Security Automation and Continuous Monitoring WG D. Waltermire
Internet-Draft NIST
Intended status: Informational D. Harrington
Expires: February 23, 2014 Effective Software
August 22, 2013

Using Security Posture Assessment to Grant Access to Enterprise Network Resources

draft-ietf-sacm-use-cases-00

Abstract

This memo documents a sampling of use cases for securely aggregating configuration and operational data and assessing that data to determine an organization's security posture. From these operational use cases, we can derive common functional capabilities and requirements to guide development of vendor-neutral, interoperable standards for aggregating and assessing data relevant to security posture.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 23, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 3
2. Requirements Language 4
3. Endpoint Posture Assessment 4
  3.1. Example - Departmental Software Policy Compliance 4
  3.2. Main Success Scenario 4
4. Use Cases 5
  4.1. Asset Management 5
    4.1.1. Asset Discovery 6
    4.1.2. Asset Identification 6
    4.1.3. Endpoint Components and Asset Composition 7
    4.1.4. Asset Characterization 7
    4.1.5. Asset Resources 9
    4.1.6. Asset Representation Reconciliation 9
    4.1.7. Asset Life Cycle 9
  4.2. Endpoint Configuration Management 9
    4.2.1. Organizing Configuration Metadata 10
    4.2.2. Publishing Recommended Configuration Posture 10
    4.2.3. Defining Organizationally Expected Configuration Posture 10
    4.2.4. Collecting Endpoint Configuration Posture 10
    4.2.5. Comparing Expected and Actual Configuration Posture . 10
    4.2.6. Examining configuration of logical to physical mappings 11
    4.2.7. Configuring Endpoint Interfaces 11
  4.3. Endpoint Posture Change Management 11
    4.3.1. Defining and Exchanging Baselines 11
    4.3.2. Detecting Unauthorized Changes 11
      4.3.2.1. Endpoint Addressing Changes 11
      4.3.2.2. Service Authorization Changes 11
      4.3.2.3. Dynamic Resource Assignment Changes 11
      4.3.2.4. Security Authorization Status Changes 12
  4.4. Security Vulnerability Management 12
    4.4.1. Example - NIDS response 12
    4.4.2. Example - Historical vulnerability analysis 13
    4.4.3. Source Address Validation 13
  4.5. Data Collection 13
  4.6. Assessment Result Analysis 13
  4.7. Content Management 14
5. IANA Considerations 14
6. Security Considerations 15
7. Acknowledgements 15

8. Change Log 15
  8.1. draft-waltermire-sacm-use-cases-05 to draft-ietf-sacm-use-cases-00 15
  8.2. -04- to -05- 16
9. References 17
  9.1. Normative References 17
  9.2. Informative References 17
Authors' Addresses 18

1. Introduction

Our goal with this document is to improve our agreement on which problems we're trying to solve. We need to start with short, simple problem statements and discuss those by email and in person. Once we agree on which problems we're trying to solve, we can move on to propose various solutions and decide which ones to use.

This document describes example use cases for endpoint posture assessment for enterprises. It provides a sampling of use cases for securely aggregating configuration and operational data and assessing that data to determine the security posture of individual endpoints, and, in the aggregate, the security posture of an enterprise.

These use cases cross many IT security information domains. From these operational use cases, we can derive common concepts, common information expressions, functional capabilities and requirements to guide development of vendor-neutral, interoperable standards for aggregating and assessing data relevant to security posture.

Using this standard data, tools can analyze the state of endpoints, user activities and behaviour, and assess the security posture of an organization. Common expression of information should enable interoperability between tools (whether customized, commercial, or freely available), and the ability to automate portions of security processes to gain efficiency, react to new threats in a timely manner, and free up security personnel to work on more advanced problems.

The goal is to enable organizations to make informed decisions that support organizational objectives, to enforce policies for hardening systems, to prevent network misuse, to quantify business risk, and to collaborate with partners to identify and mitigate threats.

It is expected that use cases for enterprises and for service providers will largely overlap, but there are additional complications for service providers, especially in handling information that crosses administrative domains.

The output of endpoint posture assessment is expected to feed into additional processes, such as policy-based enforcement of acceptable state, verification and monitoring of security controls, and compliance to regulatory requirements.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Endpoint Posture Assessment

Endpoint posture assessment involves collecting information about the posture of a given endpoint. This posture information is gathered and then published to appropriate data repositories to make collected information available for further analysis supporting organizational security processes.

Endpoint posture assessment typically includes:

- Collecting the posture of a given endpoint;

- Making that posture available to the enterprise for further analysis and action; and

- Assessing that the endpoint's posture is in compliance with enterprise standards and policy.

3.1. Example - Departmental Software Policy Compliance

In order to meet compliance requirements and ensure that corporate finance information is not revealed improperly, all computers in the finance department of Example Corporation are required to run only software contained on an approved list and to be configured to download and install software patches every night. Each computer is checked to make sure it complies with this policy whenever it connects to the network and at least once a day thereafter. These daily compliance checks assess the posture of each computer and report on its compliance with policy.

3.2. Main Success Scenario

1. Define a target endpoint to be assessed

2. Select acceptable state policies to apply to the defined target

3. Identify the endpoint being assessed

4. Collect posture attributes from the target

5. Communicate target identity and collected posture to external system for evaluation

6. Compare collected posture attributes from the target endpoint with expected state values as expressed in acceptable state policies

4. Use Cases

The use cases defined in this section support assessing endpoint posture in an automated manner as described in Section Section 3. The following sub-sections describe use cases broken out by their corresponding IT decipline.

4.1. Asset Management

Organizations manage a variety of assets within their enterprise including: endpoints, the hardware they are composed of, installed software, hardware/software licenses used, and configurations.

Managing endpoints and the different types of assets that compose them involves initially discovering and characterizing each asset instance, and then identify them in a common way. Characterization may take the form of logical characterization or security characterization, where logical characterization may include business context not otherwise related to security, but which may be used as information in support of decision making later in risk management.

Coverage involves understanding what and how many assets are under control. Assessing 80% of the enterprise assets is better than assessing 50% of the enterprise assets.

Getting asset details can be comparatively subtle - if an enterprise does not have a precise understanding of its assets, then all acquired data and consequent actions taken based on the data are considered suspect.

Assessing assets (managed and unmanaged) requires that we have visibility into the posture of endpoints, the ability to understand the composition and relationships between different assets types, and the ability to properly characterize them at the outset and over time.

The following list details some requisite Asset Management capabilities:

- Discover assets in the enterprise

- Identify and describe assets using a common vocabulary between implementations

- For a given endpoint, understand the composition and relationship of its constituent assets

- Characterize assets according to security and non-security asset properties

- Reconcile asset representations originating from disparate tools

- Manage asset information throughout the asset's life cycle

4.1.1. Asset Discovery

Many network management systems periodically test for the presence of endpoints or interfaces in a network, including discovering endpoints that have suddenly appeared in a network that are not authorized to be part of the network. Other approaches can be used to identify new endpoints as they connect to the network alowing for authentication and authorization to occur dynamically as part of a network access control decision. There are many layers of endpoints, and many standardized information models for determining endpoints in a network.

These standardized collections include ARP tables [RFC0826], Interface tables such as the Interfaces MIB (IF-MIB) [RFC2863] or the YANG module ietf-interfaces , Link Layer Discovery tables [RFC2922], DHCP tables (Ref:???), and so on.

4.1.2. Asset Identification

Identifying assets is critical for managing information provided about and collected from endpoints. It is important to have stable mechanisms for identifying assets over time to allow asset information to be correlated. It is often possible to use standardized and proprietary identification mechanisms provided by hardware and software asset vendors (e.g., CPU identifiers, product tags). In some cases these identifiers may be stable for the life of the hardware component. In other cases (e.g., MAC addresses), the identifier may be mutable within software. Organizationally provided identifiers can also be used to identify assets such as those provided by hardware and software certificates, and configurable identification sources. In other cases it may only be possible to identify an asset by the network addressing information it is currently using, requiring additional context to correlate asset

information across multiple network connection sessions. In an enterprise context it is often necessary to use multiple identification viewpoints for an asset to correlate data generated from endpoint, network, and human sources.

Some standards focus on identifying the hardware and the system software. For example, the SYSTEM-MIB [RFC1213] contains a description of the endpoint, an authoritative identifier of the type of endpoint assigned by the vendor of the endpoint, an administrative name for the endpoint, plus the endpoint's contact person, the location of the endpoint, system time, and an enumerator that identifies the layer of services provided by the endpoint. The system description includes the vendor, product type, model number, OS version, and networking software version.

Similar information is available via the YANG module ietf-system . This module includes data node definitions for system identification, time-of-day management, user management, DNS resolver configuration, and some protocol operations for system management.

4.1.3. Endpoint Components and Asset Composition

It can be important to characterize the components of an endpoint, including physical and logical components, and the relationships between the components, such as containment of components within other components, or mappings between logical entities and the physical entities used to instantiate them. The information about the physical entities might include manufacturer-assigned serial number, manufacture date, an asset identifier for the component, and so. Logical entities may be defined, and associated with the physical entities using a mapping table.

Example standardized data models include the ENTITY-MIB [RFC6933] the Q-BRIDGE-MIB MIB [RFC4363] and the MIB for Virtual Machines Controlled by a Hypervisor .

4.1.4. Asset Characterization

It is necessary to collect, store, manage, and exchange a variety of different asset characteristics that provide additional context that is useful to support automated and human decision making as part of operational and security processes. Often this information helps to bridge automated and human-oriented processes. In many cases it is impractical or infeasible to collect specific asset details using technical data collection mechanisms.

Asset characteristics can take many forms depending on the asset type.

For hardware assets the following are often useful characteristics:

- Manufacturer
- Production version
- Hardware characteristics (e.g., memory, storage, network interfaces)
- End-of-support dates

For software assets the following are often useful characteristics:

- Software version
- Supported hardware platforms
- Metadata identifying: product family, software function, edition, licensing
- Other software dependencies
- End-of-support dates

For managed endpoints, hardware, and software the following are often useful characteristics:

- Owning organization
- Responsible organizations and individuals (e.g., operations, security, inventory management)
- Assigned location for physical devices
- Location within network(s)

This information is important to provide additional context for supporting management of assets using human and automated processes. For example, it may be possible to automate assessing that an endpoint is out of compliance with organizational configuration guidelines, but additional information is needed to determine who to notify to correct the configuration. Information provided by asset characterization will enable notifications to be sent, trouble tickets to be generated, or specific reports to be generated at a dashboard for a systems administrator.

[TODO: Do we need more document characteristics or more examples?.]

4.1.5. Asset Resources

This type of asset characterization describes the resources of an endpoint, such as installed software, running software, software versions, processes, user sessions, devices (processors, disks, printers, network interfaces, etc.). This might also provides monitoring of performance and error states for the related resources.

[TODO: Its not clear if this is asset characterization or data collection. One way to look at asset characterization is that it is metadata that is provided by humans. Endpoint data collection is information provided by machines. The previous list looks like it is better oriented in the "machine" category. Do we want to move these examples to a different sub-section?]

An example is the HOST-RESOURCES-MIB [RFC2790]

4.1.6. Asset Representation Reconciliation

[TODO: We need to describe here how different asset identification viewpoints are reconciled (e.g., endpoint vs. network, passive vs. active]

4.1.7. Asset Life Cycle

[TODO: What do we want to say here?]

4.2. Endpoint Configuration Management

Organizations manage a variety of configurations within their enterprise including: endpoints, the hardware they are composed of, installed software, hardware/software licenses used, and configurations.

Security configuration management (SCM) deals with the configuration of endpoints, including networking infrastructure devices and computing hosts. Data will include installed hardware and software, its configuration, and its use on the endpoint.

[TODO: While some configuration settings might not be considered security relevant, it is not always possible to draw a clear distinction between security and non-security settings (e.g., power saving features). Do we want to make a distinction between security and non-security configuration settings?]

The following list details some requisite Configuration Management capabilities:

- [todo]

4.2.1. Organizing Configuration Metadata

Configuration metadata supports tooling helping organizations understand what configuration they should implement, using specific configuration values.

Enable data repositories containing machine-represtations of:

Configuration scoring: Characterizations of the relative security value of dsscrete configuration settings and specific values

Configuration dependencies (e.g., lists of settings, associated software, pre-requisite configurations)

Control catalog mappings supporting compliance [todo: in scope?]

4.2.2. Publishing Recommended Configuration Posture

Provide a mechanism for vendors and organizations to exchange machine-oriented descriptions of recommended configuration setting for software products. Enable organizations to apply recommended settings as expected configuration posture. Enable association of data-driven collection instructions using appropriate formats.

4.2.3. Defining Organizationally Expected Configuration Posture

Provide a mechanism for organizations to define and exchange expected configuration posture including: authorized software and associated configuration settings.

[TODO: Should software installation posture be defined seperately?]

4.2.4. Collecting Endpoint Configuration Posture

Enable collection and exchange of actual configuration posture including: installed software and values for configured settings.

[TODO: Should collecting software installation posture be defined seperately?]

4.2.5. Comparing Expected and Actual Configuration Posture

Enable evaluation of actual configuration posture against expected configuration posture. Generate a machine-oriented description of conformant and non-conformat posture including software inventory and configuration values.

[TODO: Should collecting software installation posture be defined seperately?]

[TODO: Examining software version configuration - Example - HOST-RESOURCES-MIB

4.2.6. Examining configuration of logical to physical mappings

[TODO: not sure what this is? Is it in scope?]

Example - ENTITY-MIB

4.2.7. Configuring Endpoint Interfaces

[TODO: not sure what this is? Is it in scope?]

Example - YANG module ietf-interfaces

4.3. Endpoint Posture Change Management

Organizations manage a variety of changes within their enterprise including: [todo]

The following list details some requisite Change Management capabilities:

- [todo]

4.3.1. Defining and Exchanging Baselines

[todo]

4.3.2. Detecting Unauthorized Changes

[todo]

[todo: figure out where these need to go]

4.3.2.1. Endpoint Addressing Changes

Example - DHCP addressing

4.3.2.2. Service Authorization Changes

Example - RADIUS network access

4.3.2.3. Dynamic Resource Assignment Changes

Example - NAT logging

4.3.2.4. Security Authorization Status Changes

Example - SYSLOG Authorization messages. SYSLOG [RFC5424] includes facilities for security authorization messages. These messages can be used to alert an analysts that an authorization attempt failed, and the analyst might choose to follow up and assess potential attacks on the relevant endpoint.

4.4. Security Vulnerability Management

Vulnerability management involves identifying the patch level of software installed on the device and the identification of insecure custom code (e.g. web vulnerabilities). All vulnerabilities need to be addressed as part of a comprehensive risk management program, which is a superset of software vulnerabilities. Thus, the capability of assessing non-software vulnerabilities applicable to the system is required. Additionally, it may be necessary to support non-technical assessment of data relating to assets such as aspects related to operational and management controls.

policy attribute collection

The following list details some requisite Vulnerability Management capabilities:

- o Collect the state of non-technical controls commonly called administrative controls (i.e. policy, process, procedure)
- o Collect the state of technical controls including, but not necessarily limited to:
  - * Software inventory (e.g. operating system, applications, patches)
  - * Configuration settings

4.4.1. Example - NIDS response

1. An organization's Network Intrusion Detection System detects a suspect packet received by an endpoint and sends an alert to an analyst. The analyst looks up the endpoint in the asset inventory database, looks up the configuration policy associated with that endpoint, and initiates an endpoint assessment of installed software and patches on the endpoint to determine if the endpoint is compliant with policy.

The analyst reviews the results of the assessment and takes action according to organization policy and procedures.

4.4.2. Example - Historical vulnerability analysis

When a serious vulnerability or a zero-day attack is discovered, one of the first priorities in any organization is to determine which endpoints may have been affected and assess those endpoints to try to determine whether they were compromised. Checking current endpoint state is not sufficient because an endpoint may have been temporarily compromised due to this vulnerability and then the infection may have removed itself. In fact, the vulnerable software may have been removed or upgraded since the compromise took place. And if the endpoint is still compromised, the malware on the endpoint may cause it to lie about its configuration. In this environment, maintaining historical information about endpoint configuration is essential. Such information can be used to find endpoints that had the vulnerable software installed at some point in time. Those endpoints can be checked for current or past indicators of compromise such as files or behavior linked to a known exploit for this vulnerability. Endpoints found to be vulnerable can be isolated to prevent infection while remediation is done. Endpoints believed to be compromised can be isolated for analysis and to limit the spread of infection.

4.4.3. Source Address Validation

Source Address Validation Improvement methods were developed to prevent nodes attached to the same IP link from spoofing each other's IP addresses, so as to complement ingress filtering with finer-grained, standardized IP source address validation. The framework document describes and motivates the design of the SAVI methods. Particular SAVI methods are described in other documents.

4.5. Data Collection

Central to any automated assessment solution is the ability to collect data from, or related to, an endpoint, such as the security state of the endpoint and its constituent assets.

So, is data collection a requirement or an architectural concept rather than a use case?

QUESTION: Understand more about what is meant by non-software vulnerabilities

4.6. Assessment Result Analysis

The data collected needs to be analyzed for compliance to a standard stipulated by the enterprise. Analysis methods may vary between enterprises, but commonly take a similar form.

The following capabilities support the analysis of assessment results:

- Comparing actual state to expected state
- Scoring/weighting individual comparison results
- Relating specific comparisons to benchmark-level requirements
- Relating benchmark-level requirements to one or more control frameworks

4.7. Content Management

The capabilities required to support risk management state measurement will yield volumes of content. The efficacy of risk management state measurement depends directly on the stability of the driving content, and, subsequently, the ability to change content according to enterprise needs.

Capabilities supporting Content Management should provide the ability to create/define or modify content, as well as store and retrieve said content of at least the following types:

- Configuration checklists
- Assessment rules
- Data collection rules and methods
- Scoring models
- Vulnerability information
- Patch information
- Asset characterization data and rules

Note that the ability to modify content is in direct support of tailoring content for enterprise-specific needs.

5. IANA Considerations

This memo includes no request to IANA.

6. Security Considerations

This memo documents, for Informational purposes, use cases for security automation. While it is about security, it does not affect security.

7. Acknowledgements

The National Institute of Standards and Technology (NIST) and/or the MITRE Corporation have developed specifications under the general term "Security Automation" including languages, protocols, enumerations, and metrics.

The authors would like to recognize and thank Adam Montville for his work on early edits of this draft. Additionally, the authors would like to thank Kathleen Moriarty and Stephen Hanna for contributing text to this document. The authors would also like to acknowledge the members of the SACM mailing list for their keen and insightful feedback on the concepts and text within this document.

8. Change Log

8.1. draft-waltermire-sacm-use-cases-05 to draft-ietf-sacm-use-cases-00

- Transitioned from individual I/D to WG I/D based on WG consensus call.
- Fixed a number of spelling errors. Thank you Erik!
- Added keywords to the front matter.
- Removed the terminology section from the draft. Terms have been moved to: draft-dbh-sacm-terminology-00
- Removed requirements to be moved into a new I/D.
- Extracted the functionality from the examples and made the examples less prominent.
- Renamed "Functional Capabilities and Requirements" section to "Use Cases".

  Reorganized the "Asset Management" sub-section. Added new text throughout.

  + Renamed a few sub-section headings.

  + Added text to the "Asset Characterization" sub-section.

- Renamed "Security Configuration Management" to "Endpoint Configuration Management". Not sure if the "security" distinction is important.

  * Added new sections, partially integrated existing content.

  * Additional text is needed in all of the sub-sections.

- Changed "Security Change Management" to "Endpoint Posture Change Management". Added new skeletal outline sections for future updates.

8.2. -04- to -05-

- Are we including user activities and behavior in the scope of this work? That seems to be layer 8 stuff, appropriate to an IDS/IPS application, not Internet stuff.

- I removed the references to what the WG will do because this belongs in the charter, not the (potentially long-lived) use cases document. I removed mention of charter objectives because the charter may go through multiple iterations over time; there is a website for hosting the charter; this document is not the correct place for that discussion.

- I moved the discussion of NIST specifications to the acknowledgements section.

- Removed the portion of the introduction that describes the chapters; we have a table of concepts, and the existing text seemed redundant.

- Removed marketing claims, to focus on technical concepts and technical analysis, that would enable subsequent engineering effort.

- Removed (commented out in XML) UC2 and UC3, and eliminated some text that referred to these use cases.

- Modified IANA and Security Consideration sections.

- Moved Terms to the front, so we can use them in the subsequent text.

- Removed the "Key Concepts" section, since the concepts of ORM and IRM were not otherwise mentioned in the document. This would seem more appropriate to the arch doc rather than use cases.

- Removed role=editor from David Waltermire's info, since there are three editors on the document. The editor is most important when one person writes the document that represents the work of multiple people. When there are three editors, this role marking isn't necessary.

- Modified text to describe that this was specific to enterprises, and that it was expected to overlap with service provider use cases, and described the context of this scoped work within a larger context of policy enforcement, and verification.

- The document had asset management, but the charter mentioned asset, change, configuration, and vulnerability management, so I added sections for each of those categories.

- Added text to Introduction explaining goal of the document.

- Added sections on various example use cases for asset management, config management, change management, and vulnerability management.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

[RFC0826] Plummer, D., "Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware", STD 37, RFC 826, November 1982.

[RFC1213] McCloghrie, K. and M. Rose, "Management Information Base for Network Management of TCP/IP-based internets:MIB-II", STD 17, RFC 1213, March 1991.

[RFC2790] Waldbusser, S. and P. Grillo, "Host Resources MIB", RFC 2790, March 2000.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[RFC2922] Bierman, A. and K. Jones, "Physical Topology MIB", RFC 2922, September 2000.

[RFC4363] Levi, D. and D. Harrington, "Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering, and Virtual LAN Extensions", RFC 4363, January 2006.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, March 2009.

[RFC6933] Bierman, A., Romascanu, D., Quittek, J., and M. Chandramouli, "Entity MIB (Version 4)", RFC 6933, May 2013.

Authors' Addresses

David Waltermire
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, Maryland 20877
USA

Email: david.waltermire@nist.gov

David Harrington
Effective Software
50 Harding Rd
Portsmouth, NH 03801
USA

Email: ietfdbh@comcast.net