National Institute of Standards and Technology
U.S. Department of Commerce

CAESARS Framework Extension: An Enterprise Continuous Monitoring Technical Reference Model (Second Draft)

Peter Mell, David Waltermire, Larry Feldman, Harold Booth, Alfred Ouyang, Zach Ragland, and Timothy McBride

NIST Interagency Report 7756 (Second Draft)

CAESARS Framework Extension: An Enterprise Continuous Monitoring Technical Reference Model (Second Draft)

Peter Mell, David Waltermire, Larry Feldman, Harold Booth, Zach Ragland, Alfred Ouyang, and Timothy McBride

C O M P U T E R   S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

January 2012

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analysis to advance the development and productive use of information technology. ITL's responsibilities include the development of technical, physical, administrative, and management standards and guidelines for the cost-effective security and privacy of sensitive unclassified information in Federal computer systems. This Interagency Report discusses ITL's research, guidance, and outreach efforts in computer security and its collaborative activities with industry, government, and academic organizations.

National Institute of Standards and Technology Interagency Report 7756
35 pages **(Jan. 2012)**

Acknowledgments

The authors would like to thank the original research team that developed the Department of Homeland Security (DHS) Federal Network Security's seminal work on continuous monitoring architectures. The Continuous Asset Evaluation, Situational Awareness, and Risk Scoring (CAESARS) architecture[1], created with MITRE support, formed the foundation of this work.

Also, we would like to recognize the following individuals for their participation on the continuous monitoring research team, insightful ideas, and review of this work: Stephen York, Peter Sell, and David Minge from the National Security Agency; Adam Halbardier, Adam Humenansky, Joe Debra, and Amit Mannan from Booz Allen Hamilton; and Mark Crouter from MITRE.

Finally, we would like to thank the United States Chief Information Officer Council's Information Security and Identity Management Subcommittee (ISIMC) on Continuous Security Monitoring for its leadership and direction as we created this publication. In particular we would like to thank the former and current co-chairs[2]: Colonel Michael Jones from the US Army, John Streufert from Department of State (DOS), Kevin Dulany from the Office of the Secretary of Defense (OSD), and Timothy McBride from DHS (also one of the authors).

This publication was authored through the cooperative work of the following organizations:

- National Institute of Standards and Technology (Peter Mell, David Waltermire, Harold Booth)
- Department of Homeland Security (Timothy McBride)
- Booz Allen Hamilton (Larry Feldman, Zach Ragland)
- MITRE (Alfred Ouyang)

Abstract

This publication and its supporting documents present an enterprise continuous monitoring technical reference model that extends the framework provided by the DHS Federal Network Security CAESARS architecture. This extension enables added functionality, defines each subsystem in more detail, and further leverages security automation standards. It also extends CAESARS to allow for large implementations that need a multi-tier architecture and focuses on the necessary inter-tier communications. The goal of this document is to facilitate enterprise continuous monitoring by presenting a reference model that enables organizations to aggregate collected data from across a diverse set of security tools, analyze that data, perform scoring, enable user queries, and provide overall situational awareness. The model design is focused on enabling organizations to realize this capability by leveraging their existing security tools and thus avoiding complicated and resource-intensive custom tool integration efforts.

Audience

This publication is intended for those planning to implement, develop products for, or support enterprise continuous monitoring capabilities. The model is broadly applicable to diverse networks including

[1] http://www.dhs.gov/xlibrary/assets/fns-caesars.pdf.
[2] Co-chairs are listed on the Office of Management and Budget website https://max.omb.gov/community/display/Egov/Continuous+Monitoring+Working+Group+Members.

industry, civilian government, state government, tribal, and military networks. Expected users of this document include Chief Information Security Officers, Chief Technology Officers, security tool vendors, security tool testing laboratories, security program managers, enterprise architects, and security procurement staff.

Although not a prerequisite, greater understanding of the CAESARS framework and our extensions will be gained by also reading the DHS CAESARS architecture[3].

[3] http://www.dhs.gov/xlibrary/assets/fns-caesars.pdf.

Table of Contents

List of Figures

1. Introduction and Document Overview

1.1 Introduction

The United States Office of Management and Budget (OMB) issued a memo in April 2010 requesting that the Department of State, Department of Justice, and Department of the Treasury coordinate with the Department of Homeland Security (DHS) to evaluate their continuous monitoring (CM[4]) best practices and scale them across the Government[5]. As a result of this evaluation, DHS released the *Continuous Asset Evaluation, Situational Awareness and Risk Scoring (CAESARS) Reference Architecture Report* version 1.8[6]. The CAESARS report provides a reference architecture, based on security automation standards, that guides organizations in deploying enterprise CM implementations.

In October 2010, the Federal Chief Information Officer Council's Information Security and Identity Management Committee's (ISIMC) subcommittee on CM saw the need to create a technical initiative to expand upon the CAESARS architecture to better scale it to large enterprises (e.g., the entire U.S. government). A team of researchers from the National Security Agency's (NSA) Information Assurance Directorate (IAD), the DHS Federal Network Security CAESARS team, and the National Institute of Standards and Technology's (NIST) Information Technology Laboratory (ITL) worked together to respond to this need. This publication is one of the outcomes of this joint research effort in support of the ISIMC's CM subcommittee.

The NSA's involvement ensured the architecture's applicability to U.S. military and national security systems with consideration for integrating with existing Department of Defense programs such as Computer Network Defense. DHS' participation ensured applicability to its CyberScope program[7] as well as consistency with the CAESARS vision. NIST's participation led to a model design that could support industry as well as government and a design well integrated with existing and emerging security automation standards.

This report describes the resulting CAESARS Framework Extension (FE), which builds upon the existing CAESARS architecture to make it more broadly applicable to the entire U.S. Government, including the Department of Defense, Intelligence Community, and civilian agencies. This work was also designed to be applicable to industry, state governments, and tribal networks, using a flexible model able to handle diverse customers and uses. In part, this was accomplished by extending CAESARS to allow for large implementations that need a multi-tier CM architecture. In addition, much work was done to enable additional functionality, provide more granularity within subsystem specifications, and further leverage security automation standards (e.g., for communication payloads and application interfaces).

The end goal of CAESARS FE is to enable enterprise CM by presenting a technical reference model that allows organizations to aggregate collected data from across a diverse set of security tools, analyze that data, perform scoring, enable user queries, and provide overall situational awareness. The focus is on primarily supporting cyber operations with compliance reporting as a by-product of actual security monitoring and improvement. This design is focused on enabling organizations to realize this capability by leveraging their existing security tools and minimizing custom tool integration efforts.

[4] The acronym CM in this publication is not to be confused with other NIST 800 series publications that use the acronym CM to denote "Configuration Management".

[5] The Department of Homeland Security's Continuous Asset Evaluation, Situational Awareness and Risk Scoring Reference Architecture Report, version 1.8, page xi (http://www.dhs.gov/xlibrary/assets/fns-caesars.pdf).

[6] http://www.dhs.gov/xlibrary/assets/fns-caesars.pdf.

[7] http://www.whitehouse.gov/sites/default/files/omb/assets/memoranda_2010/m10-15.pdf .

This report provides a high-level design for meeting this goal. However, success will require additional subsystem specifications, interface descriptions, and communication protocols. Achieving this goal will also require the input and participation of security-tool vendors and their customers to help develop and finalize these lower-level specifications.

Security-tool vendor participation is critical because their security sensors and controllers will need to be modestly instrumented to support the model and related security automation standards. Fortunately, much of this has already been accomplished through compliance with the NIST Security Content Automation Protocol (SCAP) Validation Program.[8] In addition, new functionality in data aggregation, analysis, and event management products will be needed. Fortunately again, many existing products already contain much of the needed functionality and should require only modest instrumentation to support the model.

If successful, CAESARS FE and the security products that support it will enable organizations to bring together diverse security products and, using those products, compose a hierarchical data aggregation model that supports a large variety of CM consumers from both the security disciplines and general information technology (IT) management domains. The challenge will be to minimally define the required functionality so that security tool vendors can cost-effectively participate, while ensuring a necessary level of interoperability between vendor products. This will require ongoing discussions, collaboration, and development within government and industry.

1.2 Document Overview

This report begins in Section 2 with a discussion of the definition of CM. From this definition, essential technical characteristics are derived that support the specific CM enterprise architecture (EA) view presented in Section 3. This EA view reveals the need for a set of goals that help in the review of the capabilities and limitations of the original CAESARS architecture presented in Section 4. Section 5 presents a high- level view of the FE to CAESARS that expands upon the original functionality and addresses the limitations. Section 6 discusses the supporting lower-level technical publications. Section 7 provides a conclusion. Appendix A lists the acronyms used in this report.

[8] NIST Security Content Automation Protocol Validation Program, http://scap.nist.gov/validation/index.html.

2. Defining and Scoping Continuous Security Monitoring

This section discusses several definitions of CM and extracts from these definitions essential technical characteristics for CM implementations. It also discusses the scope of CM to help focus the reader on what will be provided by this model and what will need to be provided by existing IT operations.

2.1 Definitions

CM can be broadly defined by the following:

> *Continuous monitoring is ongoing observance with intent to provide warning. A continuous monitoring capability is the ongoing observance and analysis of the operational states of systems to provide decision support regarding situational awareness and deviations from expectations.*

This definition applies to both Cybersecurity and general IT domains (e.g., network management). In this publication, we focus on the Cybersecurity domains, but the architecture presented is applicable to general IT domains as well. Because of the effort and expense involved in creating an effective CM solution, such solutions should be leveraged for as many uses as possible. We strive in this publication to support use across both Cybersecurity and IT management domains.

To focus on Cybersecurity, we now redefine CM in the context of security risk management using the NIST Special Publication (SP) 800-137 definition:

> Information security continuous monitoring is defined as *"maintaining ongoing awareness of information security, vulnerabilities, and threats to support organizational risk management decisions."*

Note: The terms "continuous" and "ongoing" in this context mean that security controls and organizational risks are assessed and analyzed at a frequency sufficient to support risk-based security decisions to adequately protect organization information.

For purposes of designing a technical reference architecture in this publication, we provide a more granular and process-focused description. From this we extract essential characteristics for CM implementations.

> *Continuous security monitoring is a risk management approach to Cybersecurity that maintains a picture of an organization's security posture, provides visibility into assets, leverages use of automated data feeds, monitors effectiveness of security controls, and enables prioritization of remedies.*

The essential characteristics for CM that can be derived from this definition are the following:

- Maintains a picture of an organization's security posture
- Measures security posture
- Identifies deviations from expected results
- Provides visibility into assets
- Leverages automated data feeds
- Monitors continued effectiveness of security controls
- Enables prioritization of remedies
- Informs automated or human-assisted implementation of remedies

These characteristics support the EA view of CM provided in Section 3.

2.2 Scoping and External System Interfaces

It is the intent of the architecture presented in this publication to clearly scope and bound our technical CM solution. Thus, we make a delineation in our model between what capabilities a technical CM implementation provides (e.g., providing analysis of events) and the *external* systems to which it interfaces. Multiple external systems will interface with any CM capability. For example, CM implementations must interface with asset management systems for a CM capability to determine what assets exist.

These external systems and technologies can be categorized to include at least 11 domains (see Figure 1) that could interface with a CM capability.[9]

1) Vulnerability Management
2) Patch Management
3) Event Management
4) Incident Management
5) Malware Detection
6) Asset Management
7) Configuration Management
8) Network Management
9) License Management
10) Information Management
11) Software Assurance

Figure 1. Continuous Monitoring Data Domains

Although the tools supporting these domains are not a core part of the technical CM capability, they need to be instrumented to interface with CM solutions. For this reason, they are included in our model but are clearly shown as external entities so that we can describe the needed interface requirements.

[9] NIST Special Publication 800-137, Information Security Continuous Monitoring for Federal Information Systems and Organizations, Appendix D.1. http://csrc.nist.gov/publications.

3. Enterprise Architecture View for Continuous Monitoring

This section presents an EA view[10] for CM with the purpose of illuminating high-level goals for the CAESARS FE technical reference model and associated implementations. Figure 2 displays the EA view of the enterprise continuous monitoring capability (ECMC).

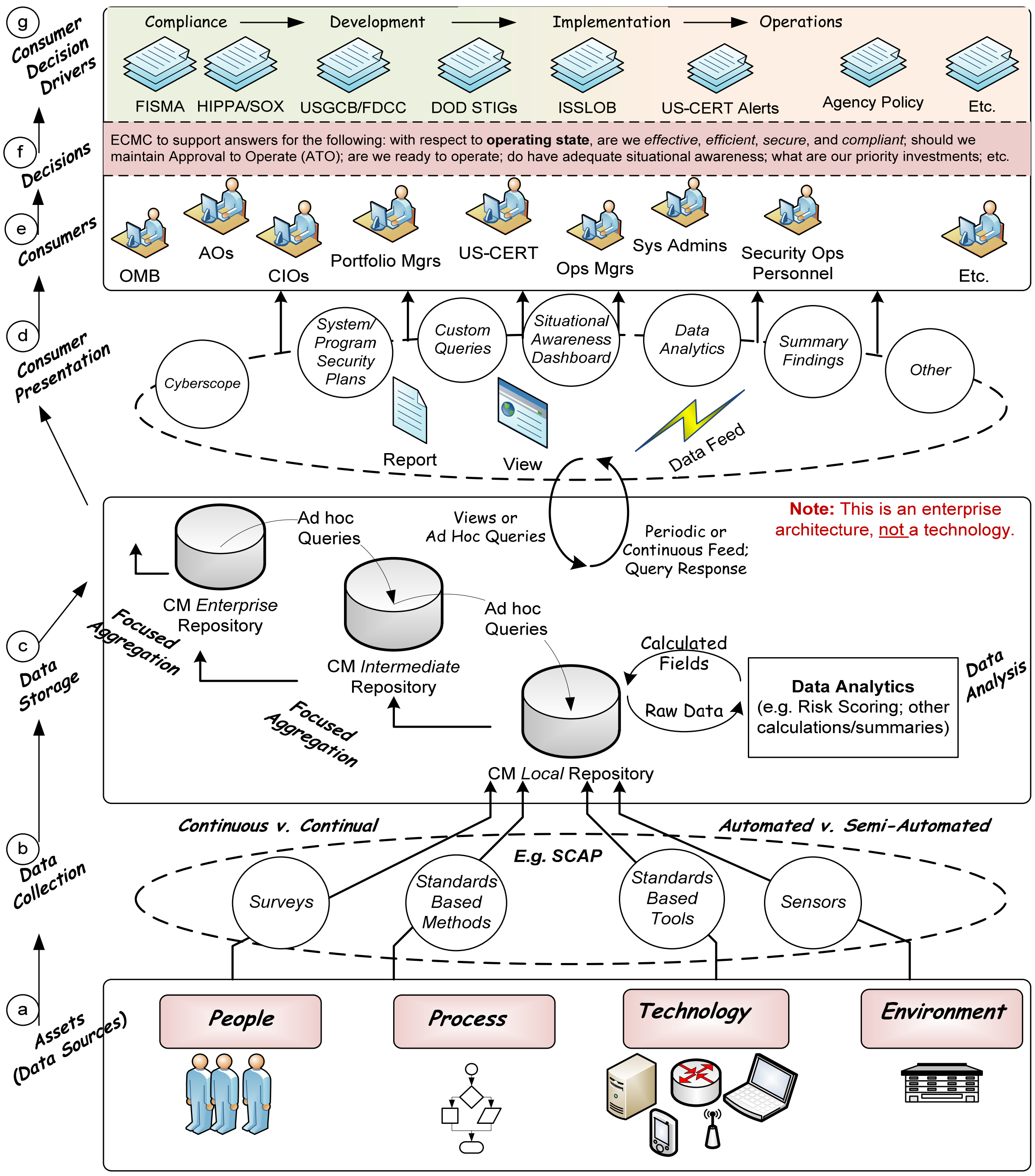

Figure 2. Enterprise Architecture View of Continuous Monitoring

[10] This was created by the National Security Agency Information Assurance Directorate and was modified slightly by the authors to depict the ad hoc queries and focused aggregation.

This is not a technical architecture but is illustrative of higher-level goals that need to be implemented by the CAESARS FE technical model. The following subsections describe each level of the EA view and make general observations regarding the required technical model. They start with an explanation of how data sources feed data collection activities and then show how that data is stored, analyzed, and presented to consumers who make decisions based on a variety of drivers.

a. Data Sources
The data sources for CM include the categories of people, process, technology, and environment. While many CM implementations may focus initially on technology, a CM technical architecture should be general enough to allow inclusion of the other categories. The people, process, and environment data types do not always lend themselves to fully automated data collection efforts and in most cases will require some human data collection effort.

b. Data Collection
A variety of methods, both automated and manual, can be used to collect data. Focus should be on using standards-based methods within tools for performing data collection to reduce integration costs, enable plug-and-play compatibility of subsystems, and enable CM implementations to incorporate diverse sets of security implementations. However, not all data feeds are currently standardized so the technical model needs to allow for acceptance and processing of proprietary data payloads. Human-generated data (e.g., from user surveys or from security compliance documentation) should be collected using mechanisms that harness automation and that leverage standardized methods. In addition, the appropriate frequency for data collection needs to be determined for each data feed. Some data feeds will be truly continual (always on) while others will be continuous (collected periodically at some set interval), and, in some cases, it may be more appropriate for the data feed to be event driven.

c. Data Storage and Analysis
The collected data will initially reside at a local repository near the point of collection and then may be aggregated at higher tiers in the organization. Having CM data available at each tier enables users at that tier to have an appropriately abstracted view into the organization's security posture. Normally, users at each tier need only an abstracted view of the lower-level CM data, and thus it is not necessary to duplicate all data up through each tier. Replicating all low-level security data at multiple levels within an organization could pose an increased security risk, along with authorization and scalability challenges. For this reason, the CM EA view shows a "focused aggregation" occurring when transferring CM data from a lower-level repository to a higher-level repository. Only the data needed by the next higher tier is transferred up and duplicated. Determining these necessary "predefined views" is an important step in any CM implementation. Ideally, the majority of the CM data, especially the most sensitive data, will stay at the local repository level. At this level, the information is closest to the authoritative sources (i.e., data collectors), allows for fine-grained access control, is the timeliest copy of the data, and poses the least aggregation risk.

This model for performing data aggregation using predefined views is a common approach for CM implementations. However, it is likely that users at higher tiers will have an operational need to occasionally query data that is outside of the available predefined view. In such cases, organizations may be tempted to aggregate more and more of the data at all tiers. At an extreme, they may attempt to aggregate and duplicate all CM data at all tiers. This may result in network bandwidth and data storage challenges while presenting additional security risks. To alleviate this need, the EA CM view enables users at one tier to issue operational, or "ad hoc," queries that are propagated down to lower tiers for data retrieval. If the requested data is not available in local repositories, the operational query from a higher-level tier may trigger additional data collection at the lower tiers. The technical CM architecture will need to allow for this operational querying while putting into place the necessary task management systems so that such operational queries are reviewed, approved, and do not result in a degradation of the local network or systems.

A final consideration on data storage is that the same technical model (e.g., interfaces, protocols, and payloads) should be used for data aggregation regardless of which tier is communicating. This avoids having to create different CM aggregation solutions for differing tiers.

d. Consumer Presentation
Each tier within the data storage layer will provide a view of the data to consumers. The presentation layer needs to be flexible enough to satisfy diverse data display needs because the CM implementation must support many types of consumers. Primarily the CM implementation should support the operational mission in helping to secure an organization (likely through situational awareness dashboards). It will also need to support compliance reporting, executive-level reporting, and reporting for non-security use cases.

e, f, and g: Consumers, Decisions, and Consumer Decision Drivers
There are many types of consumers that need CM data ranging from system administrators, to the organization Chief Information Officer (CIO), to possibly external compliance or auditing entities. These consumers need to make decisions (especially those regarding effectiveness, efficiency, security, and compliance) based on a set of drivers. The CM architecture must provide them the necessary information to make these decisions.

4. Foundational Work

DHS conducted seminal CM work published as the *Continuous Asset Evaluation, Situational Awareness and Risk Scoring (CAESARS) Reference Architecture Report.*[11] CAESARS was the result of a DHS evaluation of successful CM implementations within the civilian government: Department of State, Department of Justice, and Department of the Treasury. DHS found commonality and strengths in the approaches of these civilian agency custom solutions and used this as the basis for creating the CAESARS reference architecture. The CAESARS architecture fulfills many, but not all, of the goals of the CM EA view.

4.1 Overview of the CAESARS Reference Architecture

CAESARS enables organizations to implement a single CM instance that consists of four subsystems: Sensor, Database, Presentation/Reporting, and Analysis/Risk Scoring. All subsystems, except the Database subsystem, may contain multiple tools providing independent observation or analysis. A single database is used to aggregate monitoring data from the Sensor subsystem, all its distinct sensor products, and their instantiations throughout the enterprise. This central database is also used by the Presentation/Reporting and Analysis/Risk subsystems as their source of monitoring data. An enterprise service bus (ESB) is used for all inter-subsystem communication.

A review of the subsystem descriptions follows along with Figure 3,[12] which shows a "contextual description of the CAESARS system." All quotations in this subsection are taken from the CAESARS publication version 1.8[13] published in September 2010.

[11] http://www.dhs.gov/xlibrary/assets/fns-caesars.pdf.
[12] Diagram is from CAESARS version 1.8, page 11.
[13] http://www.dhs.gov/xlibrary/assets/fns-caesars.pdf.

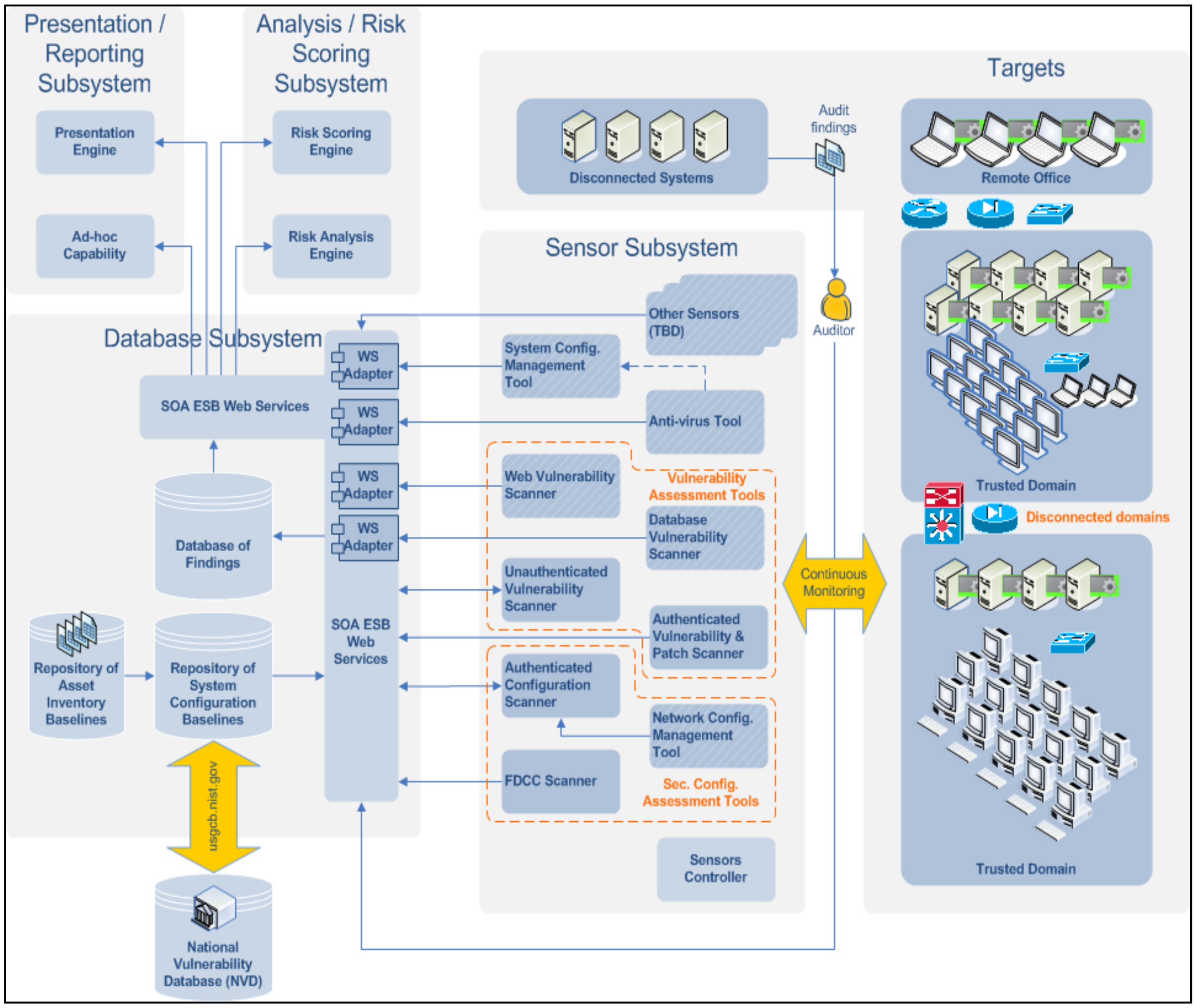
Presentation / Reporting Subsystem
Presentation Engine
Ad-hoc Capability
Analysis / Risk Scoring Subsystem
Risk Scoring Engine
Risk Analysis Engine
Targets
Disconnected Systems
Audit findings
Remote Office
Auditor
Sensor Subsystem
Other Sensors (TBD)
System Config. Management Tool
Anti-virus Tool
Web Vulnerability Scanner
Vulnerability Assessment Tools
Database Vulnerability Scanner
Unauthenticated Vulnerability Scanner
Authenticated Vulnerability & Patch Scanner
Authenticated Configuration Scanner
Network Config. Management Tool
FDCC Scanner
Sec. Config. Assessment Tools
Sensors Controller
Continuous Monitoring
Trusted Domain
Disconnected domains
Trusted Domain
Database Subsystem
SOA ESB Web Services
WS Adapter
WS Adapter
WS Adapter
WS Adapter
SOA ESB Web Services
Database of Findings
Repository of Asset Inventory Baselines
Repository of System Configuration Baselines
usgcb.nist.gov
National Vulnerability Database (NVD)

Figure 3. Contextual Description of the CAESARS System

4.1.1 Sensor Subsystem

"The Sensor Subsystem includes the totality of the IT assets that are the object of CAESARS' monitoring activities. It includes all platforms upon which CAESARS is expected to report, including end-user devices, database servers, network servers, and security appliances." Figure 3 shows nine sensor types, and it is important to note that DHS has ongoing government-wide procurements for these product types within its Information Systems Security Line of Business (ISSLOB).[14]

4.1.2 Database Subsystem

"The purpose of the CAESARS Database is to serve as the central repository of all CAESARS data, including both raw data that is reported from the sensors on the individual platforms of the Sensor Subsystem and data that is developed within the CAESARS Analysis/Risk Scoring Subsystem as a result of 'cleansing', pre-processing, analysis, and scoring processes. . . . [The CAESARS database] also includes any tools that are required by the CAESARS Database/Repository to perform data-pull operations from the Sensor Subsystem platforms." Of note is that the Database Subsystem

[14] http://www.us-cert.gov/GFIRST/presentations/Information_Systems_Security_Line_of_Business_ISSLOB_Overview.pdf.

encompasses the “repository of configuration baselines” and thus it contains machine-readable descriptions of the required baseline security posture of the organization’s systems as well as data on known vulnerabilities and their severity (e.g., from the National Vulnerability Database [NVD]). It also encompasses a “repository of asset inventory baselines,” although this asset inventory database is not described in much detail.

4.1.3 Analysis/Risk Scoring Subsystem

“The CAESARS Analysis/Risk Scoring Subsystem [consists] of multiple analytic tools, possibly querying the same or different portions of the database [subsystem], with either local (region- or site-specific) or enterprise-wide perspectives, yet still provide confidence that no single type of analysis will influence or skew any other.” It is the CAESARS’ architecture of independent plug-and-play components that enables this capability to allow a variety of risk-scoring schemes and tools to be simultaneously deployed without any one of them interfering with the others.

4.1.4 Presentation/Reporting Subsystem

“The CAESARS Presentation and Reporting Subsystem can include multiple presentation tools, with either local or enterprise-wide perspectives, yet still provide confidence that no one type of presentation will influence or skew any other.” This enables a variety of display schemes, serving diverse user types, to be simultaneously deployed without any one of them interfering with the others.

4.2 Limitations of the CAESARS Reference Architecture

The CAESARS reference architecture provides a strong foundation on which to build a CM technical architecture that meets our CM EA goals. However, it has limitations in several areas that are required to evolve CAESARS into providing stronger capabilities.

4.2.1 Lack of Interface Specifications

CAESARS does not specify the machine-level interfaces that subsystems can use to communicate with other subsystems. This limits the plug-and-play capabilities within CAESARS and requires each implementation of CAESARS to undergo custom integration efforts.

4.2.2 Reliance on an Enterprise Service Bus

CAESARS specifies using an ESB for communication between subsystems. ESBs may not be an optimal communication mechanism for some types of communication and may not be appropriate for inclusion within certain vendors’ products.

4.2.3 Incomplete Communication Payload Specifications

CAESARS does explore payload specification for the Sensor subsystem to communicate to the Database subsystem. However, it does not provide a full specification for all sensor types that can be used by vendors to build CAESARS-compatible security tools. Furthermore, CAESARS does not specify the communication payloads between the other subsystems.

4.2.4 Lack of Specifications Describing Subsystem Capabilities

CAESARS describes each subsystem in general but does not provide detailed specifications. Such specifications are needed to support procurement activities, enable vendors to instrument their tools to support subsystem capabilities, and enable development of product testing methodologies in support of product validation programs (e.g., SCAP validation[15]).

4.2.5 Lack of a Multi-CM Instance Capability

CAESARS is designed as a monolithic capability whereby each organization is to create a single CM instance. Because CAESARS also requires one of each subsystem, this means that all organizational security data will need to be duplicated within a single central CM database. This runs counter to the structure of many organizations (especially federal government ones) that tend toward a decentralized approach to IT management. This also runs counter to the EA CM goals of performing focused aggregation and leaving the most fresh and timely CM data at the leaf nodes of a hierarchical tiered structure. Thus, most large organizations will need the ability to have multiple CM instances.

4.2.6 Lack of Multi-Subsystem Instance Capability

CAESARS specifies that a CM instance have only one of each type of subsystem. However, it also allows for and encourages a variety of independent security tools within each subsystem. For example, multiple distinct presentation tools are allowed within the Presentation/Reporting subsystem. This adds complexity because there is no discussion as to how the different tools within a subsystem communicate independently with the rest of the model and also within the subsystem itself. This complexity could be eliminated by allowing multiple instances of a subsystem type and then having a separate instance for each relevant security tool (e.g., presentation system). Although users of CAESARS could avoid this problem by having only one vendor tool within each subsystem, such an approach may limit the ability to have full coverage of platforms and security features. Having multiple tools (especially security sensors) will increase trust in the collection and the scope of what is collected and allow for a deeper analysis of the data.

4.2.7 CM Database Integration with Security Baseline Content

CAESARS integrates the “repository of system configuration baselines” and the “database of findings” within the Database subsystem. A variety of vendor tools exist that implement one, but not the other, of those capabilities. To facilitate vendor adoption of CAESARS, it may be advantageous to separate the configuration baselines and database findings into multiple subsystems.

In addition, different parts of an organization may need to customize policies on security posture to fit their environment and unique mission. This is usually managed vertically with policies pushed down from above and modified at the lower levels. Having a single repository of baselines within a single Database subsystem makes it more difficult to allow customization of the expected security posture.

4.2.8 Lack of Detail on the Required Asset Inventory

CAESARS provides very little detail on how an asset inventory is to be maintained and how it relates to the repository of configuration baselines.

[15] http://scap.nist.gov/validation/index.html.

4.2.9 Requirement for Risk Measurement

CAESARS requires that the Analysis/Risk Scoring subsystem measure security risk. According to the NIST risk management publication SP 800-37 revision 1,[16] risk is defined as follows:

> "A measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of: (i) the adverse impacts that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence."

Given the requirement to calculate the likelihood of an event and the requirement to calculate the extent to which an adverse impact will harm an organization, it is difficult to measure risk, and CM solutions often do not have the necessary inputs. Thus, CM architectures should allow for a more simple and adaptable focus on measuring the effectiveness of the security posture or controls[17] while not eliminating the possibility of a CM system also calculating risk (if the necessary inputs are available).

[16] http://csrc.nist.gov/publications/nistpubs/800-37-rev1/sp800-37-rev1-final.pdf.
[17] These lower-level security effectiveness measurements can then be used as (necessary but not sufficient) inputs to a risk measurement capability.

5. CAESARS Framework Extension

The CAESARS FE is a technical reference model for enterprise CM that builds on the DHS CAESARS reference architecture. Most of the CAESARS subsystems remain in CAESARS FE, but modest revisions have been made to the higher-level architecture to provide enhanced functionality and allow multi-tier CM implementations.

This section presents the high-level CAESARS FE model, subsystem overview, and multi-tier capability. As with CAESARS, CAESARS FE can be used to support operational security needs as well as compliance evaluation and reporting. Unlike CAESARS, CAESARS FE is designed as a data domain agnostic model allowing collection, aggregation, analysis, presentation, and reporting on a variety of IT areas (both security and general IT management). This data domain-agnostic model can then be instantiated into a variety of specific architectures covering specific data domains.

This section does not provide low-level descriptions of CM workflow, subsystem specifications, and interface specifications. This detail is provided in NIST Interagency Report (IR) 7799. This section also does not apply (or bind) the model to specific data domain areas (such as asset, vulnerability, and configuration management). That detail is provided in NIST IR 7800. See Section 6 for more detail on these publications.

5.1 Variations on the CAESARS Architecture

CAESARS FE is designed to overcome the previously described limitations of the CAESARS architecture. To accomplish this, CAESARS FE will provide the following capabilities:

- Enable a variety of subsystem interfaces to be defined (removing the required ESB from within the CAESARS Database subsystem) (see 4.2.1 and 4.2.2)
- Provide detailed subsystem communications payload specifications (see 4.2.3)
- Provide detailed specifications for each subsystem that enable tool development and support product validation program or agency procurement (e.g., DHS ISSLOB) (see 4.2.4)
- Allow for the instantiation of multiple CM instances (see 4.2.5)
- Allow for the construction of a CM tiered hierarchical architecture (see 4.2.5)
- Allow for the existence of multiple instances of individual subsystems (see 4.2.6)
- Create a separate subsystem for a Task Manager (created by extracting the “sensors controller” from the CAESARS Sensor subsystem which allows each sensor to be its own Collection subsystem, 4.2.6)
- Create a separate subsystem for a security content server (extracting the CAESARS “repository of configuration baselines” from its Database subsystem) (see 4.2.7)
- Further define specifications for the asset inventory database (see 4.2.8)
- Provide flexibility to perform general enterprise measurement as opposed to being restricted to true risk measurement (see 4.2.9).

5.2 Subsystem Overview

CAESARS FE contains six distinct subsystems that together compose the CM model. These subsystems are described in more detail in subsequent sections:

1. **Presentation/Reporting:** This subsystem takes user input, creates well-specified data queries, and outputs available results.
2. **Content:** This subsystem stores digital policy and supporting data (e.g., for checking system states).

3. **Collection:** This subsystem detects system state information in accordance with organizational policy.
4. **Data Aggregation:** This subsystem stores system state information, related calculated results, and associated metadata.
5. **Analysis/Scoring:** This subsystem analyzes data and scores system state information.
6. **Task Manager:** This subsystem orchestrates the activities of the other subsystems and communicates with other CM instances in enabling fulfillment of user data queries.

A single instance of CM, its subsystems, and associated components are shown in Figure 4. CAESARS Framework Extension Subsystems and Components

.Large organizations may have multiple communicating instances. Extremely large organizations, such as the U.S. Government, may need a large hierarchy of cooperating CM instances to support diverse and localized operations, higher-level decision makers, and also government-wide security operations (e.g., DHS CyberScope).

Note that the Collection subsystems are marked as external to the model. This is because such systems will likely exist independently from the CM implementation, but they need to be instrumented to fulfill their role in collecting CM data. The Content subsystem, while a core piece of the model, may also play an independent but integrated role as a data policy management service. The other four subsystems form a tight core that will be necessary to successfully compose any CM architecture.

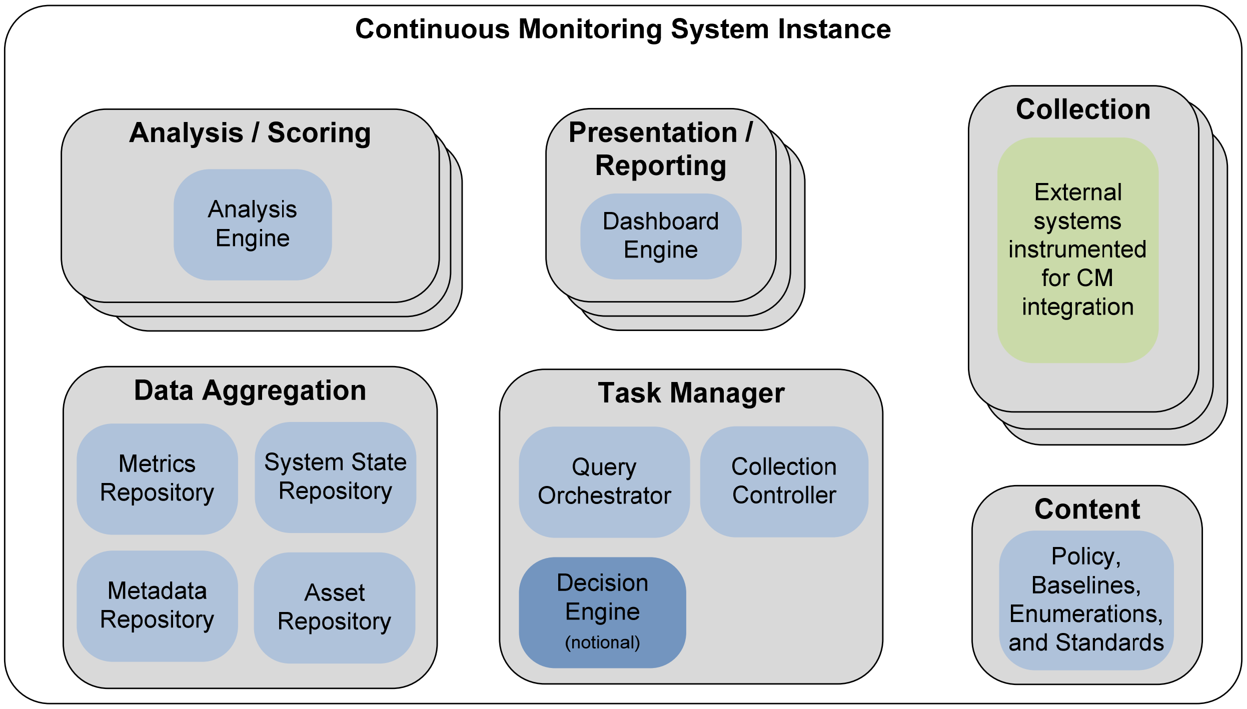

Figure 4. CAESARS Framework Extension Subsystems and Components

A primary goal of this model is to enable organizations to implement CM solutions by composing best-of-breed tools from diverse vendors. Thus, different tools will be able to fulfill different subsystem requirements, and those tools can be combined to implement a CM solution. Tool integration will be made cost-effective through adoption of common interfaces and data normalization capabilities. Custom integration is minimized except in the case of any necessary proprietary data feeds, and in such cases, the integration costs are still reduced through the provision of standard interfaces that can "wrap" proprietary data.

5.2.1 Presentation/Reporting Subsystem

The Presentation/Reporting subsystem is similar to the CAESARS subsystem of the same name. Unlike CAESARS, however, a CM instance may have more than one such subsystem. This enables different CM users to have their own interfaces customized to their particular needs.

The subsystem consists of a single component: the Dashboard Engine. As shown in Figure 5. Presentation/Reporting Subsystem Communication, it communicates with the Task Manager's Query Orchestrator.

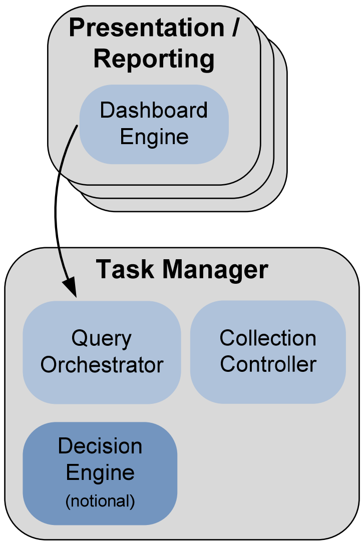

Figure 5. Presentation/Reporting Subsystem Communication

The subsystem's Dashboard Engine is the primary user interface for the CM solution. Users are provided the capability to describe needed data, and the subsystem formalizes their requests into well-specified queries. Queries are then sent to the Task Manager for fulfillment. Upon receipt of a response, the subsystem displays the data or provides a report as requested by the user.

5.2.2 Task Manager Subsystem

The Task Manager subsystem fulfills CM query requests by orchestrating the activities of other CM subsystems. This includes controlling the data collected, activating scoring routines, and providing query responses. It also communicates between CM tiers to handle resolution of queries that cover data from multiple CM instances. It thus acts as a single, central controller for a CM instance.

Although many of the other subsystems are derived from CAESARS subsystems, the Task Manager does not exist in CAESARS. In CAESARS, management of the data collection is done by the "Sensors Controller." In addition, in CAESARS, the Presentation/Reporting subsystem and the Analysis/Risk Scoring act independently and do not need the orchestration provided by a Task Manager.

CAESARS FE adds the Task Manager so that the CM solution can modify its behavior to optimally respond to user queries. This means that queries from the Presentation/Reporting subsystem should dynamically trigger the required data collection. Completion of data collection should trigger data deconfliction (described in section 5.2.5), analysis, and scoring. Completion of scoring should trigger a response to the relevant query. In addition to this, communications must be established to respond to queries covering multiple CM instances. The Task Manager then enables diverse security tools, all with their own roles and functions, to work together as a team in providing situational awareness to the enterprise.

The Task Manager intra-CM instance communications are shown in Figure 6. Task Manager Subsystem Communication. The Task Manager receives queries from the Presentation/Reporting subsystem and directs the Collection subsystems to collect the needed data. It communicates with lower-tier CM instances, receives their data, and stores the data in the Data Aggregation subsystem. Finally, it sends queries to the Analysis/Scoring subsystem to get the required results. The Task Manager then delivers the results back to the requestor (usually the Presentation/Reporting subsystem but possibly a higher-tier CM instance). The Task Manager communicates with higher- and lower-tier CM instances, but those communications are not shown here.

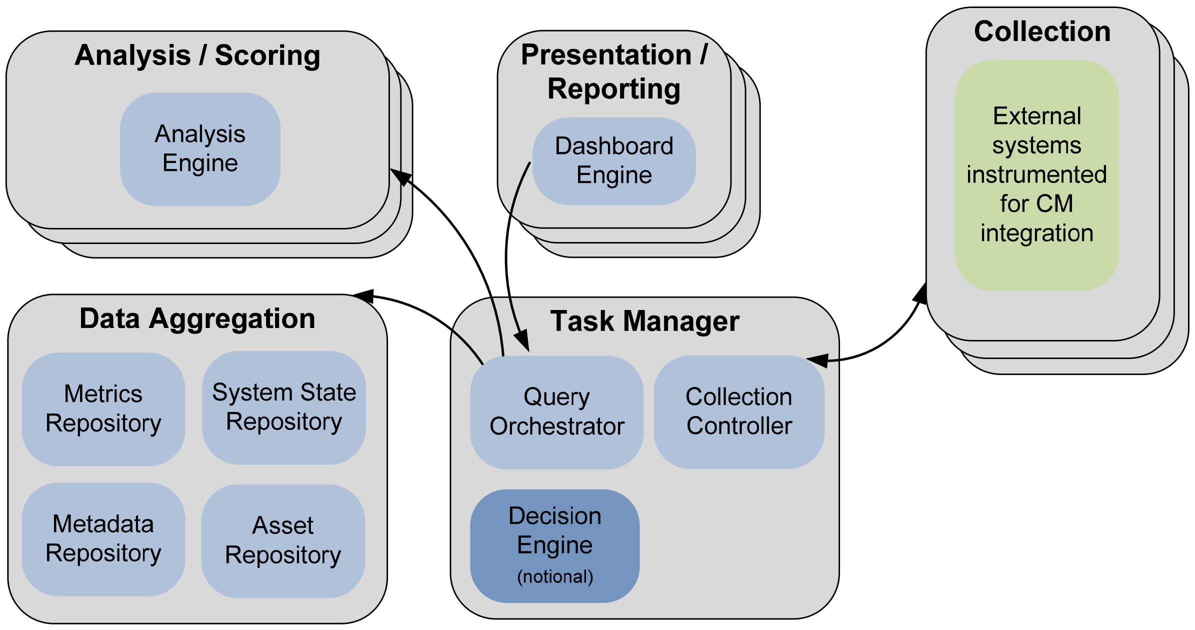

Figure 6. Task Manager Subsystem Communication

Each CM instance has one Task Manager subsystem composed of three components: the Query Orchestrator (QO), the Collection Controller (CC), and the Decision Engine (DE), which is currently notional).

<u>Query Orchestrator (QO)</u>
The QO component receives query requests either from the Presentation/Reporting subsystem or from a higher-tier CM instance. It then applies policy to determine whether it will let the query execute and may activate a human approval process. It works with the Analysis/Scoring subsystem to attempt to retrieve the results without having to collect data. If unsuccessful, it will orchestrate data collection. If data needs to be collected from lower-tier CM instances, it propagates the query to those instances. When it receives results, it stores them in the Data Aggregation subsystem. If data needs to be collected from the current CM instance, it sends the query to the CC component. Once all data has been collected, it sends the query to the Analysis/Scoring subsystem and receives the query response in reply. It then forwards that response to the requestor.

The QO thus orchestrates query fulfillment among multiple subsystems and possibly multiple CM tiers. Thus, this component enables the teamwork among the diverse security tools that make up a CM instance.

Collection Controller (CC)
The CC component receives queries from the QO and ensures that the necessary data collection occurs (within the CM instance) to fulfill the query. It does this by decomposing the query into a set of data collection tasks that are sent out to the relevant Collection subsystems. It keeps track of task completion and informs the query orchestrator when all data has been collected to support a particular query.

Decision Engine (DE)
The DE is a notional component that may support and help enforce digital policies. The idea is that the DE could receive a digital policy and then monitor compliance against that policy. The DE would do this by ensuring that the right data is being collected by the Collection subsystems and then by periodically analyzing that data using the Analysis/Scoring subsystem. It could accomplish these goals by simply issuing (machine generated) queries to the QO.

Implementation of the DE then would take the CM solution beyond responding to human queries and toward full security automation based on digital policy directives.

5.2.3 Collection Subsystem

The Collection subsystem is similar to the CAESARS "Sensor subsystem" except that each instance of a Collection subsystem must contain only a single vendor's solution. Thus, there will be a single management console that may cover multiple tools that map to one or more of the CAESARS sensor types. Another change is that the CAESARS sensors controller has been moved to the CAESARS FE Task Manager subsystem and been renamed the "collection controller" (see the Task Manager section for more details).

Collection subsystem communication is shown in Figure 7. Collection Subsystem Communication. The subsystem can accept tasking from the Task Manager that will direct it to collect specific data in support of a user query. It can also retrieve content from a Content subsystem to enable it to collect the correct data according to organizational policy. Finally, the subsystem can communicate collected data to the Data Aggregation subsystem for storage. In alternate implementations of the model, collected data may be sent back to the Task Manager for processing prior to being stored.

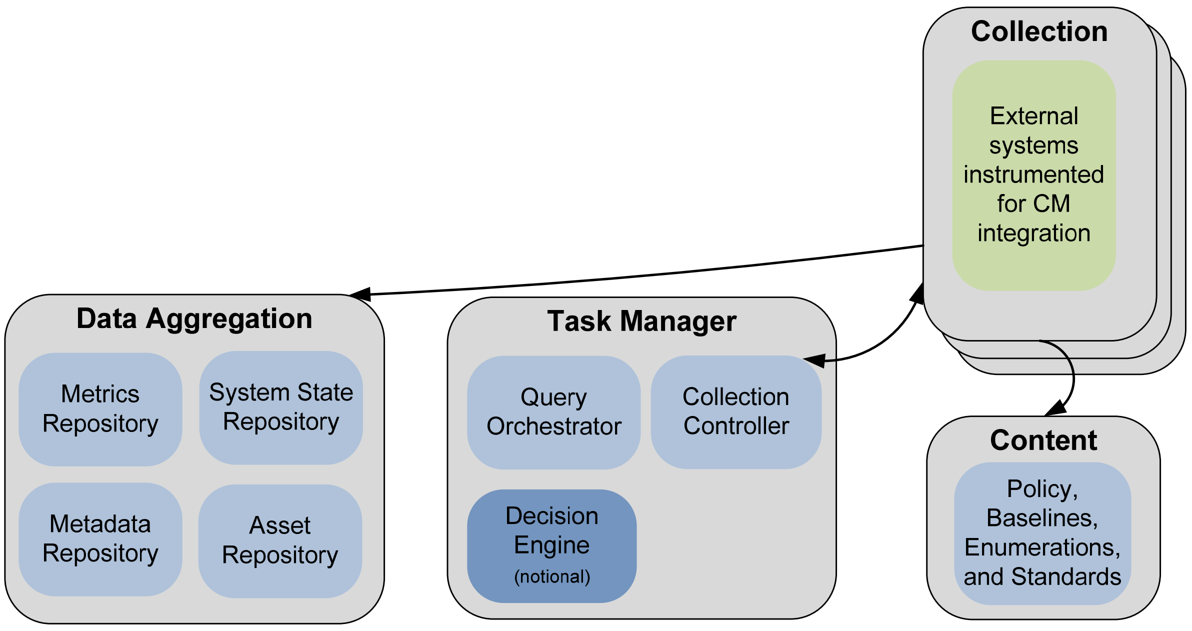

Figure 7. Collection Subsystem Communication

A particular CM instance is not required to have a Collection subsystem. This may happen in a multi-tiered, hierarchical implementation of CM instances where a higher-tier CM instance does not monitor any assets. In this case, the CM instance simply relies on data feeds from lower-tiered CM instances. This said, most CM instances will have not only a Collection subsystem but also multiple Collection subsystems to enable them to use a diverse set of sensor tools.

5.2.4 Data Aggregation Subsystem

The Data Aggregation subsystem has similarities to the CAESARS "Database subsystem" but is very different. Like CAESARS, the Data Aggregation subsystem provides a repository to store and retrieve data. However, it does not contain an ESB nor is it the central hub linking together the other subsystems. It also does not contain configuration baseline information because that is covered by the Content subsystem. The two do overlap in storing asset data and system state information (called findings in CAESARS). However, the Data Aggregation subsystem stores additional data not found within the CAESARS version including raw data, analyzed data (CAESARS FE findings), calculated scores, and metadata.

The Data Aggregation subsystem communications are shown in Figure 8. Data Aggregation Subsystem Communication. The subsystem receives raw data (and possibly findings) from the Collection subsystems. It may also receive the query results for lower-tier CM instances from the Task Manager if a hierarchical CM architecture has been deployed. Finally, it offers data retrieval and storage services to the Analysis/Scoring subsystem to enable analysis and scoring of data.

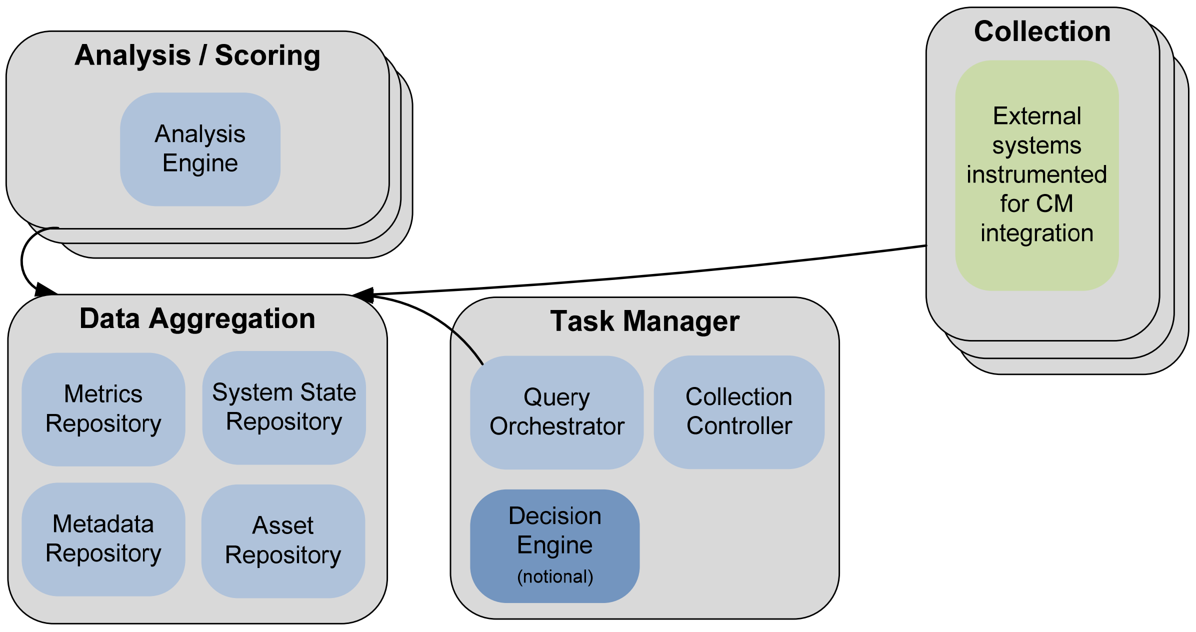

Figure 8. Data Aggregation Subsystem Communication

Each CM instance must have exactly one Data Aggregation subsystem. This ensures having a single authoritative source of CM data per CM instance. This single instance contains four components:

1. System State Repository

The system state repository contains the raw data and findings that are provided by the Collection subsystems. While not necessary, some preprocessing of the data may occur at this point to improve the quality of the data. If done, this preprocessing will alter the analysis of the scoring process. For this reason, the primary data deconfliction will normally occur within the Analysis/Scoring subsystems so that each Analysis/Scoring subsystem can use its own techniques.

2. Asset Repository

The asset repository stores standards-based representations of asset data retrieved from the Collection subsystems. Thus, the asset repository component itself is not an asset inventory system but simply an aggregation database. Collection subsystems specializing in asset management and assessment may feed the asset repository. However, it can also be fed by tools that perform asset identification as a secondary function to their primary purpose (e.g., configuration scanning).

3. Metrics Repository

The metrics repository stores the scoring results generated by the Analysis/Scoring subsystems. The data is tagged with the name of the component that created it, enabling multiple Analysis/Scoring subsystems to cache different metrics without one corrupting the other’s data and for traceability.

4. Metadata Repository

The metadata repository stores metadata used for raw data deconfliction and scoring. The “deconfliction rules” are essentially filtering rules on how to treat apparently duplicative, but different, data elements. These rules may include the relative accuracy of different Collection subsystems.

5.2.5 Analysis/Scoring Subsystem

The Analysis/Scoring subsystem is similar to the CAESARS subsystem of the same name. It differs in that it provides generalized analysis and scoring services as opposed to focusing only on “risk” measurement. This does not preclude using this subsystem for risk measurement, though, because this type of measurement is a subset of the more general concept of scoring and, for security-focused CM implementations, risk scoring should be an objective.[18] Another difference is that analysis and scoring (as well as data deconfliction) are handled by a single component called the Analysis Engine.

Analysis/Scoring subsystem communications are shown in Figure 9. Analysis/Scoring Subsystem Communication. The subsystem can accept queries from the Task Manager to be analyzed and scored. It can retrieve data from, and store data in, the Data Aggregation subsystem. It can update itself with scoring algorithms, scoring parameters, and associated scoring data from the Content subsystem.

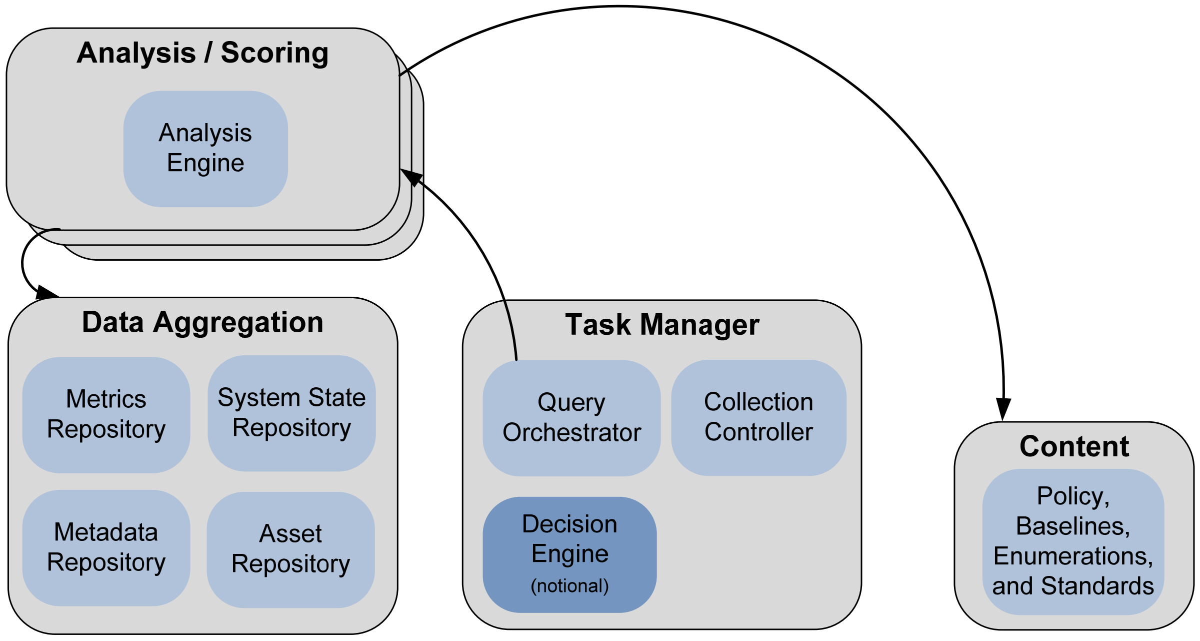

Figure 9. Analysis/Scoring Subsystem Communication

CM instances must have at least one Analysis/Scoring subsystem but they may have more than one. This allows for organizations to leverage multiple analysis engines that may support different scoring algorithms. The subsystem’s analysis is based on the principle that raw data will be collected in the Data Aggregation subsystem’s System State Repository. The Analysis Engine will then deconflict that data according to rules within the Data Aggregation subsystem’s Metadata Repository (if any). Deconfliction involves determining authoritative sources when the same data is reported multiple times (possibly with differing results). Once the raw data has been deconflicted, the subsystem will use the scoring algorithm, along with policy from the Content subsystem, to create “findings.” Findings are often Boolean values (i.e., true and false values) representing the comparison of raw data against policy describing the expected value of the raw data. Findings are stored in the Data Aggregation subsystem’s Metrics Repository so they can be cached and reused. Next, the subsystem must convert findings into scores. This means taking one or more findings and applying an algorithm to generate a numerical answer. A sequence of findings steps followed by a sequence of scoring steps may be necessary to fulfill a particular query request. This

[18] **True risk scoring can be difficult to achieve using the NIST SP 800-37 revision 1 definition, and many “risk scoring” methodologies do not demonstrate a correlation to true risk measurement. Instead, they usually measure the state of a collection of security capabilities or controls.**

delineation of raw data, findings, and scoring is an important architectural component of the CAESARS FE model because it enables reuse of intermediate computations. It is also useful because it provides a structure around which we can design specifications to express arbitrary analysis and scoring methodologies.

5.2.6 Content Subsystem

The Content subsystem is similar to the CAESARS "repository of system configuration baselines" within the Database subsystem. However, its scope has been expanded to include general organizational digital policy and supporting data (e.g., enumerations and standards for data normalization). Its scope has also been expanded to include non-security policy and related data. Thus, the Content subsystem is more of a digital policy management repository than was originally envisioned in the CAESARS. Given this recasting, the data in the Content subsystem support comparison of system state information against organizational policy as well as data normalization.

The subsystem consists of a single component that communicates with the Collection subsystem and Analysis/Scoring subsystem as shown in Figure 10. Content Subsystem Communication**Error! Reference source not found.**. Also shown is the Content subsystem's communication with content providers and content development tools. Not shown is the ability of a Content subsystem to communicate with other Content subsystems within an organization's CM solution.

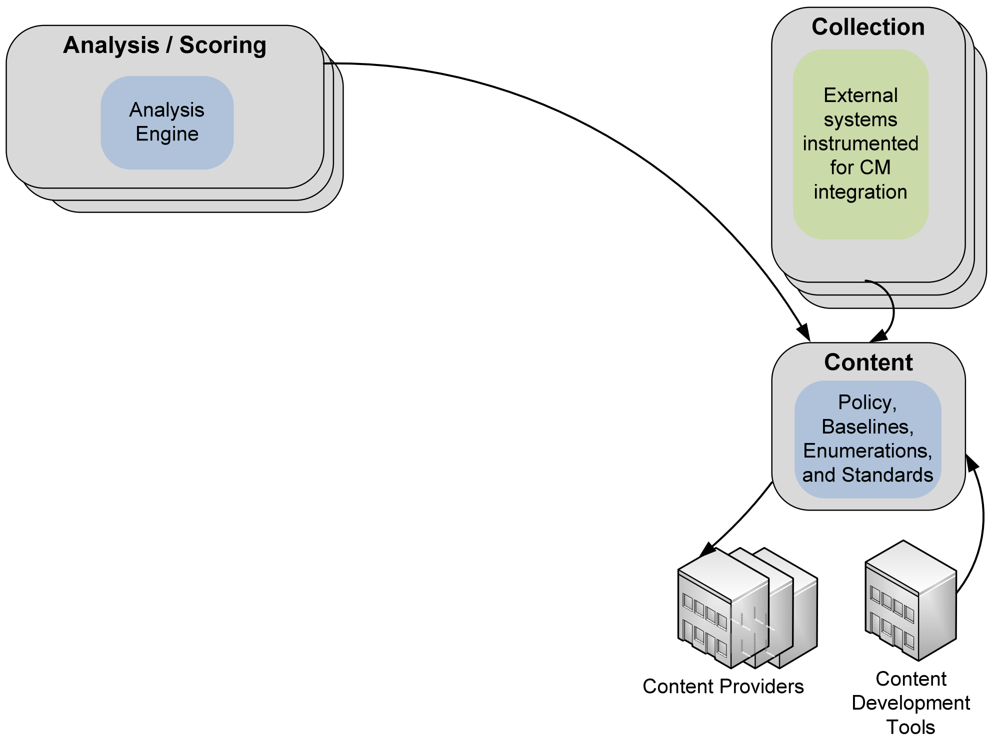
Analysis / Scoring
Analysis Engine
Collection
External systems instrumented for CM integration
Content
Policy, Baselines, Enumerations, and Standards
Content Providers
Content Development Tools

Figure 10. Content Subsystem Communication

The subsystem communicates with content providers and other Content subsystems to acquire content. It communicates with content development tools to allow tailoring or authoring of content. Retrieved content is stored, time stamped, tagged with its source, and tagged with whether the content was obtained from within the CM implementation or from an external source. Two types of data can be retrieved: organizational policy and supporting data. Organizational policy data includes benchmarks (e.g., organization-wide security configuration policy) and baselines (e.g., customized security configuration policy for specific systems). Supporting data elements include enumerations of data types and associated scores (e.g., product names, vulnerability names, and vulnerability impact scores) as well as information on how to check system state. The purpose of having this data within a CM architecture is to enable measurement of the system against a required state. Another purpose is to enable measurement using normalized data elements so that multiple CM instances will report on the state of systems using comparable languages.

The following are examples of typical (but not required) types of organizational policy included within a Content subsystem:

- Software asset baselines (allowed software and required versions)
- Security configuration benchmarks (e.g., Federal Desktop Core Configuration)
- Lists of required patches
- Lists of authorized software
- Required network port configurations

The following are examples of typical (but not required) types of supporting data elements included within a Content subsystem:

- Lists of known vulnerabilities and their impact scores
- Lists of known configuration issues and their impact scores
- Lists of asset names that may be applicable (not the list of actual assets in a system)

To the greatest extent possible, this data should be represented using widely adopted specifications to promote reuse, modification, and normalization. The following are examples of typical (but not required) specifications that may be leveraged:

- Extensible Configuration Checklist Description Format (XCCDF)
- Open Vulnerability and Assessment Language (OVAL)
- Open Checklist Interactive Language (OCIL)
- Common Vulnerabilities and Exposures (CVE)
- Common Configuration Enumeration (CCE)
- Common Platform Enumeration (CPE)
- Common Vulnerability Scoring System (CVSS)
- Common Configuration Scoring System (CCSS)

Specifications can be used to facilitate the communication of requirements to vendor tools. More important, adoption of specifications supports automated aggregation and comparison of the tool output. Without the adoption of specifications, it is difficult to impossible to aggregate and compare system state data from a diverse set of tools. In the security domain, the use of SCAP[19] and the NVD[20] provide a variety of specifications that creating a strong foundation for policy expression and system state data normalization. CM implementations focused on security will likely leverage SCAP but not be limited to this suite of specifications. Many other specifications are in development that can support CM domains outside of the scope of SCAP.

The Collection and Analysis/Scoring subsystems will retrieve content from the subsystem. This enables the Collection subsystems to collect the correct data and enables the Analysis/Scoring subsystems to correctly analyze and score the data according to organizational policy.

[19] http://scap.nist.gov.
[20] http://nvd.nist.gov.

A CM instance is not required to have a Content subsystem, but at least one Content subsystem must exist somewhere in the organization's CM implementation. Choosing the number of Content subsystems for an organization's CM implementation is a matter of organizational policy. If all parts of the organization are required to use the exact same security posture settings, only a single Content subsystem is needed. If some requirements are universal across an organization while others can be tailored for specific environments, a hierarchy of Content subsystems could be implemented where each tier in the hierarchy would have the ability to tailor a particular set of the requirements but not all of them. Alternately, a completely decentralized approach could be implemented where only the lowest tiers have Content subsystems. This would allow each part of the organization the greatest flexibility in defining its security requirements but would pose challenges when trying to aggregate and compare CM results.

5.3 Multi-tier Capability

Large organizations often need more than one CM instance. This may be due to a need to avoid aggregating all security data into a single repository. It may also be because organizational components are structured such that they independently manage their IT, and having a collection of federated CM instances is their preferred approach. CAESARS FE supports this by enabling CM instances that are arranged in a tree structure forming a logical hierarchy or tiers (shown in Figure 11. Federated Hierarchical Continuous Security Monitoring Instance Model

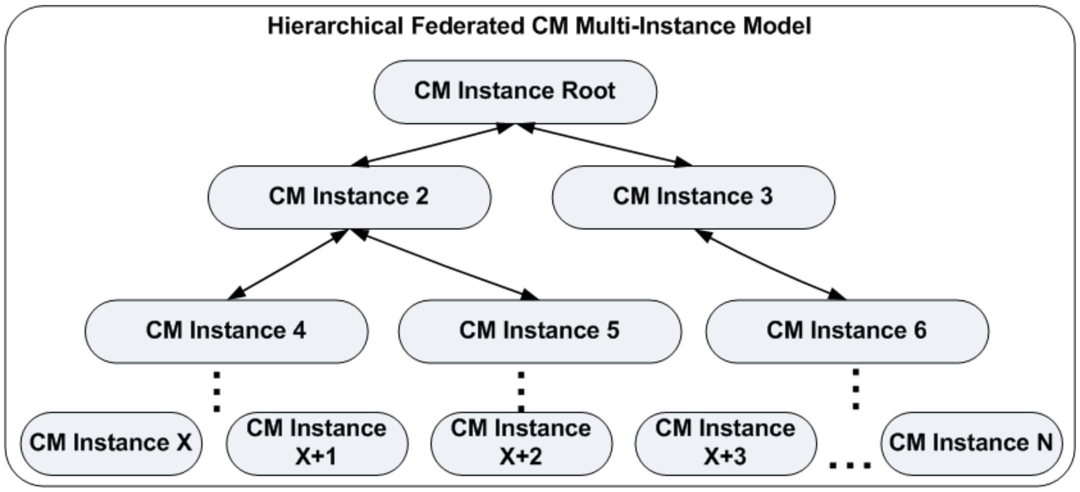

Figure 11. Federated Hierarchical Continuous Security Monitoring Instance Model

Aggregated data reports and compliance information typically travel up the tree. Data calls and security configuration requirements typically travel down the tree. This enables only limited data to be sent up the tree while enabling higher tiers to request more data as needed. As discussed previously, this model limits the bandwidth, storage, security, and data-freshness problems associated with trying to send all CM data up to all tiers in the hierarchy.

Lateral communication between nodes is possible (and supported by the lower level specifications) but is not explicitly part of the CAESARS FE model. Depending on the policy of the organization, CM instances may be given an element of autonomy, resulting in a more federated relationship while preserving the hierarchy. In this case, CM instances may require human approval of requests from higher tiers prior to releasing data or performing additional data collection. This may be important in cases where the incoming request would consume significant system resources and thus require planning and scheduling to minimize the impact of the load on the organization.

6. Supporting Documentation Architecture

To help organizations implement the CAESARS FE EA and subsystem models, additional publications outlining technical design specifications have been developed separately from this publication. This approach provides modularity that will enable us to regularly update the specifications as we tweak the design and slowly raise the bar to add new functionality Furthermore, the technical specifications will be provided within two separate publications: one that focuses on data domain agnostic specifications that apply to all CM instances and another that focuses on data domain specific CM requirements (e.g., asset management). This will enable us to revise the data domain specific requirements more frequently as we add new data domains (e.g., vulnerability management) to the CAESARS FE model. Figure 12 shows the planned publication model for supporting technical CM implementations.

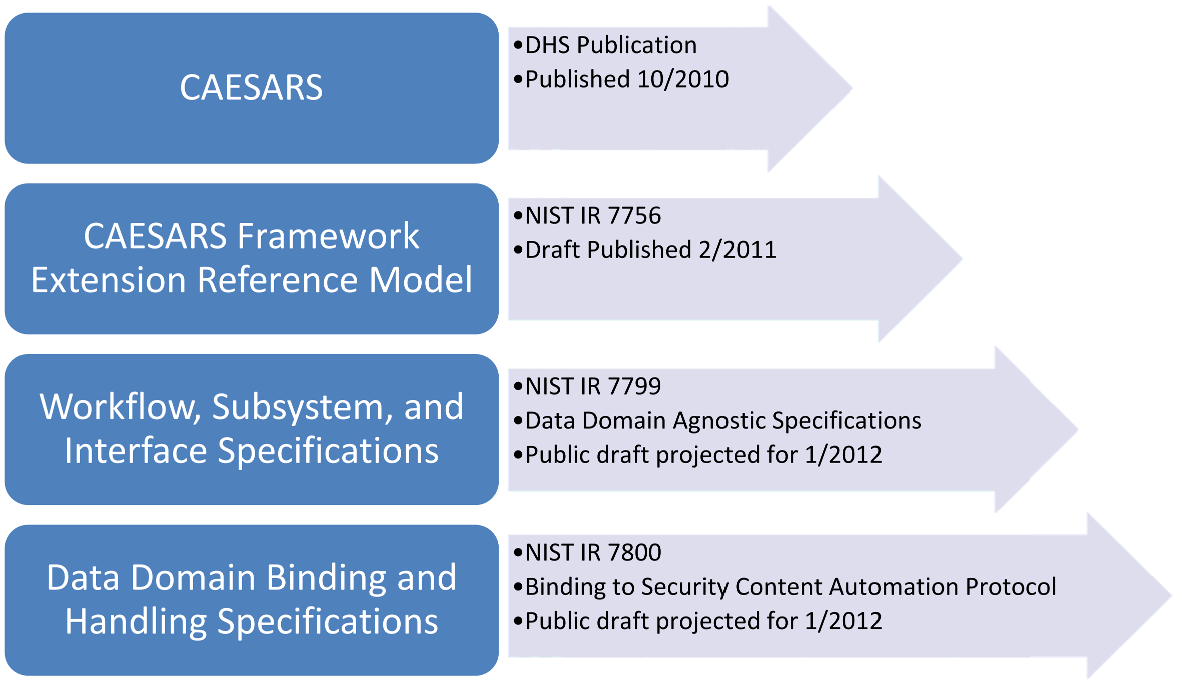
CAESARS
•DHS Publication
•Published 10/2010
CAESARS Framework Extension Reference Model
•NIST IR 7756
•Draft Published 2/2011
Workflow, Subsystem, and Interface Specifications
•NIST IR 7799
•Data Domain Agnostic Specifications
•Public draft projected for 1/2012
Data Domain Binding and Handling Specifications
•NIST IR 7800
•Binding to Security Content Automation Protocol
•Public draft projected for 1/2012

Figure 12. Continuous Monitoring Document Architecture

The data domain agnostic specifications will focus on workflows that must exist within any CM implementation regardless of the data domain being monitored. These workflows enable an organization to collect CM data and to deconflict, analyze, score, and store the results. They then enable result retrieval to answer diverse queries from multiple consumers, thus providing situational awareness. To enable these workflows, we will define required interfaces that must exist between CM subsystems and components. For each interface, we will provide communication specifications to enable interoperability between products composing a CM solution (leveraging NIST specifications such as Asset Reporting Format [ARF] and Asset Identification [AI]). Finally, we will provide specifications for each subsystem and component to describe functionality necessary to implement the workflows.

The data domain-specific specifications will describe how each subsystem, component, and interface must support specific data domains. As directed by the ISIMC CM subgroup, our initial focus will be to provide specifications for asset, configuration, and vulnerability management. These specifications serve to bind the higher-level data agnostic specifications to low-level communication specifications that are focused on particular domain areas. For the areas of our initial focus, we will strongly leverage security automation standards such as SCAP. The higher level data domain agnostic specifications will strongly but not exclusively leverage ARF and AI.

7. Conclusion

This publication provides an EA and subsystem model for implementing CM capabilities. This model builds on the CAESARS foundation to add new functionality, especially addressing the needs of large organizations.

While this model can provide benefit to organizations designing their own CM systems, it will only be fully effective when combined with the lower-level technical specifications in NIST IR 7799 and NIST IR 7800 and when vendor tools adopt those specifications. For this reason, we will be working closely with the vendor community to facilitate adoption and vetting of the specifications.

We have designed these lower level specifications with the goal of providing general functional enhancements to benefit vendor's products and their customers. Once vendor tools have adopted the CM specifications, organizations will be able to use their existing security tools to compose CM implementations. When creating such implementations, the integration costs will be dramatically reduced because of tool adoption of the specified interoperability standards. Furthermore, the CM implementations that use this model will be interoperable, making unified reporting, data analysis, and correlation possible across multiple organizations (even one as large as the entire U.S. Government).

Appendix A—Acronyms

This appendix contains selected acronyms used in the publication.

AI	Asset Identification
AO	Authorizing Official
ARF	Asset Reporting Format
ATO	Approval to Operate
CAESARS	Continuous Asset Evaluation, Situational Awareness, and Risk Scoring
CC	Collection Controller (a Task Manager component)
CCE	Common Configuration Enumeration
CCSS	Common Configuration Scoring System
CIO	Chief Information Officer
CM	Continuous Monitoring
CPE	Common Platform Enumeration
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DE	Decision Engine (a Task Manager component)
DHS	Department of Homeland Security
DOD	Department of Defense
DOS	Department of State
EA	Enterprise Architecture
ECMC	Enterprise Continuous Monitoring Capability
ESB	Enterprise Service Bus
FDCC	Federal Desktop Core Configuration
FE	Framework Extension
FISMA	Federal Information Security Management Act
HIPPA	Health Insurance Portability and Accountability Act
IAD	Information Assurance Directorate
IR	Interagency Report
ISIMC	Information Security and Identity Management Committee
ISSLOB	Information Systems Security Line of Business
IT	Information Technology
ITL	Information Technology Laboratory
NIST	National Institute of Standards and Technology
NIST IR	National Institute of Standards and Technology Interagency Report
NIST SP	National Institute of Standards and Technology Special Publication
NSA	National Security Agency
NVD	National Vulnerability Database
OCIL	Open Checklist Interactive Language
OMB	Office of Management and Budget
OSD	Office of the Secretary of Defense

OVAL	Open Vulnerability and Assessment Language
QO	Query Orchestrator (a Task Manager component)
SCAP	Security Content Automation Protocol
SOA	Service-oriented Architecture
SOX	Sarbanes Oxley
SP	Special Publication
STIG	Security Technical Implementation Guide
US-CERT	United States Computer Emergency Readiness Team
USGCB	United States Government Configuration Baseline
WS	Web Service
XCCDF	Extensible Configuration Checklist Description Format