OWASP AppSensor
The Future of Application Security

Dennis Groves, MSc
dennis.groves@owasp.org

November 18, 2013
About Me
A Thought Experiment
The Bottom Line on OPSEC:

We all have information that the Bad Guys need to hurt us. We don't want them to get it. The OPSEC process helps us to look at our world through the eyes of an adversary and to develop measures in order to deny them. Get it?

**The OPSEC Process:**

1. Identify Critical Info
2. Analyze Threats
3. Analyze Vulnerabilities
4. Assess the Risks
5. Apply Countermeasures

Think about it... all the time!

5 STEPS... 1 MINDSET

What is Operations Security? Operations Security, or OPSEC, is a risk management methodology used to deny an adversary information concerning our intentions and capabilities by identifying, controlling, and protecting critical information associated with the planning and execution of a mission.
1: Identify Critical Info
Vitorian Duel
Theorem

Protection time must be greater than or equal to detection time plus reaction time.

\[ P_t \geq D_t + R_t \]
Wester Duel
Our Opponent

Pistol Duel
- Novice Shooter
- Weekend Shooter
- Professional Shooter
- Quick Draw Champion

Application
- Script Kiddies
- Hacktivists
- Criminals
- Disgruntled Employee
- Corporate Spy
- Cyber Warrior
Out Gunned
2: Analyze Threats

Pistol Duel
- Handgun Skills
- Nervousness
- Psychological Readiness

Application
- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privilege
3: Analyze Vulnerabilities

Pistol Duel
- Jam
- Misfire
- Backfire

The OWASP Top-10
- A1 Injection
- A3 Cross-Site Scripting
- A5 Security Misconfiguration
- A7 Missing Access Control
4: Analyze Risks

The probable frequency and probable magnitude of future loss

(2) \[ \text{Risk} = P(\text{Impact}) \]
(3) \[ \text{Risk} = P(\text{Impact} \times \text{Vulnerability}) \]
(4) \[ \text{Risk} = \text{Impact} \times \text{Vulnerability} \times \text{Threat} \]
(5) \[ \text{Risk} = P(\text{Impact} \times \text{Vulnerability} \times \text{Threat}) \]
(6) \[ \text{Risk} = \frac{\text{Impact} \times \text{Vulnerability} \times \text{Threat}}{\text{Countermeasures}} \]
(7) \[ \text{Risk} = \text{Impact} \times \frac{P(\text{Threat}) \times P(\text{Vulnerability})}{\text{Countermeasures}} \]
5: Apply Countermeasures

- Tolerate: Do nothing.
- Transfer: Outsource the risk.
- Terminate: Eliminate the asset.
- Treat: Reduce the risk.
Risk Reduction Methods

Reducing the risk (treatment) is the most common strategy used today.

- Reduce the probability of a threat.
- Reduce the probability of a vulnerability.
Reduce Attack Surface

Pistol Duel
- Turn To The Side
- Crouch Down Low
- ???

Application
- Penetration Testing
- Code Review
- Patching
Predicting the Future
### Game Theory

#### Prisoners' Dilemma

<table>
<thead>
<tr>
<th>Prisoner A</th>
<th>Confess</th>
<th>Remain Silent</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Confess</strong></td>
<td>5 years</td>
<td>20 years</td>
</tr>
<tr>
<td><strong>Remain Silent</strong></td>
<td>0 year</td>
<td>1 year</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prisoner B</th>
<th>Confess</th>
<th>Remain Silent</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Confess</strong></td>
<td>5 years</td>
<td>0 year</td>
</tr>
<tr>
<td><strong>Remain Silent</strong></td>
<td>20 years</td>
<td>1 year</td>
</tr>
</tbody>
</table>
Basic Statistics

The diagram represents a normal distribution with the following characteristics:

- The area under the curve is divided into sections corresponding to $-3\sigma$, $-2\sigma$, $-1\sigma$, $\mu$, $1\sigma$, $2\sigma$, and $3\sigma$.
- The percentage of data within these intervals is as follows:
  - $0.1\%$ within $-3\sigma$ of $\mu$.
  - $2.1\%$ within $-2\sigma$ of $\mu$.
  - $13.6\%$ within $-1\sigma$ of $\mu$.
  - $34.1\%$ within $1\sigma$ of $\mu$.
  - Another $34.1\%$ within $1\sigma$ of $\mu$.
  - $13.6\%$ within $2\sigma$ of $\mu$.
  - $2.1\%$ within $3\sigma$ of $\mu$.
  - $0.1\%$ within $3\sigma$ of $\mu$.

This distribution is symmetric around the mean ($\mu$), and the majority of the data falls within $1\sigma$ of the mean.
Risk Optimisation is rarely practiced, but highly effective method.

- Reduce the impact of an event
Add Risk Reduction Practices

- AppSensor is not a panacea nor is a vest
- You do not want to get shot, but if you do, you want to be wearing a vest
- If you get shot while wearing a vest, it is going to hurt, but you will survive
Michael Coates, Colin Watson, John Melton Ryan Barnett, Simon Bennetts, Marc Chisinevski, Robert Chonjnacki, August Detlefsen, Sean Fay, Randy Janida, Alex Lauerman, Manuel Arredondo, Bob Maier, Craig Munson, Giri Nambari, Abdul Rauf, Jay Reynolds, Eric Sheridan, John Steven, Alex Thissen, Don Thomas, Kevin Wall, Mehmet Yilmaz, Jim Manico, Dinis Cruz, myself and many, many others...
Defensive Measures

- Detection & Reaction in the App
- Attack-Aware Detection
- Normal and Malicious Behavior
- Evasion and Unknown Attacks
<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Signature</td>
<td>RE</td>
<td>Request Exceptions</td>
</tr>
<tr>
<td></td>
<td>AE</td>
<td>Authentication Exceptions</td>
</tr>
<tr>
<td></td>
<td>SE</td>
<td>Session Exceptions</td>
</tr>
<tr>
<td></td>
<td>ACE</td>
<td>Access Control Exceptions</td>
</tr>
<tr>
<td></td>
<td>IE</td>
<td>Input Exceptions</td>
</tr>
<tr>
<td></td>
<td>EE</td>
<td>Encoding Exceptions</td>
</tr>
<tr>
<td></td>
<td>CIE</td>
<td>Command Injection Exceptions</td>
</tr>
<tr>
<td></td>
<td>FIO</td>
<td>File IO Exceptions</td>
</tr>
<tr>
<td></td>
<td>HT</td>
<td>Honey Trap</td>
</tr>
<tr>
<td>Behavioural</td>
<td>UTE</td>
<td>User Trend Exceptions</td>
</tr>
<tr>
<td></td>
<td>STE</td>
<td>System Trend Exceptions</td>
</tr>
<tr>
<td></td>
<td>RP</td>
<td>Reputation</td>
</tr>
<tr>
<td>Response Type</td>
<td>Examples</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Logging Change</td>
<td>Full stack trace of error messages logged</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Record DNS data on user’s IP address</td>
<td></td>
</tr>
<tr>
<td>Account Logout</td>
<td>Session terminated and user redirected</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Session terminated only (no redirect)</td>
<td></td>
</tr>
<tr>
<td>Account Lockout</td>
<td>User account locked permanently</td>
<td></td>
</tr>
<tr>
<td></td>
<td>One user’s IP address range blocked</td>
<td></td>
</tr>
<tr>
<td>Application Disabled</td>
<td>Website shut down and replaced with static page</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Application taken offline</td>
<td></td>
</tr>
</tbody>
</table>
Future AppSensor Developments

- AppSensor-core
- AppSensor/ws-soap
- AppSensor/ws-rest
- AppSensor Guide
Goal: Produce viable implementation that allows intrusion detection to move towards a functional primitive in any language

- It should be as simple as possible to detect and respond to events in your environment
Reference Implementation

- Existing V1
  - Java only (requires developers to re-implement full system in any other language)
  - Built to work with ESAPI (difficult to remove dependency)
  - Functional, but missing many features

- It should be as simple as possible to detect and respond to events in your environment
New V2 (in progress)

- Java core backend
- Services (rest/soap) enable front-end in any lang
  - only re-implement minor portions, significant analysis done 1.me
  - We can build several reference front-ends (help!)
- Basic correlation between applications
- Allows input from external systems (WAF, IDS, etc.)
- Enables reporting
AppSensor V1 (Java Only)
How Can You Help?

- Join the Mailing List and Participate
- Help us develop reference implementations
- Tell your friends, and employers
Obrigado!
Questions?

You have Questions

We have Answers