WHAT IT TAKES TO RUN AMERICA’S VULNERABILITY MANAGEMENT TEAM

Panelists: Sandy Radesky (moderator), Chris Hughes (co-moderator), Bob Lord, Lindsey Cerkovnik, Pat Garrity
VM Operational Design | What We Do

VM Mission | Enable cyber risk reduction to proactively strengthen national infrastructure resilience.

VM Vision | Trusted, credible, and timely vulnerability ecosystem management advantages for the nation.

VM Essential Tasks

- Security at Scale
  - Scale Technology Safety, Security, and Quality
- Vulnerability Response & Coordination
  - Understand Vulnerabilities to Minimize Attack Risk
  - Prioritize Vulnerabilities
- Attack Surface Evaluation
  - Maintain Visibility of the Attack Surface
  - Prioritize Vulnerabilities
- Cyber Resilience
  - Measure and Assess Cyber Risk
  - Prioritize Risk Practices

VM Mission

- Understand Vulnerabilities to Minimize Attack Risk
- Prioritize Vulnerabilities

VM Vision

- Trusted, credible, and timely vulnerability ecosystem management advantages for the nation.

Insights

VULNERABILITY MANAGEMENT

Scale Technology Safety, Security, and Quality

Understand Vulnerabilities to Minimize Attack Risk

Attacks, Cyber

Vulnerabilities

Strengthen Resilience

Operational Cyber Risk

March 26, 2024
VM Customers equal “America”

Federal Civilian Executive Branch (FCEB) Agencies

State, Local, Tribal, and Territorial (SLTT)

Critical Infrastructure (CI) and Private Sector Partners
What we’ve all Done over the years!
Software manufacturer introduces security defect

Software manufacturer learns of defect

Software manufacturer issues software update

Customers deploy software update

Outsiders learn of the security defect

Exploitation in the wild (A)

Outsiders notify software manufacturer

Exploitation in the wild (B)

Defenders unable to remediate (A)

Defenders unable to remediate (B)

Topic 1: Vulnerability Response Life Cycle (Today)

• CVD Process
• Rapid Action Force
• Industry Collaboration
• Notifications
• KEV
Software manufacturer introduces security defect

Outsiders learn of the security defect

Exploitation in the wild (A)

Outsiders notify software manufacturer

Exploitation in the wild (B)

Defenders unable to remediate (B)

Software manufacturer learns of defect

Software manufacturer issues software update

Customers deploy software update

Defenders unable to remediate (A)

How do we drive feedback into the system to eliminate the unforgivable vulnerabilities?

- Secure by Design
- PSIRTs
- CWE linkage
- C-SCRM
- SBOM/VEX
### 2007 Unforgivable Vulnerabilities

<table>
<thead>
<tr>
<th>CWE-ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWE-120</td>
<td>1) Buffer overflow using long strings of “A” characters</td>
</tr>
<tr>
<td>CWE-79</td>
<td>2) XSS using well-formed SCRIPT tags</td>
</tr>
<tr>
<td>CWE-89</td>
<td>3) SQL injection using '</td>
</tr>
<tr>
<td>CWE-98</td>
<td>4) Remote file inclusion from direct input such as: a. include($_GET['dir'] . &quot;/config.inc&quot;);</td>
</tr>
<tr>
<td>CWE-23</td>
<td>5) Directory traversal using &quot;/a/b/c&quot; in &quot;GET&quot; or &quot;SEND&quot; commands of frequently-used file sharing functionality</td>
</tr>
<tr>
<td>CWE-425</td>
<td>7) Direct requests of administrator scripts</td>
</tr>
<tr>
<td>CWE-472</td>
<td>8) Grow-your-own crypto</td>
</tr>
<tr>
<td>CWE-61</td>
<td>9) Authentication bypass using &quot;authenticated=1&quot; cookie/form field</td>
</tr>
<tr>
<td>CWE-271</td>
<td>10) Turtle race condition - symlink</td>
</tr>
<tr>
<td>CWE-259</td>
<td>11) Privilege escalation launching &quot;help&quot; (Windows)</td>
</tr>
<tr>
<td>CWE-190</td>
<td>12) Hard-coded or undocumented account/password</td>
</tr>
<tr>
<td>CWE-190</td>
<td>13) Unchecked length/width/height/size values passed to malloc()/calloc()</td>
</tr>
</tbody>
</table>

### 2023 Stubborn Weaknesses

<table>
<thead>
<tr>
<th>CWE-ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CWE-787</td>
<td>Out-of-bounds Write</td>
</tr>
<tr>
<td>CWE-79</td>
<td>Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')</td>
</tr>
<tr>
<td>CWE-89</td>
<td>Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')</td>
</tr>
<tr>
<td>CWE-416</td>
<td>Use After Free</td>
</tr>
<tr>
<td>CWE-78</td>
<td>Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')</td>
</tr>
<tr>
<td>CWE-20</td>
<td>Improper Input Validation</td>
</tr>
<tr>
<td>CWE-125</td>
<td>Out-of-bounds Read</td>
</tr>
<tr>
<td>CWE-22</td>
<td>Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')</td>
</tr>
<tr>
<td>CWE-352</td>
<td>Cross-Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>CWE-476</td>
<td>NULL Pointer Dereference</td>
</tr>
<tr>
<td>CWE-287</td>
<td>Improper Authentication</td>
</tr>
<tr>
<td>CWE-190</td>
<td>Integer Overflow or Wraparound</td>
</tr>
<tr>
<td>CWE-502</td>
<td>Deserialization of Untrusted Data</td>
</tr>
<tr>
<td>CWE-119</td>
<td>Improper Restriction of Operations within Bounds of a Memory Buffer</td>
</tr>
<tr>
<td>CWE-798</td>
<td>Use of Hard-coded Credentials</td>
</tr>
</tbody>
</table>

Memory safety CWEs

TLP CLEAR
Thoughts on Next Steps

- How do we get to actual risk reduction?
- How can this group become a catalyst for change?
- What do we think software companies need to do to make safer higher quality software?