CWE REST API Working Group Meeting – May 19, 2022

1) Call to order
2) Agenda approval
3) Minutes approval
4) Regular order
   - Note – email list reflector: CWE-CAPEC REST API Working Group cwe-capec-rest-api-working-group-list@mitre.org
   - Note – GitHub: https://github.com/CWE-CAPEC/REST-API-wg
   - Note – Swagger (API Documentation & Design Tools for Teams | Swagger)
   - Note – GraphQL and GraphQL Playground (https://graphql.org/)
   - Note – JSON (schema example: https://github.com/oasis-open/cti-stix2-json-schemas)
   - DB/tool chain
     - Examples of schema changes (Recent HW CWE Schema Extension)
     - CWE/CAPEC entry with "overlay" of terminology for database items/properties
5) Action item review
   - Principals – security
   - Metrics on API use, yet protect security of users' information
     - Does MITRE consider analytics to be especially critical
     - Suppose focusing on characteristics of behavior as opposed to individual token tracking
     - Datadog – tracking performance problems
     - OWASP based scanning of the API, from development
       - Can we standardize methods to follow while developing the API [Application Security]
       - ^MITRE does app security reviews internally, no intention to discuss in working group
   - Discuss what CWE data response content looks like and whether MITRE needs to try and make it harder for folks to XSS themselves.
     - Just add warning that content in JSON makes no guarantees
   - Discuss presence of XHTML in various elements
     - DemEX – highly structured so proper rendering can be important
   - Means to restrict use (to avoid DoS issues, etc.) to registered users; API keys/tokens?
     - Not a massive amount of data, can do blind rate limiting – oppose custom built version
     - Authentication side
   - EOL of API versions?
     - And the communication therof
   - Communication?
     - Incidents or Changes (any)
   - Original larger goal was API versioning
   - GraphQL vs REST poll results
6) Adjourn

Rapid fire notes following: “Examples of schema changes (Recent HW CWE Schema Extension)”

Regarding enums – it is tough to track verbiage changes in leaf level enumerations. Ken – wouldn’t have to be tracked as schema change. Checking for specific enumeration values is generally brittle.

No matter what, client will have to adjust behavior.

Mapped as DB, wouldn’t change schema any time a row is added.

Suppose someone uses a bunch of components which are some TechName, could look up by tech name. Ken replies with doing mapping between versioned values in this case. Dave replies that no “Accelerator IP” remain after. Suppose that hard id’s to allow simple renaming.

Support schemas with availability window.

DB metaphor – some schema material is rows – storing in DB and converting to XML shouldn’t be terrible

MITRE generates encoding for enumerations automagically

How to communicate changes properly. And identify internally how to return appropriate forward mapped content.

Return latest available at that schema

Luke to return with an explanation of what the CWE XML schema has as required and optional

diff --git a/master_files/XSD/cwe_schema_v6.7.xsd b/master_files/XSD/cwe_schema_v6.7.xsd
index
--- a/master_files/XSD/cwe_schema_v6.7.xsd
+++ b/master_files/XSD/cwe_schema_v6.7.xsd
@@ -1080,72 +1080,72 @@
<xs:restriction base="xs:string">
<xs:enumeration value="Web Server"/>
<xs:enumeration value="Database Server"/>
- <xs:enumeration value="Accelerator IP">
+ <xs:enumeration value="Accelerator">

+ <xs:enumeration value="Accelerator">
<xs:annotation>
- <xs:documentation>IP dedicated to offload a specific workload to enhance performance: DSP, packet processing, m
athematical, compression, etc.</xs:documentation>
+ <xs:documentation>hardware Intellectual Property (IP) dedicated to offload a specific workload to enhance perfo
rmance: DSP, packet processing, mathematical, compression, etc.</xs:documentation>
</xs:annotation>
</xs:enumeration>

```
======================
  <xs:simpleType name="TechnologyNameEnumeration">
    <xs:annotation>
      <xs:documentation>The TechnologyNameEnumeration simple type contains a list of values corresponding to different technologies. A technology represents a generally accepted feature of a system and often refers to a high-level functional component within a system.</xs:documentation>
      <xs:documentation>Within this context, "IP" stands for "Intellectual Property" and is the term used to distinguish unique blocks within a System on Chip, with each block potentially coming from a different source.</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Web Server"/>
      <xs:enumeration value="Database Server"/>
      <xs:enumeration value="Accelerator">
======================
      <xs:element name="Technology" minOccurs="0" maxOccurs="unbounded">
        <xs:complexType>
          <xs:attribute name="Name" type="cwe:TechnologyNameEnumeration"/>
          <xs:attribute name="Class" type="cwe:TechnologyClassEnumeration"/>
          <xs:attribute name="Prevalence" type="cwe:PrevalenceEnumeration" use="required"/>
        </xs:complexType>
      </xs:element>

<Weakness ID=”some_number”>
        <Description>Something New</Description>
        <Technology Name=”Accelerator”>
+.      <Technology Name=”Sensor”>
```