Architecture Update

M & M (not Eminem)

Virtual Interim

May 1, 2020

Draft Updates (in progress)

- We’re still working on the -05 update.
  - Removing the XMPP-specific and “solution”-ey appendices
  - Moving section on “Security Program Workflows” to an appendix

- “This is how you collect state” and “This is how you evaluate state”
  - If collection/evaluation of state are standardized, they can be applied to the security program workflows. That will be illustrated in the appendix.

Collection

- The draft will discuss 3 classifications of collection
  - Ad-Hoc
    - Single collection of posture attributes; a snapshot in time
  - Continuous/Scheduled
    - Ongoing, periodic collection of posture attributes
  - Observational/Event-based
    - Collection triggered by an observation, external to an endpoint, of events occurring on that endpoint.

Interactions

- Following the descriptions of the various roles in Collection/Evaluation sub-architectures, we're adding Interactions

- Interaction Categories
  - Broadcast (Pub/Sub)
  - Directed (Point-to-Point)
    - Synchronous: Request/Response
    - Asynchronous: Request, including callbacks to handle eventual Response

Collector Onboarding/Registration

- Describe how a Posture Collection Service (a collector) becomes part of the ecosystem
  - Registration with the Orchestrator
  - Advertisement of capabilities
  - Establish an “administrative interface”
  - Subscription to relevant collection topics

Collector Onboarding/Registration

- PCS authenticates to the integration service
- PCS initiates registration per the described “taxonomy”
  - “Taxonomy” is a new section we’re adding, standardizing the different actions, integration service topics, payloads in a “convention over configuration” way of thinking.
  - PCS sends a payload to `/orchestrator/pcs/registration`
  - Orchestrator generates unique identifier for the registering PCS
  - Orchestrator establishes “administrative interface” using the `/orchestrator/pcs/{pcs_unique_id}` topic.
  - Orchestrator sends a response payload to PCS indicating its assigned unique identifier (pcs_unique_id)

Collector Onboarding/Registration

- PCS receives response containing {pcs_unique_id}
- PCS initiates “capability advertisement handshake”
  - PCS directs a payload over the administrative interface: `/orchestrator/pcs/{pcs_unique_id}`
  - Payload contains PCS’ collection capabilities
    - How it advertises/formats this TBD
  - The Orchestrator receives these collection capabilities and responds with the list of topics to which the collector should subscribe
  - PCS subscribes to each of the prescribed topics

Collection Interactions

- Initiate Ad-Hoc Collection
- Coordinate Periodic Collection (Schedule, Cancel)
- Coordinate Observational/Event-based Collection (Initiate, Cancel)
- Persist Collected Posture Attributes
  - Data normalization

Evaluation Interactions

- Initiate Ad-Hoc Evaluation
- Coordinate Periodic Evaluation (Schedule, Cancel)
- Coordinate change-based Evaluation
  - For example, if a posture attribute in the repository is changed, trigger evaluation of particular policy items

Taxonomy

- Conventions for interactions
  - Interaction Name and Description
  - Interacting Parties
  - Topic
  - Request Payload
  - Response Payload

- We have some to add
  - PCS registration
  - Orchestrator-to-Collector Administrative Interfaces
  - Publishing collection instructions

Moving Ahead…

- Get the -05 out there
- We definitely need to start/continue/move on the discussion of information model/data models/normalization

The Questions

- What really are the next steps and who can help?
- How much detail does this draft need to show?
  - What does the draft need to “prove” that it’s ready to move towards last call?
  - Who can help us answer that question?
  - How much information model is necessary?
- How much implementation do we need to prove this works?
- How do we make this a “living” document?
  - i.e., IANA tables for things like topic naming conventions