



An Information Assurance / Cyber Security Company

*VA Certified Service Disabled Veteran Owned Small Business (SDVOSB)*

## ***An Introduction to API Security***

***[AUDIENCE]***

***October 11, 2012***

ISO 9001: 2008 ♦ Maturity Level 2 of CMMI® ♦

DCAA Approved Accounting System ♦

Approved Earned Value Management (EVM) ♦ TS Facility Clearance

Contract Vehicles

GSA Schedule 70 ♦ VA BPA ♦

ALLIANT ♦ ENCORE II ♦ Seaport E ♦ SDV GWAC: Team SMS ♦ FAA eFAST

# Agenda



- API Security Overview – What are the basic principles of API Security?
- API Security Issues - What are some common security issues associated with APIs?
- API Security Team – Who can secure an API? What resources are needed?

# General Principles of APIs



- APIs are very similar to websites from a security perspective - the same vulnerability checks that work against websites will work against APIs
- The risks of deploying an API can often be mitigated using the same measures used to secure a website
- *If a strong application security program is already in place, additional resources (human or technical) are generally unnecessary when securing an API*



# APIs vs. Websites



- From a security standpoint, APIs can essentially be treated just like websites due to the similar (often exact same) technologies being used
  - URLs and APIs are both used to access information
  - Information is requested, an action is performed, and information is returned
- A few key differences
  - While a website is meant for humans to look at, APIs are like websites for machines – they are meant almost entirely for machines to read
  - Information is generally returned in much safer formats via APIs than via standard websites; most times there is no client-side code execution with APIs



# APIs vs. Websites - Example



URL Requested: <http://www.website.com/resource>

Response when requested from browser:

## Affected Platforms

**Architectures:** N/A

**Environments:** N/A

**Hardware Architectures:** N/A

**Operating Systems:** N/A

**Programming Languages:** C, C++, Java, .NET

**Technology Classes:** N/A

## Consequences

Scope	Technical Impact	Notes
Other	Quality degradation, Varies by context	

## Potential Mitigations

Phase	Strategy	Description	Effectiveness	Notes
Architecture and Design		Use a hash table instead of an alist.		
Architecture and		Use an alist which checks the uniqueness of hash keys with each entry		

# APIs vs. Websites - Example



URL Requested: <http://www.website.com/resource>

Response when requested from tools using an API:

```
<Weakness ID="462" Name="Duplicate Key in Associative List (Alist)" Status="Incomplete" Weakness_Abstraction="Base">
  <Description>
    <Description_Summary>Duplicate keys in associative lists can lead to non-unique keys
      being mistaken for an error.</Description_Summary>
    <Extended_Description>
      <Text> A duplicate key entry -- if the alist is designed properly -- could be
        used as a constant time replace function. However, duplicate key entries
        could be inserted by mistake. Because of this ambiguity, duplicate key
        entries in an association list are not recommended and should not be
        allowed. </Text>
    </Extended_Description>
  </Description>
  <Relationships>
    <Relationship>
      <Relationship_Views>
        <Relationship_View_ID Ordinal="Primary">699</Relationship_View_ID>
      </Relationship_Views>
      <Relationship_Target_Form>Category</Relationship_Target_Form>
      <Relationship_Nature>ChildOf</Relationship_Nature>
      <Relationship_Target_ID>461</Relationship_Target_ID>
      <!--Data Structure Issues-->
    </Relationship>
    <Relationship>
      <Relationship_Views>
        <Relationship_View_ID Ordinal="Primary">1000</Relationship_View_ID>
      </Relationship_Views>
      <Relationship_Target_Form>Weakness</Relationship_Target_Form>
      <Relationship_Nature>ChildOf</Relationship_Nature>
      <Relationship_Target_ID>694</Relationship_Target_ID>
      <!--Use of Multiple Resources with Duplicate Identifier-->
    </Relationship>
    <Relationship>
      <Relationship_Views>
        <Relationship_View_ID Ordinal="Primary">734</Relationship_View_ID>
      </Relationship_Views>
      <Relationship_Target_Form>Category</Relationship_Target_Form>
      <Relationship_Nature>ChildOf</Relationship_Nature>
```



# APIs vs. Websites



- In the previous example, the same data is returned from a website and the corresponding API, except one is pretty for normal humans using a browser and the other is pretty for developers and machines



# API Security Principles



- Common website security checks can also be used for APIs
- Ensure well-known and secure protocols are being used
- Secure code reviews – API code should be reviewed for vulnerabilities and best practices before being put into production
- Vulnerability scanning – Regular automated scanning of servers and applications
- Data returned can sometimes be sensitive – treat it as such, limit access to specific URLs and data when necessary





# Resources To Secure an API



- Automated web scanning tools
  - Website vulnerability scanners
  - Server-side scanners
  - Manual checks for vulnerabilities
- Secure code reviews of API functionality can be (and often are) conducted by the same team as website code reviews
  - From a coding perspective, API code is very similar to website code, technologies used are identical and sometimes even coupled with the website

# Resources to Secure an API



- Good News! If an effective website security program already exists, very few additional resources are needed to secure an API
- Due to the similarities of APIs and websites, a website security team can easily handle API security
- As no new security technologies need to be adopted, the costs of API security to an organization are minimal

# Contact Us



For more information please contact:

Mr. Alejandro Caceres, CNO Engineer

Phone: 304-989-1431

Email: [alejandro.caceres@lunarline.com](mailto:alejandro.caceres@lunarline.com)

Mr. Waylon Krush, CEO

Phone: 571-481-9300 or Mobile: 703-200-8123

Email: [waylon.krush@lunarline.com](mailto:waylon.krush@lunarline.com)

Keith Mortier, President & COO

Phone: 571-481-9302 or Mobile: 571-236-4190

Email: [keith.mortier@lunarline.com](mailto:keith.mortier@lunarline.com)



Small Business  
Solutions



ISO 9001:2008 ♦ Maturity Level 2 of CMMI® ♦ NSA/CNSS Approved Training ♦ VA Certified SDVOSB