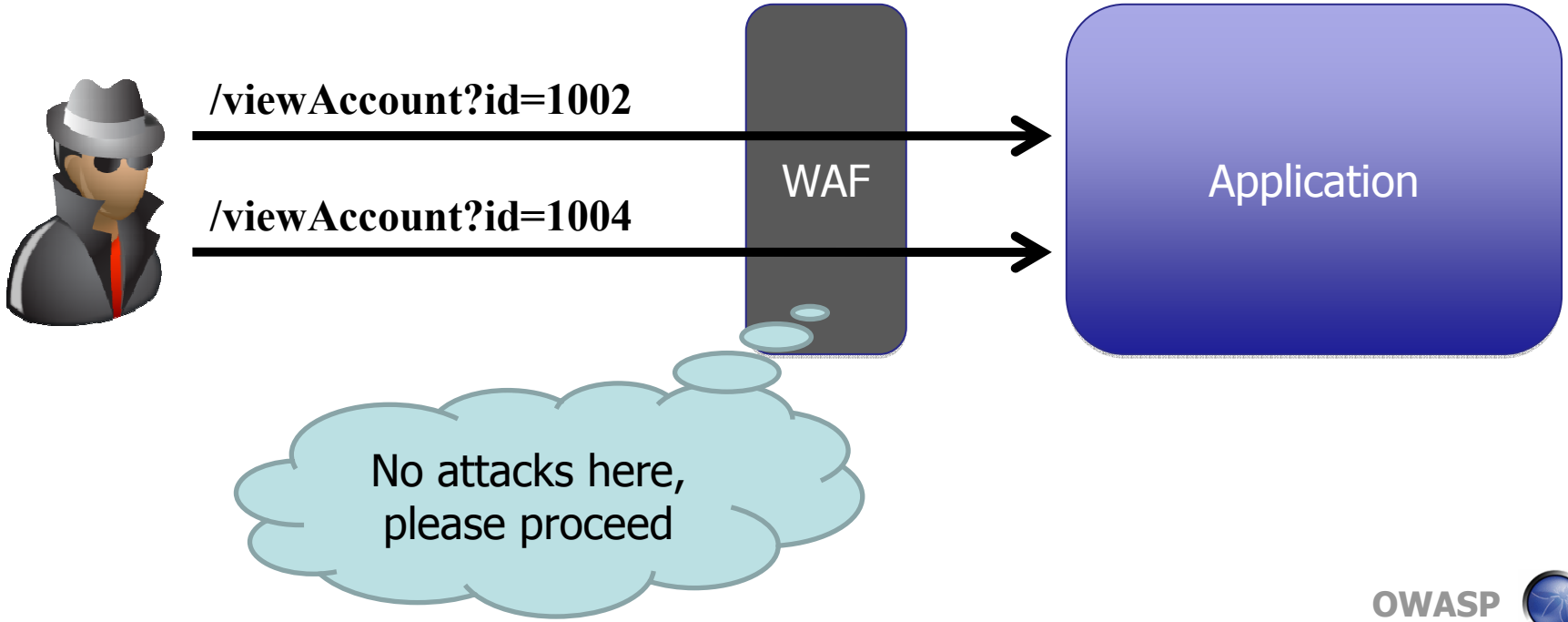
- Performed by Generic Network Appliance
- No Knowledge of Application Attacks

- Example Attack: Access Control Attack via Direct Object References
- Not Detected by DPI

```
GET /updateProfile?id=52473&pass=newpass
Host: yourSite.com
```

"We installed a web application firewall"

- Custom application + Generic Solution != success
- Application context not available
- No concept of access violations

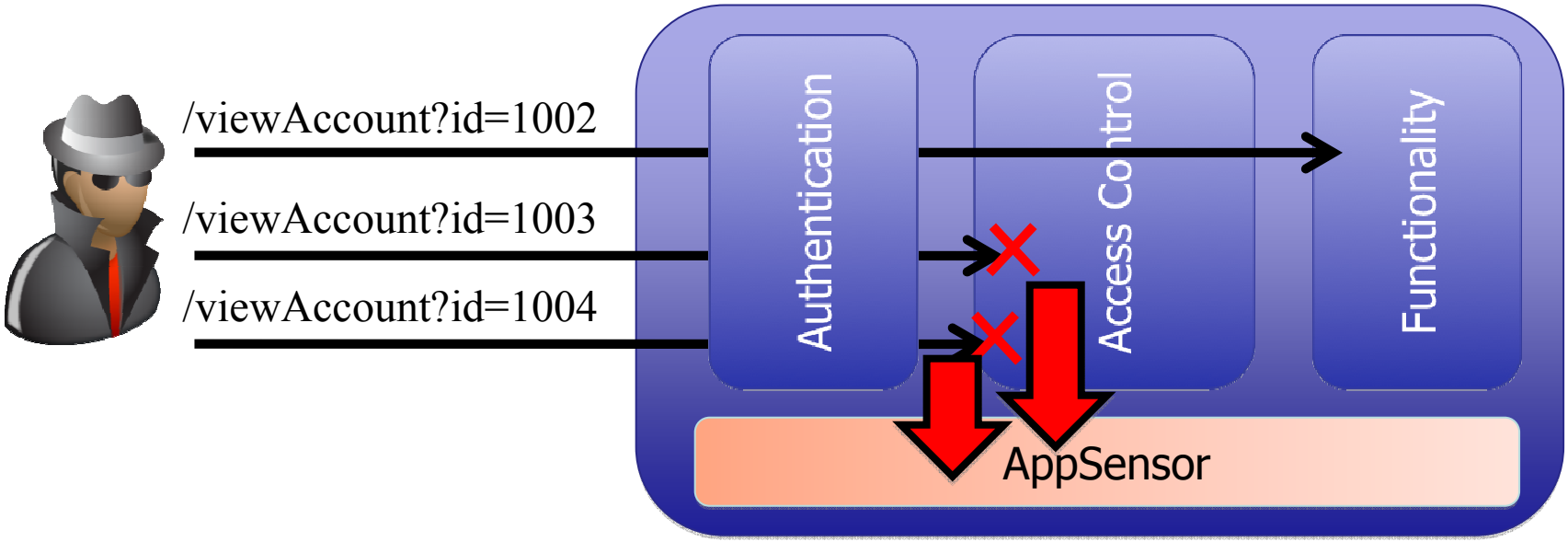


Detecting attacks the right way

- Integration
  - Detect INSIDE the application
  - Understand business logic
- Effectiveness
  - Minimal false positives
  - Immediate response
- Effort
  - Automatic detection
  - No manual work required

Inside the application is best

- Understand application & business context
- Integration with authentication & user store



Establishing detection points

Signature based events:

- Request
- Authentication
- Session
- Access control
- Input
- Exception
- Command injection
- File input/output
- Honey trap

Behaviour based events:

- User trend
- System trend
- Reputation

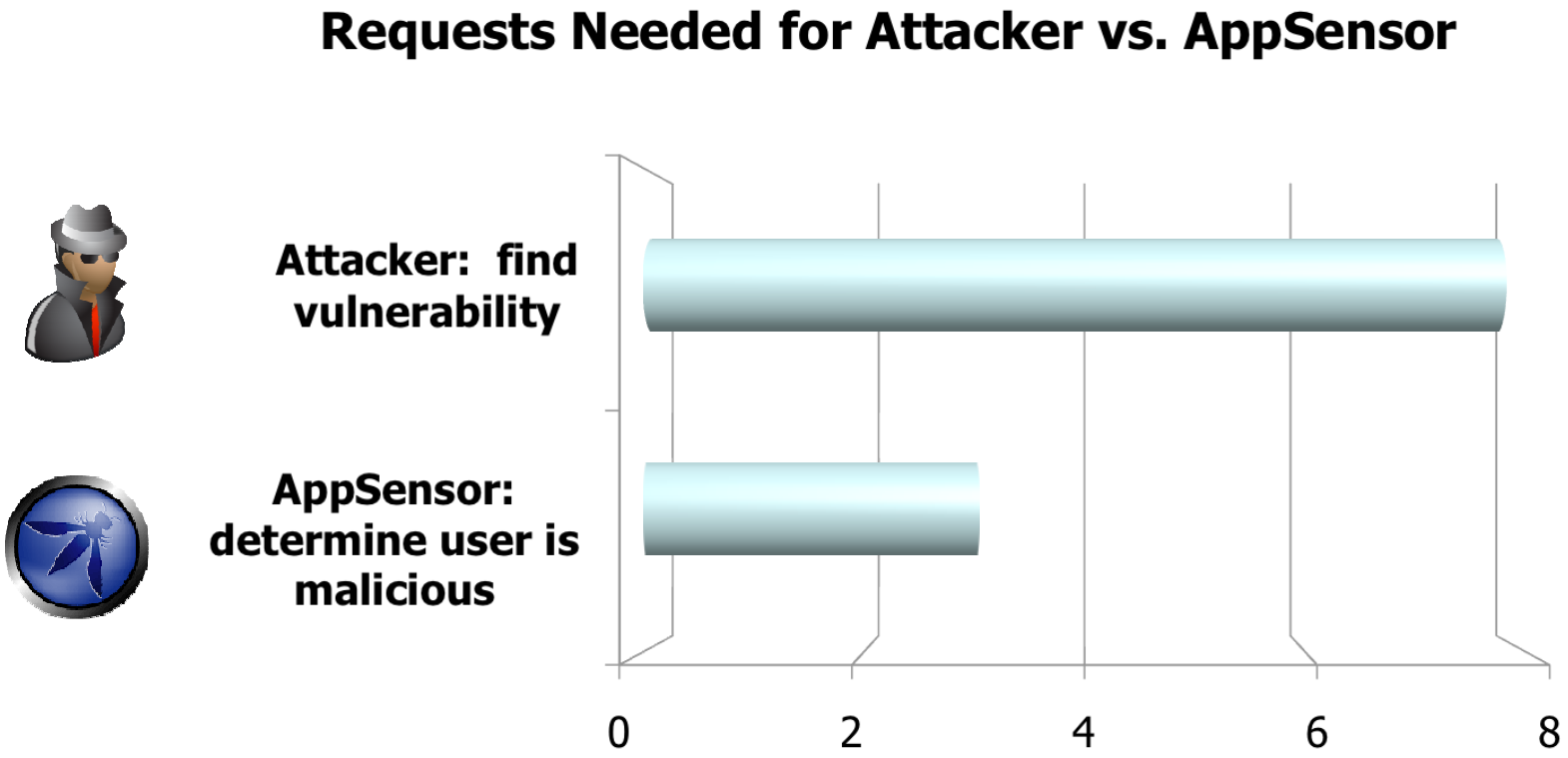
Detecting malicious users

- Many malicious attacks are obvious and not "user error"
  - POST when expecting GET
  - Tampering with headers
  - Submission of XSS attack

Examples of malicious actions

- Bypassing client side input validation
- Transaction using functionality
  not visible to user role
- Multiple access control violations
- Change of user agent midsession
- Double encoded data

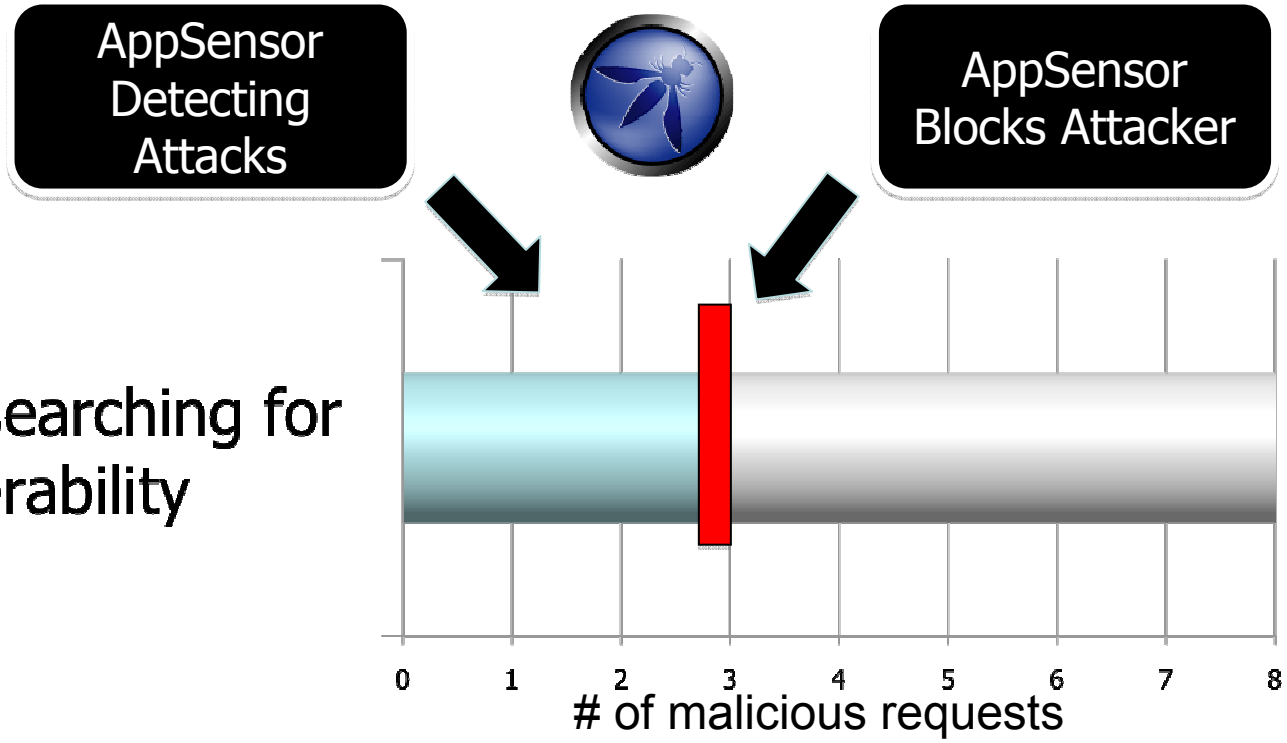
How does AppSensor protect the app?



AppSensor is faster than an attacker

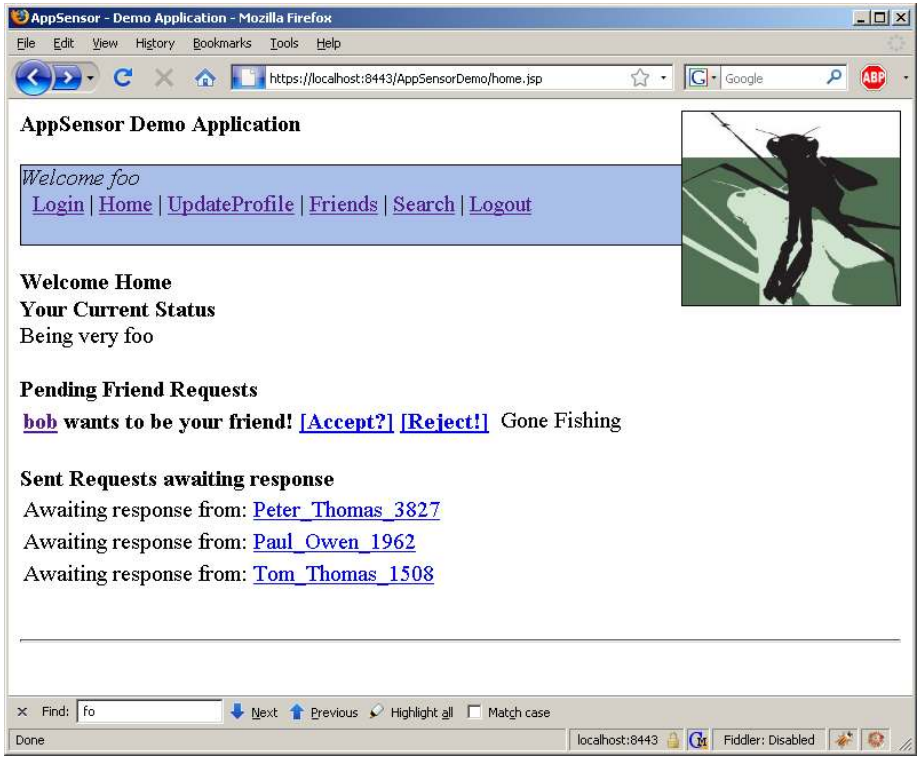
- User identified as malicious & blocked before vulnerability is found

Attacker: searching for vulnerability

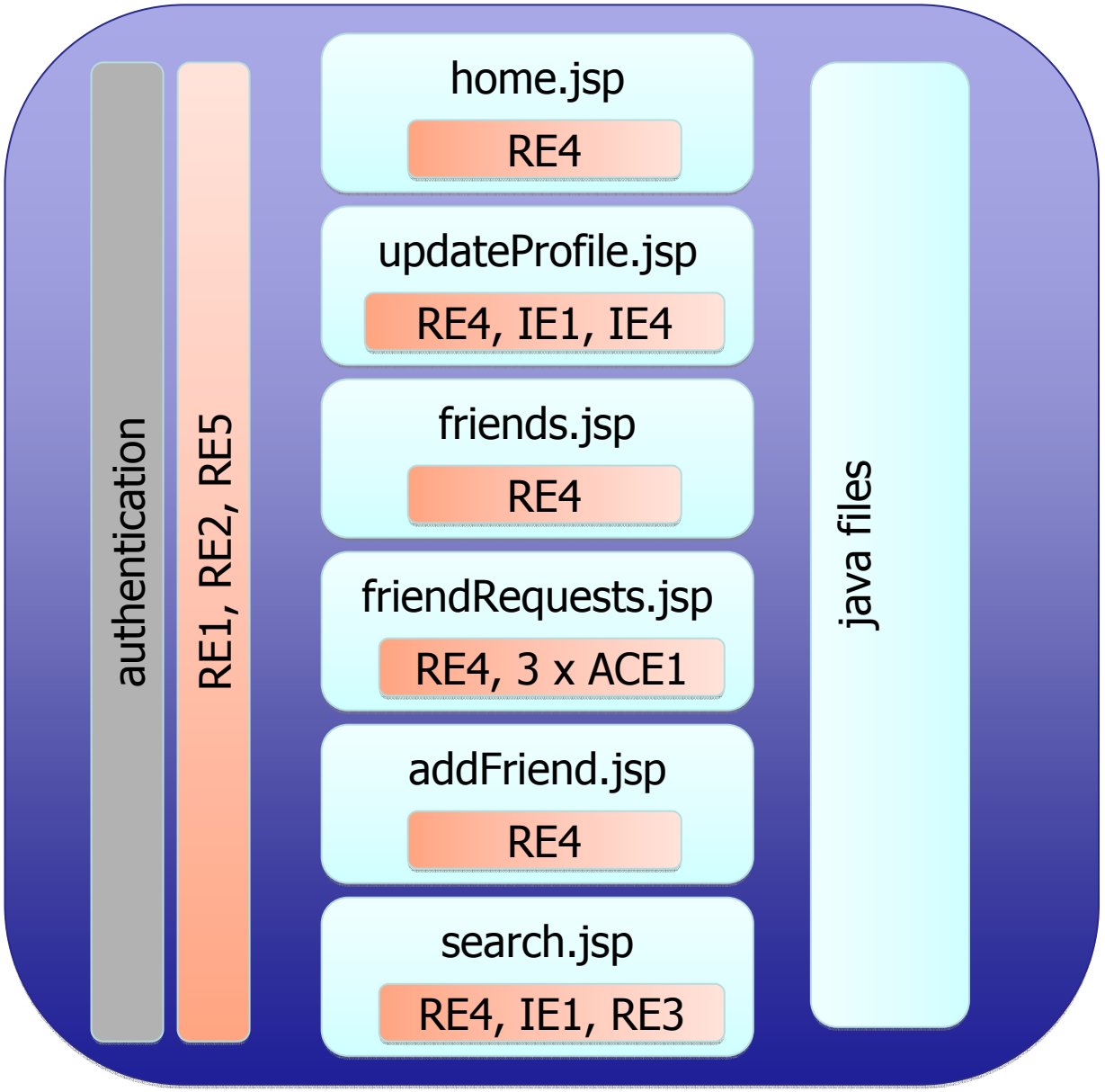


From theory to reality

- Demo Social Networking Application
- Leverages AppSensor Principles

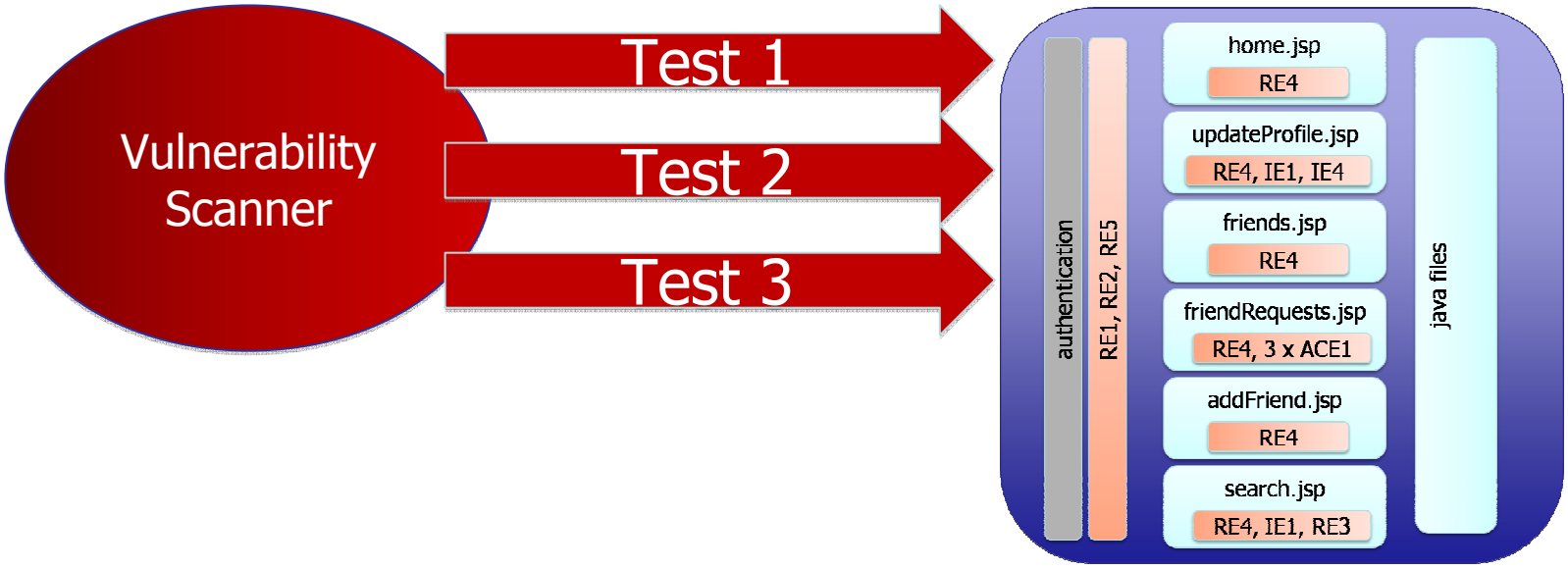


Detection points



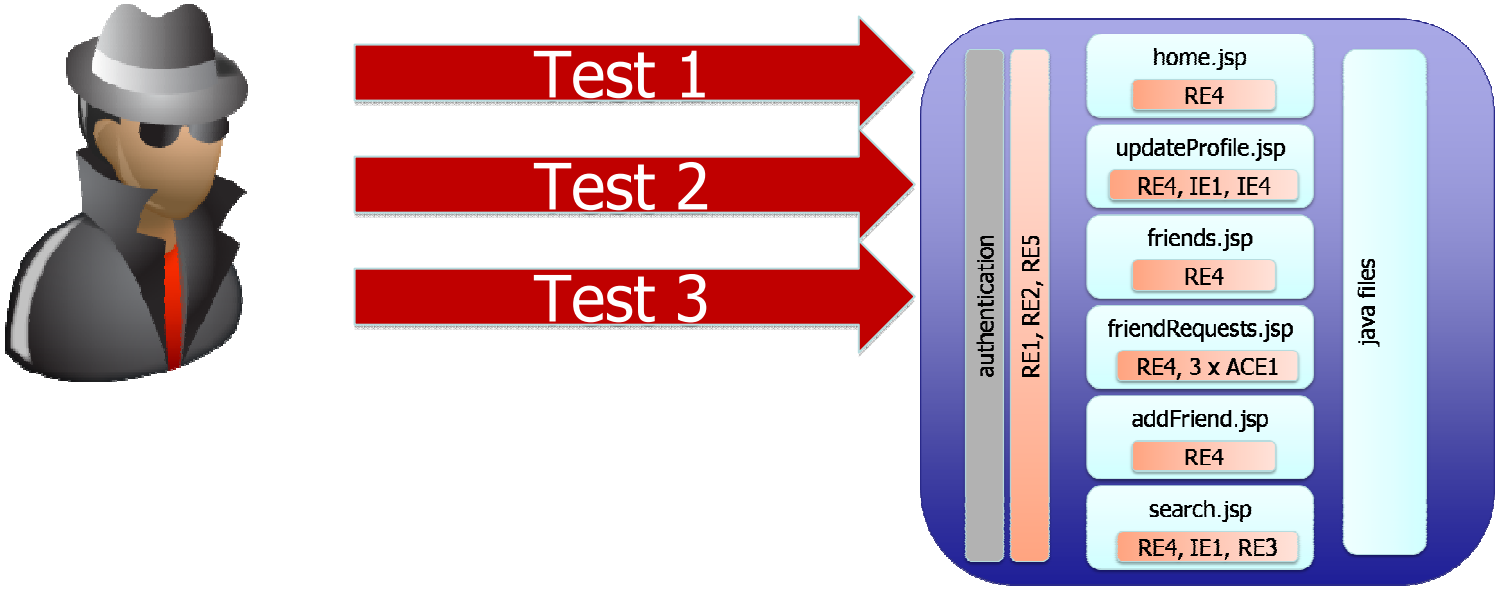
AppSensor vs scanners

- Tools attempt 10,000s of generic attacks
- AppSensor stops automated scans nearly instantly



AppSensor vs advanced attackers

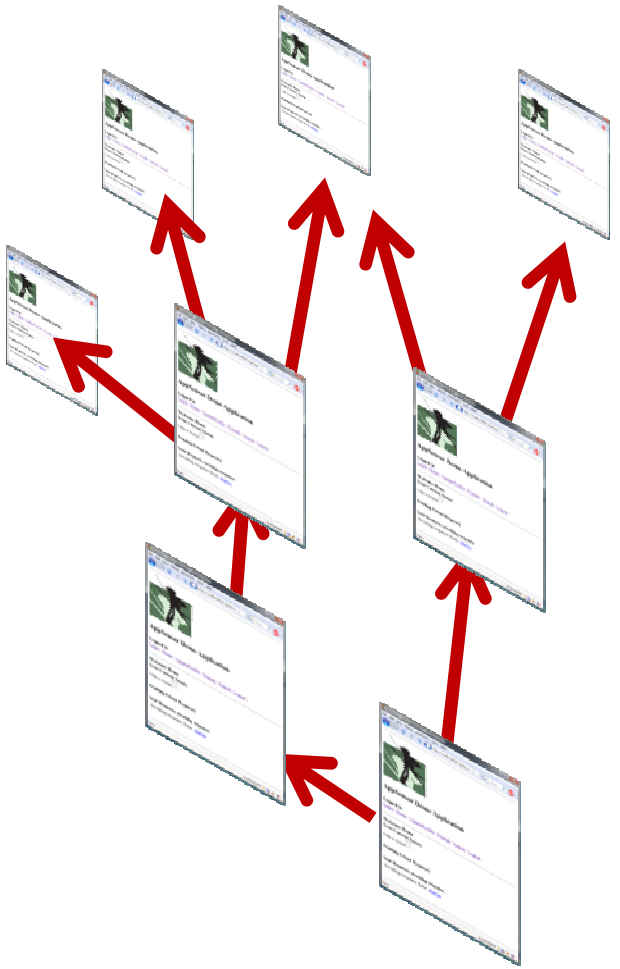
- Very difficult for attacker
- Requires advanced obfuscation for each attack
- Multiple probes == detection



Detecting/preventing an application worm

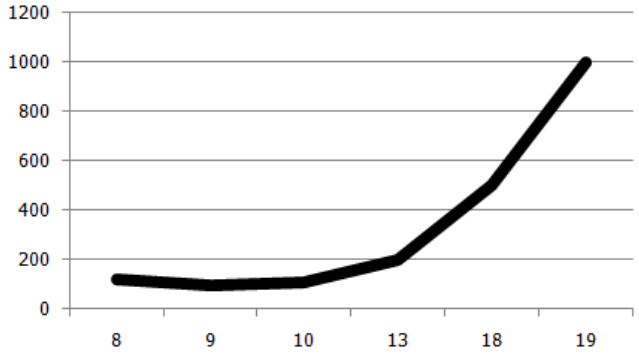
- Can you find / fix all XSS ?
- Pattern matching easily foiled
- Block the common factor!
  - Worms use XSS and CSRF for propagation
  - 1000% usage increase → problem
  - Our example:

    (updateProfile, updateStatus, updateName)



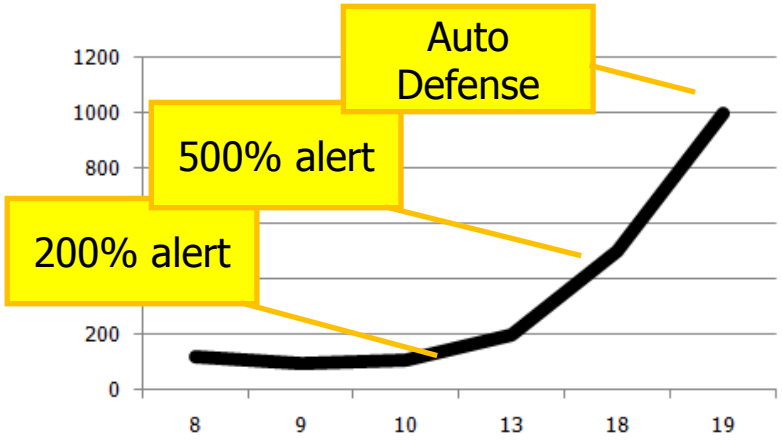
Case study: Samy

- MySpace Application Worm
- XSS worm embedded in User Profile
  - Added Samy as friend
  - Infected viewer's profile with XSS
- Exponential Growth of Samy's friends
  - 10 hours – 560 friends,
  - 13 hours – 6400 friends,
  - 18 hours – 1,000,000 friends,
  - 19 hours – site down for repair



Samy vs AppSensor

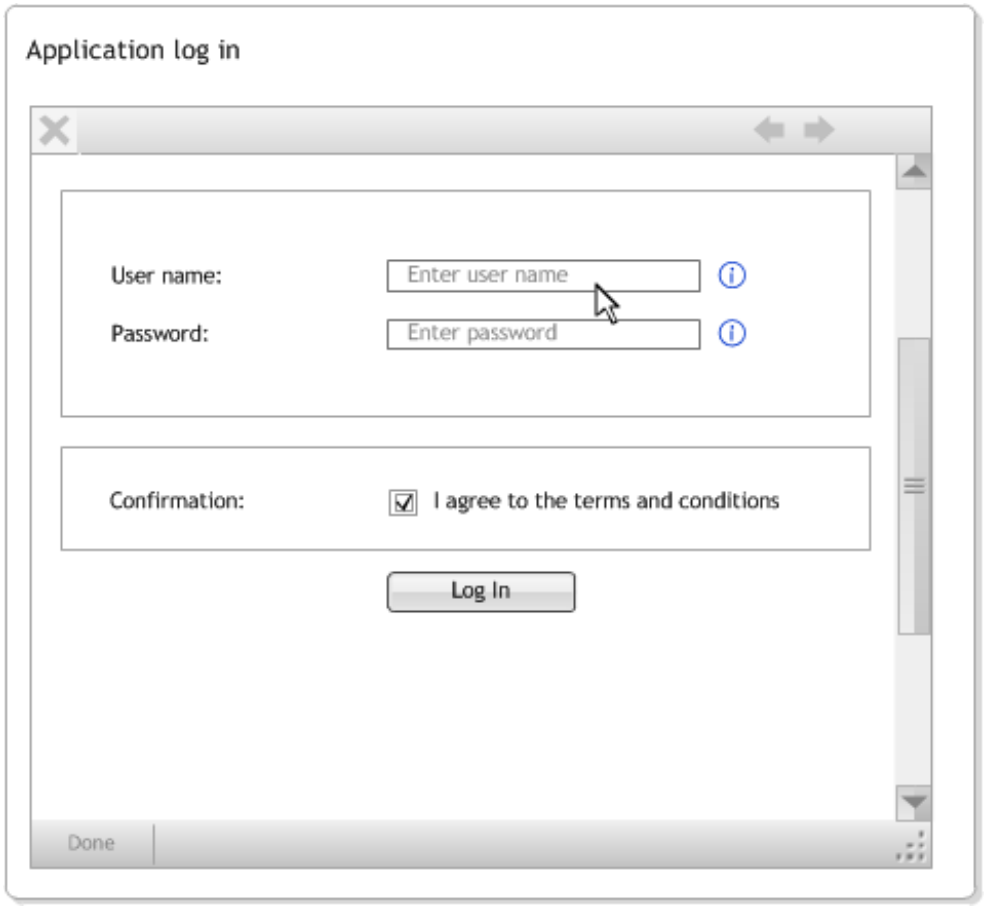
- AppSensor detects uptick in addFriend usage
- Compares against trended info
- Automatic response initiated
  - Alerts Admin +%200 Add Friend Usage
  - Alerts Admin 2nd time +%500 Add Friend Usage
  - Automatically shuts down Add Friend Feature
- Result:
  - Worm Contained,
  - Add Friend Temporarily Disabled,
  - Site Stays Up



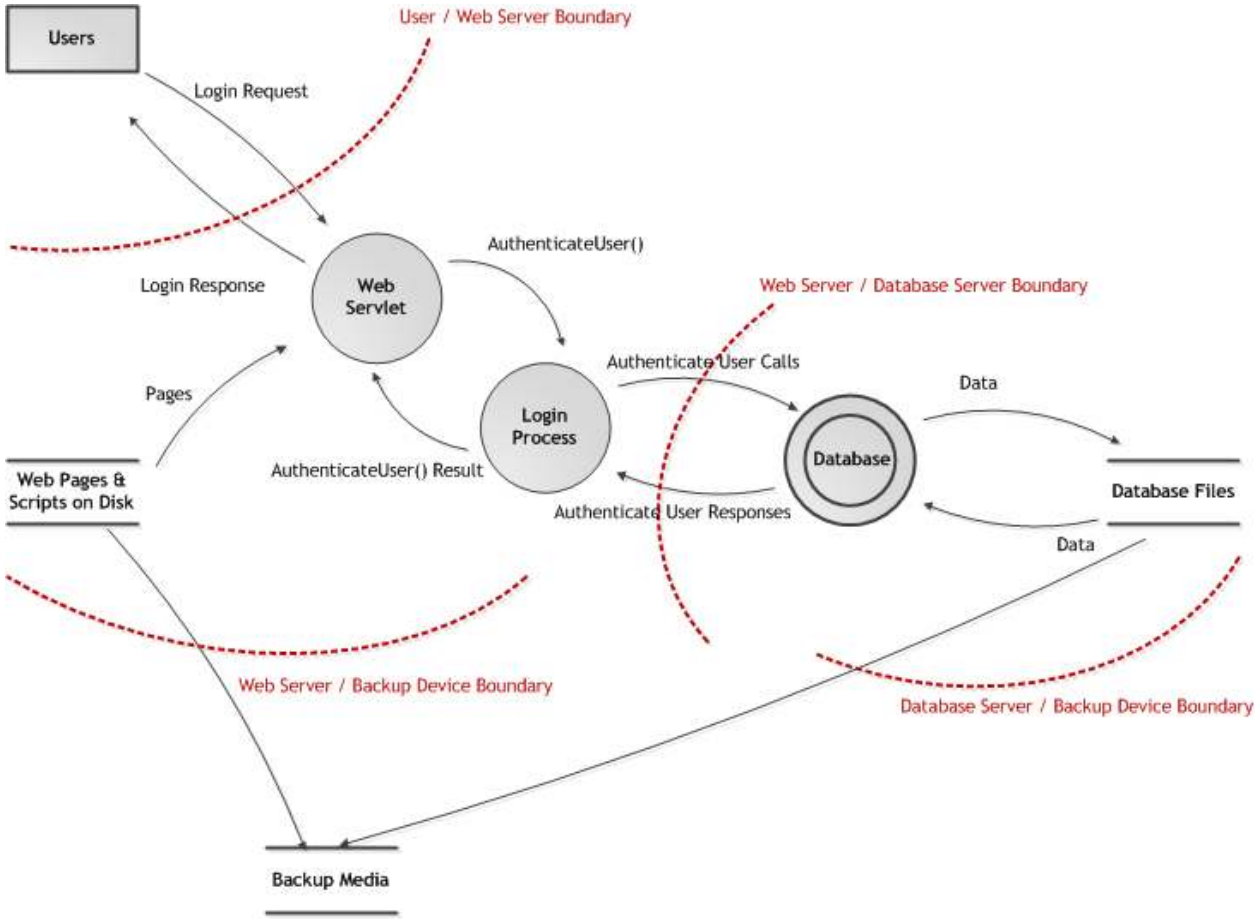
Trend monitoring benefits

- General
  - Insight to scripted traffic / attack probing
- Application worms
  - Auto detection of attacks
  - Automatic worm containment
  - Maintain overall site availability
- Fraud detection
  - Real time detection
  - Context specific
  - System-wide knowledge

AppSensor Specification & Design 1



AppSensor Specification & Design 2

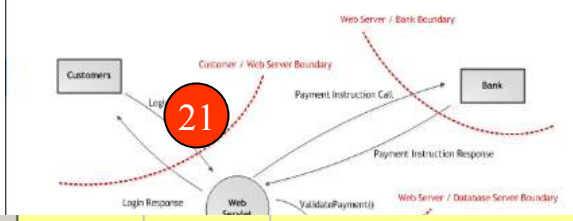


AppSensor Specification & Design 3

Detection Points		
Summary		
Request Exceptions (RE)	Access Control Exceptions (ACE)	User Trend Exceptions (UTE)
RE1: Unexpected HTTP Command	ACE1: Modifying URL Argument Within a GET for Direct Object Access Attempt	UT1: Irregular Use of Application
RE2: Attempt to Invoke Unsupported HTTP Method	ACE2: Modifying Parameter Within A POST for Direct Object Access Attempt	UT2: Speed of Application Use
RE3: GET When Expecting POST	ACE3: Force Browsing Attempt	UT3: Frequency of Site Use
RE4: POST When Expecting GET	ACE4: Evading Presentation Access Control Through Custom POST	UT4: Frequency of Feature Use
RE5: Additional/Duplicated Data in Request		
RE6: Data Missing from Request	Input Exceptions (IE)	System Trend Exceptions (STE)
RE7: Unexpected Quantity of Characters in Parameter	IE1: Cross Site Scripting Attempt	STE1: High Number of Logouts Across The Site
RE8: Unexpected Type of Characters in Parameter	IE2: Violation Of Implemented White Lists	STE2: High Number of Logins Across The Site
	IE3: Violation Of Implemented Black Lists	STE3: Significant Change in Usage of Same Transaction Across The Site
Authentication Exceptions (AE)	IE4: Violation of Input Data Integrity	
AE1: Use of Multiple Usernames	IE5: Violation of Stored Business Data Integrity	
AE2: Multiple Failed Passwords	IE6: Violation of Security Log Integrity	
AE3: High Rate of Login Attempts		
AE4: Unexpected Quantity of Characters in Username	Encoding Exceptions (EE)	
AE5: Unexpected Quantity of Characters in Password	EE1: Double Encoded Character	
AE6: Unexpected Type of Character in Username	EE2: Unexpected Encoding Used	
AE7: Unexpected Type of Character in Password		
AE8: Providing Only the Username	Command Injection Exceptions (CIE)	
AE9: Providing Only the Password	CIE1: Blacklist Inspection for Common SQL Injection Values	
AE10: Additional POST Variable	CIE2: Detect Abnormal Quantity of Returned Records	
AE11: Missing POST Variable	CIE3: Null Byte Character in File Request	
AE12: Utilization of Common Usernames	CIE4: Carriage Return or Line Feed Character in File Request	
Session Exceptions (SE)	File IO Exceptions (FIO)	
SE1: Modifying Existing Cookie	FIO1: Detect Large Individual File	
SE2: Adding New Cookie	FIO2: Detect Large Number of File Uploads	
SE3: Deleting Existing Cookie		Reputation (RP)
SE4: Substituting Another User's Valid Session ID or Cookie	Honey Trap (HT)	RP1: Suspicious or Disallowed User Source Location
SE5: Source Location Changes During Session	HT1: Alteration to Honey Trap Data	RP2: Suspicious External User Behavior
SE6: Change of User Agent Mid Session	HT2: Honey Trap Resource Requested	RP3: Suspicious Client-Side Behavior
	HT3: Honey Trap Data Used	RP4: Change to Environment Threat Level

AppSensor Specification & Design 4

Example Data Flow Diagram



Locally Scoped Detection Points

Type	Identity	Description

	Type	Identity	Description
3			
4	Locally Scoped Detection Points		
5	Type	Identity	Description
6	**AE6**	**21**	**Username form value contains invalid characters**
7			

AppSensor Specification & Design 5

Detecti	Detection Point Definition
Details	
Type	Details
Description	The username field is compared to a whitelist of allowable characters. This property is defined in the site's database, and allocated to the form identity and specific element name. NB This detection point ONLY matches on the specific form identity, entry point and field name. Invalid entry points, form identities and other field names need to be examined separately.
Pre-Requisites	None

AppSensor Specification & Design 6

Detection Point Definition					
Application Summary					
Series					
Application					
Domain(s)/Port(s)					
Detection Points					
Identity	Target	Module	Function	Entry Points	Comments

Detection Points				
Identity	Target	Module	Function	Entry Points
21	Username	dite.dbo/auth	checkUser	/login.aspx /loginTask.aspx

AppSensor Specification & Design 7

- Model development
- Optimisation
- Code location
- Attack analysis

AppSensor Specification & Design 8

Response Actions				
Summary				
ASR-A	Logging Change		ASR-D	User Status Change
Classifications	Logging | One, some or all users | Instantaneous (request) or for a period		Classifications	Logging | One user | For a period
Category	Silent		Category	Passive
Description	The granularity of logging is changed (typically more logging).		Description	A parameter related to the user is modified. This may have an impact on functionality or usability of the application, but only for the one user.
Considerations	-		Considerations	-
Examples	Example 1: Capture sanitised request headers and response bodies Example 2: Full stack trace of error messages logged Example 3: Record DNS data on user's IP address Example 4: Security logging level changed to include 'informational' messages		Examples	Example 1: Internal trustworthiness scoring about the user changed Example 2: Reduce payment transfer limit for the customer before additional out-of-band verification is required Example 3: Reduce maximum file size limit for each file upload by the forum user Example 4: Increase data validation strictness for all form submissions by this citizen Example 5: Reduce the number of failed authentication attempts allowed before the user's account is locked (ASR-K below)
ASR-B	Administrator Notification		ASR-E	User Notification
Classifications	Logging and notifying | One, some or all users | Instantaneous		Classifications	Logging, notifying and disrupting | One user | Instantaneous
Category	Silent		Category	Passive
Description	A notification message is sent to the application administrator(s)		Description	A visual, audible and/or mechanical (e.g. vibration) signal or message is activated, displayed, or sent by other means, to the user.
Considerations	-		Considerations	-
Examples	Example 1: Email alert sent to everyone in the administration team Example 2: SMS alert sent to the on-call administrator Example 3: Visual indicator displayed on an application monitoring dashboard Example 4: Audible alarm in the control room		Examples	Example 1: On-screen message about mandatory form fields (e.g. "The 'occupation' must be completed") Example 2: On-screen message about data validation issues (e.g. 'The bank sort code can only contain six digits with optional hyphens') Example 3: Message sent by email to the registered email address to inform them their password has been changed

AppSensor Specification & Design 9

- Strategic requirements
- Thresholds
- Model tuning

- Implementation

- Monitoring and tuning

Bring AppSensor into your application

A. Build it into requirements
B. Develop your own
   - Detection points:
   - http://www.owasp.org/index.php/AppSensor_DetectionPoints
   - AppSensor methodology:
   - https://www.owasp.org/images/2/2f/OWASP_AppSensor_Beta_1.1.pdf
C. ESAPI
   - AppSensor Integration into Java ESAPI
D. Security Information/Event Management?
   - Add detection points into application
   - Integrate logging into real time monitor

Trending topic

"Other elements of the Pentagon's strategy include developing active defenses - technologies that detect attacks and probes as they occur, as opposed to defenses that employ only after-the-fact detection and notification..."

Deputy Defense Secretary William Lynn, US Dept of Defense , February 2011

"[develop a] framework for capturing and analyzing application and session data in order to isolate criminal behaviors"

CISO, US bank, February 2011

Full day AppSensor training (provisional)

- AppSec EU
  6th-10th June, Dublin, Ireland
  http://www.appseceu.org

- AppSec USA
  20th-23rd September, Minneapolis, USA
  http://www.appsecusa.org

End

colin.watson(at)owasp.org