Hackathon, v105

• Bill Munyan
• Carl-Heinz Genzel (remotely)
• Henk approves of this hackathon
Objectives

• Determine a data model representing “what to collect”
• Determine a data model representing “what was collected”
• Implement a simple collector
• Do cool things with XMPP
  • Use XMPP’s eXtensible <iq> stanza to orchestrate collection
  • Use various XMPP features (IQ’s and PubSub) to push collected information between XMPP entities
• Do cool things with concise map
  • Translate collected information to MAP CBOR data
  • Publish translated CBOR data to MAP
  • Extract CBOR data from MAP and reconstruct collected XML data
Data Model(s)

• Prior to hackathon, Bill worked on some (quick & dirty) modifications to OVAL
  • Current OVAL structure couples collection and evaluation
  • There’s no way to indicate collection only

• Redefined some namespaces, and created a new “OVAL collections” schema
  • Allows for OVAL XML to only specify collection activities
  • The <oval_objects> element
  • Reduced scope (for this hackathon) to just the core and platform-independent schemas
<?xml version="1.0" encoding="UTF-8"?>
<oval_objects collection-id="oval.org.cisecurity:collection:9999"
    xmlns="http://oval.cisecurity.org/XMLSchema/oval-collections-6"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ind-def="http://oval.cisecurity.org/XMLSchema/oval-definitions-6#independent"
    xmlns:oval="http://oval.cisecurity.org/XMLSchema/oval-common-6"
    xmlns:oval-coll="http://oval.cisecurity.org/XMLSchema/oval-collections-6"
    xmlns:ind-coll="http://oval.cisecurity.org/XMLSchema/oval-collections-6#independent"
    <generator>
        <oval:product_name>OVAL Collections Generator</oval:product_name>
        <oval:product_version>0.0.1</oval:product_version>
        <oval:schema_version>6.0.0</oval:schema_version>
        <oval:timestamp>2019-07-20T10:41:00-05:00</oval:timestamp>
    </generator>
    <objects>
        <ind-coll:family_object id="oval.org.cisecurity:obj:1" version="1"
            comment="This family object represents the family that the operating system belongs to."/>
        <ind-coll:environmentvariable_object id="oval.org.cisecurity:obj:2" version="1" comment="The COMPUTERNAME environment variable">
            <ind-coll:name>COMPUTERNAME</ind-coll:name>
        </ind-coll:environmentvariable_object>
    </objects>
</oval_objects>
  <generator>
    <product_name xmlns="http://oval.cisecurity.org/XMLSchema/oval-common-6">OVAL XMPP</product_name>
    <product_version xmlns="http://oval.cisecurity.org/XMLSchema/oval-common-6">0.0.1</product_version>
    <schema_version xmlns="http://oval.cisecurity.org/XMLSchema/oval-common-6">6.0.0</schema_version>
  </generator>
  <system_info>
    <os_name>Windows 10</os_name>
    <os_version>10.0</os_version>
    <architecture>x64</architecture>
    <primary_host_name>CIS-CAT-DEV</primary_host_name>
    <interfaces/>
  </system_info>
  <collected_objects>
    <collected_object id="oval.org.cisecurity:obj:1" version="1" comment="This family_object represents the family that the operating system belongs to." flag="complete">
      <reference item_ref="1"/>
    </collected_object>
    <collected_object id="oval.org.cisecurity:obj:2" version="1" comment="The HOME environment variable" flag="complete">
      <reference item_ref="2"/>
    </collected_object>
  </collected_objects>
  <system_data>
    <family_item xmlns="http://oval.cisecurity.org/XMLSchema/oval-system-characteristics-6#independent" id="1">
      <family datatype="string">Windows</family>
    </family_item>
    <environmentvariable_item xmlns="http://oval.cisecurity.org/XMLSchema/oval-system-characteristics-6#independent" id="2">
      <name>COMPUTERNAME</name>
      <value>CIS-CAT-DEV</value>
    </environmentvariable_item>
  </system_data>
</oval_system_characteristics>
Who did what?

• Bill was again joined (remotely from Germany) by **Carl-Heinz**
  • CH is a MAP ninja and a java wizard

• Bill
  • Create/Enable XMPP extensions to handle collection requests (OVAL objects) and collection results (OVAL system characteristics)
  • Trigger collection through XMPP IQ stanzas
  • Collect Items at an endpoint via OVAL collection implementation
  • Push collected OVAL system characteristics to CH (2 methods)
    • Publish OVAL system characteristics to XMPP pub/sub topic
    • Enable OVAL system characteristics to be sent directly to CH via XMPP IQ stanzas

• Carl-Heinz
  • Receive collected system characteristics via XMPP adapter (pub/sub, <iq>, <message>)
  • Translate OVAL system characteristics to MAP CBOR data
  • Publish translated CBOR data to MAP
  • Search via MAP Client for Data
  • Translate Data from MAP to XML and see if it is the same as original OVAL Results
Who did what? Bill Edition

• Create/Enable XMPP extensions to handle collection requests (OVAL objects) and collection results (OVAL system characteristics)
• Trigger collection through XMPP IQ stanzas
• Collect Items at an endpoint via OVAL collection implementation
• Push collected OVAL system characteristics to CH (3 methods)
  1. Publish OVAL system characteristics to XMPP pub/sub topic
  2. Enable OVAL system characteristics to be sent directly to CH via XMPP <iq> stanzas
  3. Enable OVAL system characteristics to be sent directly to CH via XMPP <message> stanzas
Workflow 1: Pub/Sub

1. IQ request containing `<oval_objects>`
2. Orchestrator publishes system characteristics
3. XMPP Adapter receives subscription event containing system characteristics
4. XMPP Adapter is subscribed to topic
5. IQ response containing `<oval_system_characteristics>`
6. MAP Client translates system characteristics XML to CBOR and publishes to the MAP Server
7. MAP Server
Workflow 2: Direct XMPP <iq>
Workflow 3: Direct XMPP 

Workflow 3: Direct XMPP using Message stanzas

- Orchestrator

IQ request containing <oval_objects>

IQ response containing <oval_system_characteristics>

- Collector

Collector sends <message> stanza containing system characteristics directly to XMPP Adapter

XMPP Adapter

MAP Client translates system characteristics XML to CBOR and publishes to the MAP Server

MAP Server

• Subscribe to XMPP pub/sub topic to receive collected system characteristics
• Translate OVAL system characteristics to MAP CBOR data
• Publish translated CBOR data to MAP
• Search via MAP Client for Data
• Translate Data from MAP to XML and see if it is the same as original OVAL Results
Storage: Concise Map

IETF 105 Hackathon
Concise MAP implementation for collected system characteristics

Assoc. Generator (Metadata)
- System (Identifier)
  - Hostname: host.abc.com
  - Architecture: x86_64

Current Value (Metadata)
- Value: CIS-CAT-BEV

Generator (Identifier)
- ProductName: Something
- ProductVersion: 6

Installed OS (Metadata)
- Version: 02020220
- Arch: x86_64 + Gibbon
- Name: Ubuntu

Environment Variable (Identifier)
- Name: COMPUTERNAME

Reference (Metadata)
- ID: oval.org.cisecurity:collection:9999
- Type: Collection

OS (Identifier)
- Family: Linux

Identifier (Identifier)
- ID: 1
  - Type: Item

Identifier (Identifier)
- ID: 2
  - Type: Item

Reference (Metadata)
- Reference (Metadata)
- Reference (Metadata)
Translation to CBOR

This is the CBOR data in pretty printing aka. human readable format.
"1": [21911, 0, 9, 1] is a name of the data structure surrounded by [] in this case for the publish operation.
1204 Means a list of metadata.
1203 Means a list of one or two (in case of a link between) identifiers.
Identifiers usually have an attribute numbered 3, which is the payload of the identifier.
Metadata usually have an attribute numbered 7, which is the payload of a metadata.

Refer to the CDDL files in the SACM repository for further explanations about the numbers.

```
{
    "1": [21911, 0, 9, 1], // Name for the publish operation
    "2": "26be0140-6cd5-4436-9714-991e1700ac9",
    "3": [
        { "1204": [
            { "1": [65535, 65535, 1, 0],
                "2": 1,
                "3": 0
            }
        ],
        "1203": [
            { "1": [65535, 65535, 2, 0],
                "2": "",
                "3": { "0": "CIS-CAT-DEV",
                        "1": "amd64"
                }
            },
            { "1": [65535, 65535, 2, 0],
                "2": "",
                "3": { "0": "OWAL XMPP",
                        "1": "0.0.1",
                        "2": ["null:6.0.0"],
                        "3": "2019-07-21T15:46:53.575Z"
                }
            }
        ]
    }
}
```

“1203” Identifies a link between System Identifier and Generator

System Identifier

Generator
Things we Learned

• We were able to move data between components using 3 methods supported by XMPP
  • Publish/Subscribe
  • Direct messaging via <iq> stanzas containing custom payload
  • Direct messaging via <message> stanzas containing custom payload

• Carl-Heinz was also able to query the MAP data and reconstruct the OVAL system characteristics from it.
  • May enable downstream operations if they require the XML data

• Right now there's no way to configure a MAP client with the things you want to know. There's only pre-configured clients.
  • Meaning, specific CDDL and implementation was required before the MAP client could translate the system characteristics.
  • This could fall under capability discovery, i.e. “what specific system characteristics can my MAP client handle?”
What’s next?

• Keep Going!
  • As we define operations, start to include them as part of the architecture draft and build a library of data models
  • Can we define a core set of operations, and build upon them with extension points?
    • (use XMPP as an example – there’s only 3 core operations, and XEPs build upon that)
    • Can we define an “architecture core” and enable “SACM extension protocols”?

• Continue to refine the OVAL collection models
  • More platform-specific schema migrations
  • Propose to SCAP 2.0 working groups (endpoint data collection, OVAL)

• Build more collection capabilities based on the models
• Evaluation (or other downstream) operations
Thanks

• **BIG THANKS** to Carl-Heinz for his contributions to the Hackathon
  • Especially for staying awake into the wee hours of the morning

• Also thanks to Henk for enabling collaboration between Bill and Carl-Heinz, and getting us going on various calls before Hackathon