Vulnerability Assessment Scenario Follow-Up Work

IETF 98

03/30/2017

Agenda

- Components
- Interactions
- Tasks
- IE selection
- Next Steps

A few quick reminders

- Trying to clear roadblocks with the Architecture and scope of the IM
- Assuming a single flow through the assessment process
- Assuming components are pre-configured to talk with each other
- Assuming components are black boxes with interfaces and data model expectations

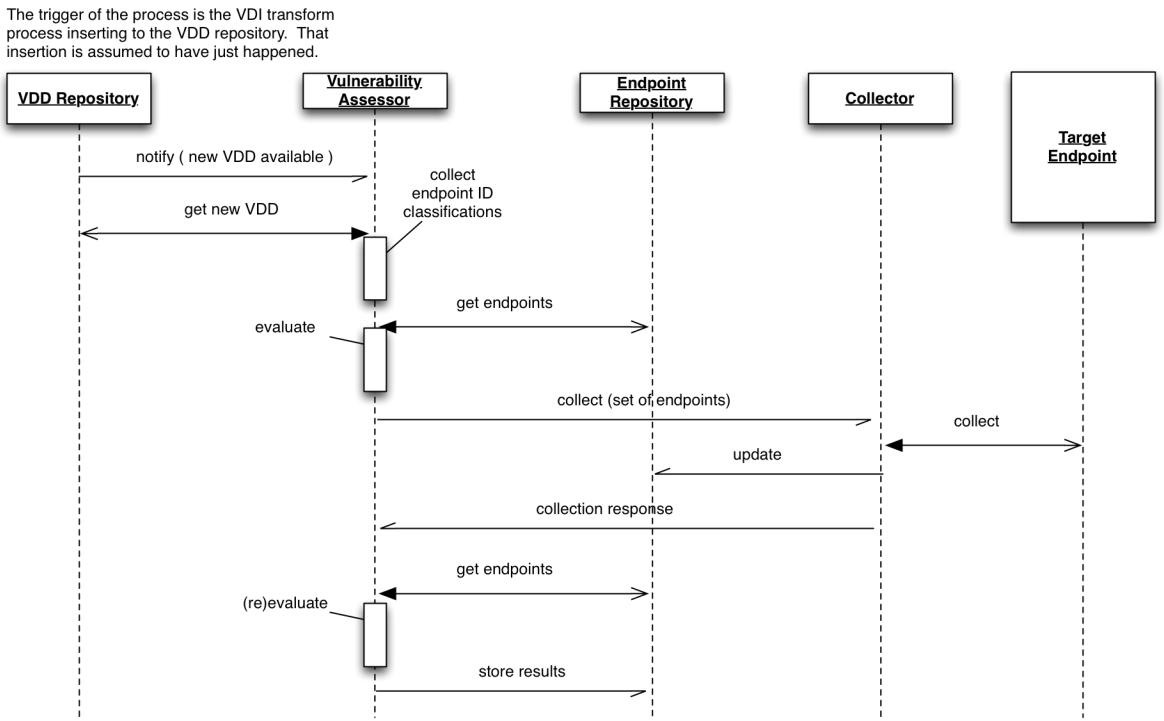
What we are working on

- Phase 1
  - Create a diagram that shows the components and their interactions with each other (distinguish between required and optional interactions)
  - Identify where different tasks occur in the diagram
- Phase 2
  - For each component, identify the IEs needed to carry out the required interactions and define the necessary interfaces and operations
  - Validate using CVEs

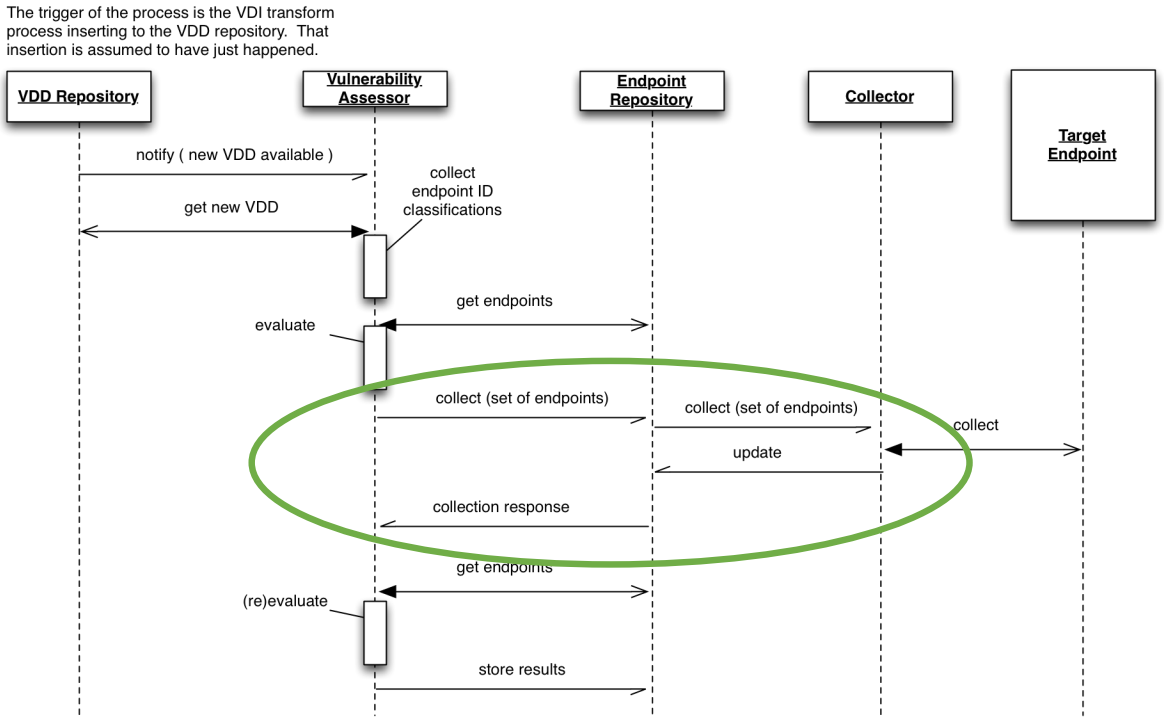
Components

- **Vulnerability Detection Data Repository:** controller that contains functions to consume, store, and provide vulnerability detection data
- **Vulnerability Assessor:** software that assesses the current state of a target endpoint against vulnerability detection data
- **Endpoint Repository:** controller that contains functions to consume, store, and provide information about target endpoints
- **Collector:** software that acquires information about target endpoints by conducting collection tasks
- **Target Endpoint:** endpoint under assessment
- **Assessment Results Repository:** controller that contains functions to consume, store, and provide information about assessment results

Component interactions diagram



Component interactions diagram (alternate)



Component-related issues

- Does the first "get endpoints" operation include the judgement of data? Or, is a separate operation needed?

- How to represent repositories (separate, combined, hybrid)?
  - There was some consensus that combining components would be feasible
  - We want to support any combination of components
  - Do we need different interfaces for different types of repositories?
  - If not, maybe we can simplify the workflow with a single repository?

Tasks I/O - collection

- Collect endpoint attributes about a target endpoint

- **Input:** collection guidance
  - Which target endpoint attributes to collect
  - Frequency when attributes are collected (ad-hoc, scheduled, continuously)
  - Method used to collect attributes (self-reporting, remote-acquisition, behavior-observation)

- **Output:** collection result

Tasks I/O - evaluation

- Comparison of endpoint attributes against a specified state

- **Input:** evaluation guidance
  - Which target endpoint attributes to evaluate and any requirements
  - Expected endpoint attribute values
  - Frequency when endpoint attributes are evaluated

- **Output:** evaluation result

Tasks I/O - storage

- Entering information into a repository

- **Input:** storage guidance
  - Which data to enter into the repository
  - Storage requirements (access control, retention period, etc.)

- **Output:** to be determined

Tasks I/O - query

- Retrieve data from a repository
- **Input:** query guidance
  - Which data to retrieve from the repository
  - Requirements for the data (is it fresh enough, etc.)
- **Output:** data retrieved from the repository

Tasks I/O - target endpoint characterization

- Continuously adding acquired endpoint attributes to a target endpoint characterization record

- **Input:** various security automation data
    - Discovered target endpoint attributes
    - Endpoint attribute collection results
    - Existing characterization records

- **Output:** target endpoint characterization records

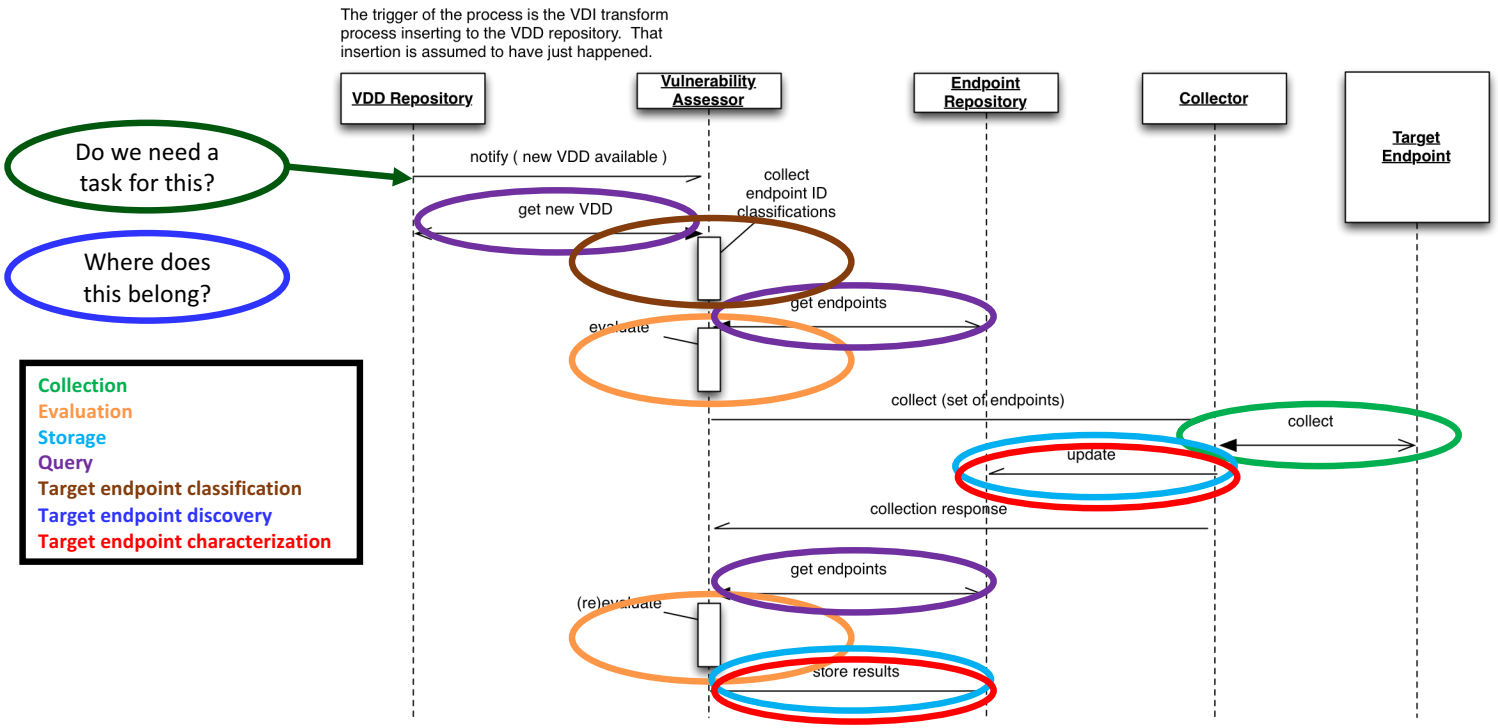
Tasks I/O - target endpoint classification

- Associating a class with an endpoint characterization record
- **Input:** various security automation data
  - Endpoint characterization records (without classification)
  - Classification guidance (how to classify a record)
- **Output:** endpoint characterization records (with classification)

Tasks I/O - target endpoint discovery

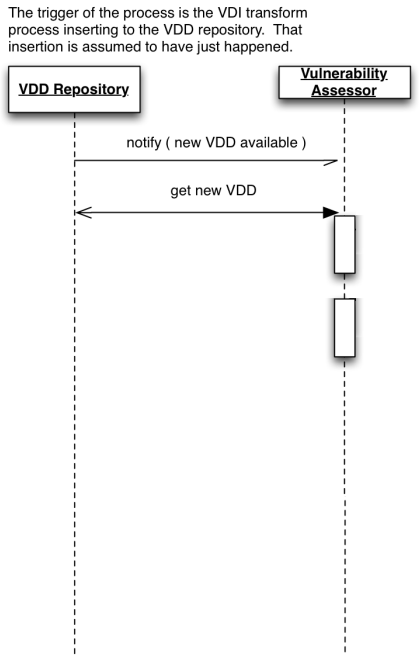
- Detecting previously unknown interactions of a potential target endpoint in the SACM domain
- **Input:** endpoint attributes acquired via local or remote interfaces
- **Output:** endpoint attributes including metadata such as data source or data origin

Component interactions diagram with tasks



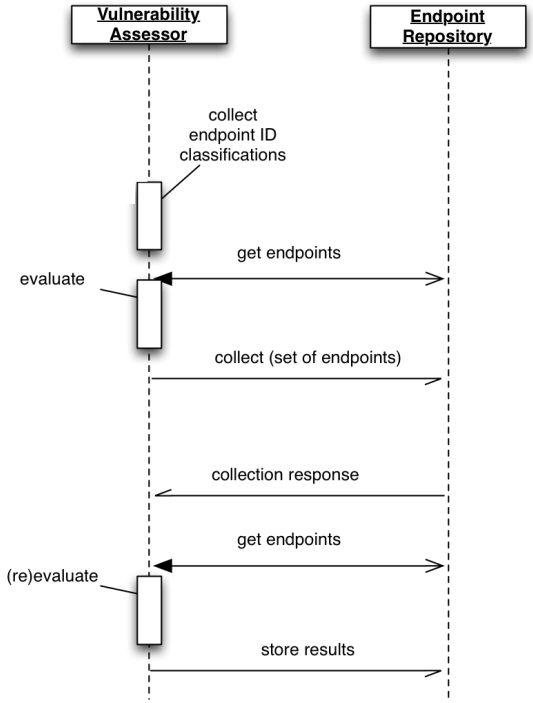
IE Selection - VDD repository and vulnerability assessor

- Ingest date
- Date of release
- Version
- External vulnerability ID
- Severity score
- Vulnerability description
- Vulnerability remediation

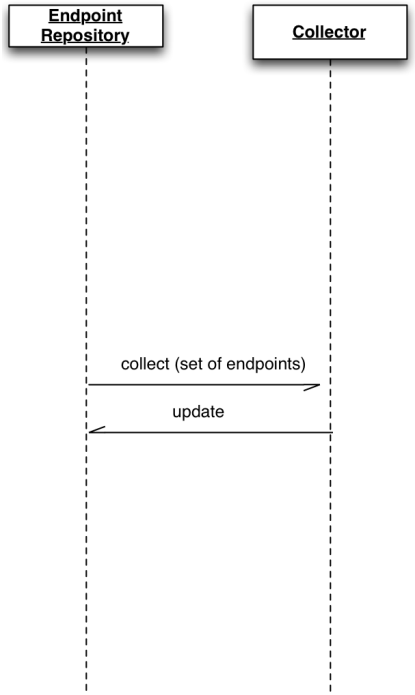


- Other IEs?
- ….

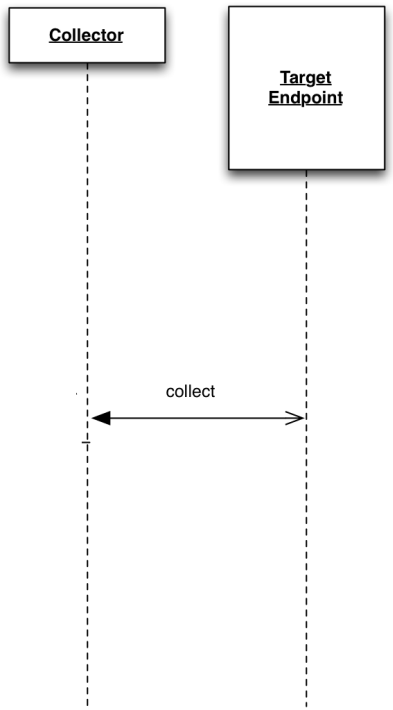
IE selection – Vulnerability assessor and endpoint repository



IE selection – Endpoint repository and collector



IE selection – Collector and target endpoint



Thoughts on a proof of concept

- Suggestion to develop a generic proof of concept implementation that provides the functions of SACM components (takes input and creates output)

- May be able to leverage existing solutions for this such as NEA, SWIMA, OVAL, NETCONF, etc.

- Would anyone be interesting in supporting this effort?

Next steps

- Reach consensus on components, interactions, and tasks on the mailing list
- Continue to enumerate the IEs necessary to carry out the required interactions
- Update wiki as needed

Objectives

- Identify the tasks that are necessary to execute the scenario and define their inputs and outputs
- Identify the interactions between components in the scenario and define the interfaces and operations required to support them
- Identify the IEs necessary to facilitate the communication of security automation data between components in the scenario
- Validate tasks, interfaces, operations, and IEs by exercising the scenario using multiple CVEs

Approach (2)

- Phase 3
  - Identify the IEs needed to carry out the optional interactions and define the necessary interfaces for those interactions
  - Validate using CVEs
- Phase 4
  - Add support for discovery of components and capabilities by other components
  - Validate using CVEs

Deliverables

- Updated Architecture that defines the interfaces and operations required for the scenario
- Updated IM that contains the minimum set of IEs necessary for the scenario
- Multiple examples that use CVEs to validate the scenario