SAMSON Technology Demonstrator Architectural Design Document

Phase IV SD002

Daniel Charlebois
DRDC-CSS

Prepared by: Bell Development Team,
Bell Canada 160 Elgin St. 17th Floor
Ottawa ON. K1S 5N4

CSA: W7714-08FE01

Contract Report
DRDC-RDDC-2013-C3
March 2013

IMPORTANT INFORMATIVE STATEMENTS

Defence Research and Development Canada

SAMSON Technology Demonstrator
Architectural Design Document
Phase IV
SD002

Bell Canada
160 Elgin Street
17th Floor
Ottawa, Ontario
K1S 5N4

Final
March 2013

Confidentiality

This document is UNCLASSIFIED.

Disclaimer

The contents of this report do not constitute original work on behalf of the author. A number of sections of the report are comprised of material contributed by multiple authors, most notably members of the Bell SAMSON team.

Authors

Bell Development Team	Role
Glen Henderson	Lead System Architect
Brent Nordin	Intermediate Programmer
Bill Pase	Intermediate Programmer
Dan Seguin	Intermediate Programmer

Bell Q&A Team	Role
Alan Clason	Testing Specialist
William Dziadyk	Certification and Accreditation

Review

DRDC SAMSON Team	Role
Bruce Carruthers	Technical Advisor
Daniel Charlebois	Scientific Authority
Darcy Simmelink	Project Manager

Table of Contents

List of Figures

1.0 Introduction

The SAMSON TD implements a new security architectural approach to provide data-centric information protection in a multiple community network environment. Applications are enabled for SAMSON data protection through the insertion of information protection security services into the applications' data handling routines. The focus of the DRDC "Technical Demonstrator" project was to develop a capability that provides separation of Canadian Eyes Only (CEO) and Canadian-US (CANUS) information caveats in a single network environment. However, the SAMSON TD functionality also allows the use of other attributes to achieve the needed data-centric protection in a multiple community network environment.

The SAMSON TD system uses cryptographically bound data tagging (metadata labels) and Attribute-based Access Control (ABAC) to enforce the required "need-to-know" security controls. Based on the policies required by the enterprise, the data is tagged with relevant attributes (i.e. classification, releasability caveats,…) and the system performs access control decisions based on a comparison of the attributes of the data object and the attributes of the person subject requesting access to the data.

The SAMSON TD system enables collapsing network-centric infrastructures and security policy based enclaves. An enterprise infrastructure, with SAMSON TD incorporated into the architecture, will enforce users to only be able to access data to which they are entitled to access. It provides a capability to transform an enterprise's multi-level (e.g. different sensitivity classifications, caveat-separations, compartmentalized, releasable to, community-of-interest need-to-know access controls, etc…) security architecture from a network-centric (i.e. separate "system high" network segments, complex multi-tier defence-in-depth network architectures, separate operational "enclaves", etc…) implementation approaches to data-centric implementation approaches. The SAMSON TD system can be implemented using traditional physical servers or virtual servers. The deployment architectures can range from small user community "Entry Level" to large user community with "High Availability" architecture environments.

Data-centric security controls provide a foundation for intelligent networking and seamless unified communications using a variety of communications methods where the operational and business requirements may require that sensitive data must be made broadly available (perhaps using cloud technologies) to end-users in operational environments located beyond the enterprise's physical network infrastructures.

The Attribute Based Access Control in the SAMSON TD is instantiated as a Service Oriented Architecture (SOA) and uses eXtensible Messaging and Presence Protocol (XMPP) based Messaging Services and Policy Enforcement Points (PEPs) to securely and reliably enforce caveat protection of data using XACML policy based Authorization.

The SAMSON TD provides the ability to protect itself and associated data from unauthorized access or modification and provides an audit trail to ensure accountability for authorized actions.

The SAMSON TD relies on six core services:

1. User Identity Attribute Management
2. Authorization
3. Secure Labeling
4. Cryptographic Protection
5. Key Management
6. Trusted Audit,

and leverages external services:

1. Authentication Services (Active Directory);
2. Endpoint Labelling Solution (Titus);
3. Key Escrow services;
4. Cryptographic Modules; and
5. Security Information and Event Management (SIEM) and

An application such as, file server, email, instant messaging, web services, and database access can be enabled for SAMSON TD data protection through the insertion of the SAMSON TD Application Policy Enforcement Points into the applications' data flow and handling routines. The SAMSON TD Application Policy Enforcement Point integrated into the SAMSON TD core services constitutes the complete SAMSON TD.

The SAMSON TD implementation does not provide user workstation endpoint security.

1.1 History

The Secure Access Management for Secret Operational Networks (SAMSON) technology demonstrator (TD) project originated from a research concept developed at the Network Information Operations (NIO) Section of Defence Research and Development Canada - Ottawa (DRDC Ottawa). The NIO, now CyberOps, section of DRDC Ottawa initiated a series of studies which produced a proposal for the creation of a new security architectural approach to provide *data-centric* information protection in a multiple community network environment. Further work developed plans for, and implemented, two Secure Access Management Proof-of-Concept Demonstrators (SAMPOC I in 2002 and SAMPOC II in 2004) based on an initial architectural design.

Although the SAMPOC demonstrations were successful in proving the technical capability of the approach and generated substantial interest within the Department of National Defence (DND), these implementations were not sufficiently robust, secure, or large enough to be considered for operational deployment. The current *SAMSON TD* project seeks to:

- address the deficiencies of the SAMPOC I & II demonstrators;
- extend the capabilities of the original SAMPOC concept to encompass a complete set of information protection services;
- raise the Technology Readiness Level (TRL) of the SAMSON concept through a more robust implementation of the extended SAMSON design; and

- Highlight, through a series of demonstrations and exercises, the technical challenges, business transformation issues and process changes that would be encountered in transitioning this technology from prototype stage to an operational deployment.

The SAMSON TD target architecture also builds upon the following research initiatives, promoted by DRDC:

- Trusted Audit Collection System Design (2004) and Prototype (2005)
- Security Policy Engine Surveys (2002 and 2006)

1.2 Purpose

The overarching objective of the SAMSON TD, as defined in the project requirements specification, is to implement and demonstrate a capability that provides separation of the Canadian Eyes Only (CEO) and Canadian-US (CANUS) information caveats in a single network environment. This report serves to describe the SAMSON TD high-level architecture.

1.3 Project Scope

The SAMSON TD project calls for the creation of a security enabling infrastructure that leverages, through open protocol standards, six core security services: user identity attribute management, authorization, security labelling, cryptographic protection, key management and trusted auditing. A specific set of applications has been enabled for SAMSON data protection through the insertion of information protection security services into the applications' data handling routines. This set includes representative applications that provide the following data services:

- File sharing,
- Email,
- Instant messaging, and
- Web content delivery.

The following aspects of the SAMSON TD are considered out of scope in terms of their inclusion in this report:

- Certification & Accreditation - Certification and accreditation is not addressed within this document. Readers interested in this subject are encouraged to consult the *SAMSON TD Certification and Accreditation Plan*.

- Complementary Security Services – The SAMSON TD outlines six core security services required for caveat separation. While these security services provide the functionality required for caveat separation, they are not intended to address all

aspects of security. Consequently, there are other security services, most notably threat detection and response, which are complementary to the six core SAMSON security services. These complementary security services are considered outside the scope of the report.

The SAMSON TD requirements are defined in the specification document *SAMSON TD Functional Specifications V2.2*. Development activities for the demonstrator were defined across three phases, with a functionally complete system to be delivered at the end of the first phase. The Phase I capability target was extended in February 2010 to include specific enhancements to support participation in the EC2010 exercise. The Phase II capability target was delivered for and demonstrated at two military exercises:

- Empire Challenge 2011 (hosted at both DRDC Shirley's Bay and Fort Huachuca, Arizona); and
- Coalition Warrior Interoperability Demonstration 2011 (CWID).

The Phase III capability target was delivered for and integrated into the operational network for the Coalition Attack Guidance Experiment II (November 2012), again hosted at the Warfare Center at Shirley's Bay, Ottawa.

The documentation set to which this Architectural Design belongs uses as its reference architecture the SAMSON TD deployment that was utilized in the Coalition Attack Guidance Experiment (CAGE) II operational SECRET network environment. The requirements set for this reference architecture was a union of:

- A subset of the capabilities defined in the SAMSON TD functional specification that defined the phase II target functionality; and
- Application support, robustness, stability and administration enhancements needed to support CAGE II participation.

This documentation set includes the following:

- SD002 - the *Architectural Design Document* (this document)
- SD004 - the *Detailed Design Document*

A complete mapping of the SAMSON Phase II demonstrator capabilities to SAMSON TD functional specification is provided in the *Requirements Traceability Matrix* which is included as a supplement to this documentation set.

Those readers interested in a more theoretical discussion of SAMSON design principles are encouraged to consult *Authorization: An Historical Perspective* and *The Bell SAMSON Architecture & Backgrounder*.

2.0 The SAMSON Design Philosophy

Current trends in information technology are fostering a need for security solutions that can work at the data level. For example, cloud computing and cloud storage are predicated on the idea that data owners must trust external third parties to be the reliable custodians of information assets. Similarly, device agnosticity, the growth and power of mobile computing such as tablets, means that data is finding its way onto devices over which organizations have less control. Even within the bounds of data processing capabilities, the interest in data mining and particularly data fusion tools is evidence of a need for security tools that can protect information at a granular level.

The focus on data-level security is occurring in the context of a fundamental schism in the mandate to which government and military networks operate. Specifically, as data owners and custodians of information in the national interest, security officers are required to enforce the *need-to-know* principle: only users with an express policy right should have access to that information. Conversely, the requirement for coordination between agencies and multi-national coalitions is driving the need for the ability to exchange information in a *need-to-share* context. It is the SAMSON premise, that current methods for information assurance that encompass network and application centric security models do not have the flexibility, granularity or adaptability to achieve the needed balance between these conflicting demands.

SAMSON is a data-centric security solution. That is, SAMSON is a system of information protection that cryptographically binds security metadata to the information asset. This idea is complementary to the concept of “smart data”, that is, data that carries with it its own security policy in terms of access and acceptable use restrictions.

This project is examining the concept of data-centric security and its application to create the next generation of secure networks though its ability to address four basic challenges in information assurance.

1. Create unified security policy across all information assets: SAMSON allows for the *specification, application and enforcement of a unified and holistic security policy across all information assets*. Operationally, such a security policy may be an aggregation of policies (e.g. mandate-level, mission-level and departmental-level policies) but once unified, this policy would be applied to all information assets being processed within that domain. The implication here is that the format of the data to be protected is not a barrier to the application of SAMSON protection mechanisms. However, the closer an application adheres to open and published standards, the easier is the task of applying data protection at the asset level.

2. Create flat sharing environment while maintaining need-to-know separation and control: SAMSON restricts operations to be performed on information assets to specific communities. By applying a security policy across all data sets, it is possible to create communities of interest (COI) that have a shared level of privilege over specified assets. Membership within a COI becomes a fluid concept since the joining and expelling users

from a community is a simple process for privileged users that have the policy right to manage community memberships. This policy enforcement over SAMSON user identity attributes and entitlements is an example of a higher order information protection operation, namely, the SAMSON security protection mechanisms also protect SAMSON policy, audit and identity administrator access control. SAMSON protecting SAMSON is a core concept of the design and leads to a clear definition of the security target for certification activities. In a like vein, SAMSON provides the same level of protection over its own security policy.

The result is that there are several trusted ways that a piece of information can be shared:

1. The identity attribute administrator can add the user to the community that has access to the information;
2. The policy administrator can add a new policy that grants the user access to the information; and
3. The data owner can change the security attributes on the data that makes it accessible to a community to which the user belongs.

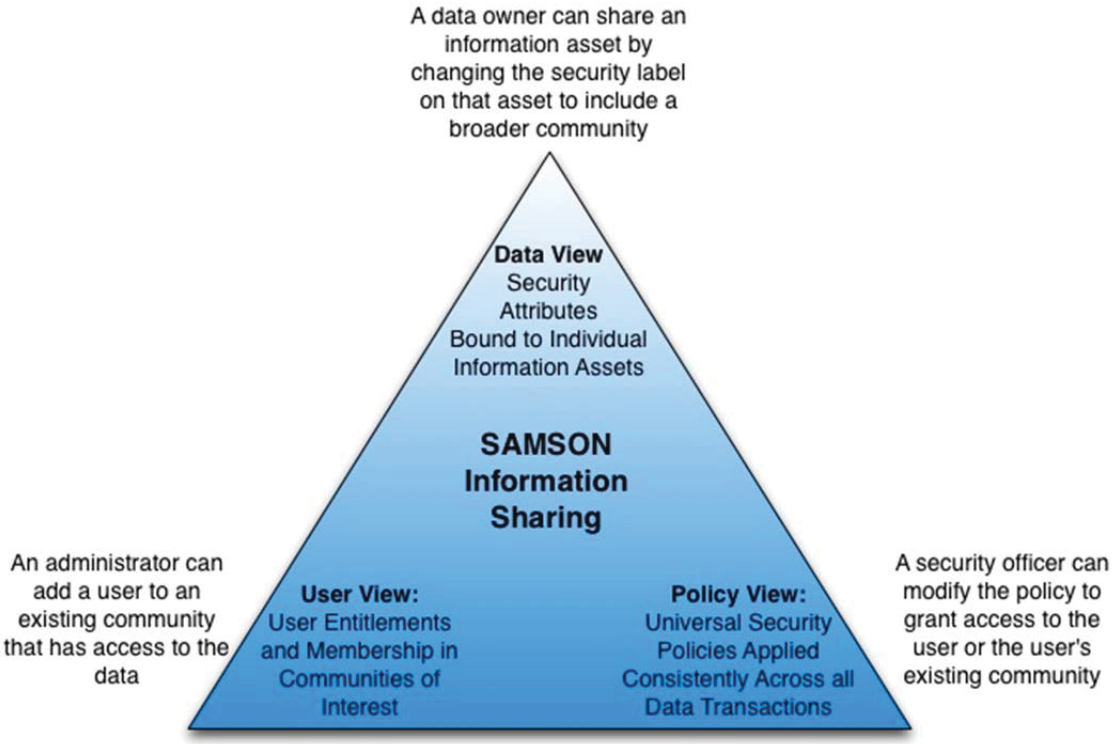

Figure 1: Information Sharing in a SAMSON Context

The fact that there are three possible methods to enable sharing of information assets with each method able to be performed by a different (and audited) administrative role, illustrates the flexibility of the SAMSON architecture and how it enables the *need-to-share* principle.

3. Cryptographically maintain control of data to users who have authorized need-to-know: SAMSON ensures that information is only released to those users that have a policy right to access it. This access restriction is achieved through the use of cryptography. Information assets that have been protected by SAMSON are individually encrypted with a unique symmetric key that is only available through the SAMSON infrastructure. In this way, the only way to obtain and apply the key that was used to protect the data is through the SAMSON protection processes, which includes the valid security policy check. Only after SAMSON has determined that the user has the right to access the data is the cryptographic transformation applied which allows the user to gain access to the information.

This is a significant improvement over existing data protection infrastructures where operational administrative activities such as backups, migration and upgrades require that low-privilege system administrators have direct access to information for which they do not have a *need-to-know*. In the SAMSON model, system administrators have full access to the cryptographically protected data asset only and can perform maintenance operations on those assets without the need to gain access to the content itself. This also has direct application infrastructures that leverage cloud data storage or similar constructs; information can be handed over to these agencies with confidence that the content itself is cannot be disclosed since the cryptographic keys remain with the local SAMSON deployment.

These two concepts form two of the basic principles of SAMSON design:

1. The release of information assets to an authorized user is a SAMSON policy decision that is based on the user's community memberships and the organization's security policy; and
2. Having SAMSON grant the right to access information is the only manner by which the cryptographic transformations to release the data are performed.

4. Audit all access decisions in a trusted audit log: SAMSON provides a trusted audit facility that records the details related to the release of information in a tamper-resistant form. The process within the SAMSON information handling routines where access to data is granted is the ideal point where an audit record can be created that captures the conditions, environment and rationale by which the data disclosure was made. Within the SAMSON design, audit records are generated, submitted and stored in a trusted manner that provides a *chain-of-custody* of evidence of user and system behavior.

By providing the means to share, at a granular level, data assets between users and communities but maintaining strict control over how data access is granted and maintaining a trusted record of those transactions, SAMSON achieves the needed balance between the need-to-know and the need-to-share.

2.1 The SAMSON Approach

This section presents the operational nature of the solution and the architectural composition of a security system built on SAMSON principles. This composition includes: SAMSON as a security overlay, its modularity and its service definition

SAMSON is a *security overlay*, that is, a set of interconnected services that communicate through the exchange of messages on top of an existing network deployment. Any network security or application security based environment can be enabled for data-centric protection without the need to remove or de-emphasize existing security protections. In this way, an environment protected by SAMSON retains the existing safeguards that were in place prior to the deployment of the security overlay. Additionally, SAMSON itself is able to leverage existing physical, administrative, network and application safeguards as part of its deployment profile. Most significantly, the deployment of SAMSON does not change the underlying accreditation of the target network by modifying the security architecture that was initially certified. SAMSON when deployed to a SECRET network inherits the SECRET level protections and does not expose additional threats to the environment that would place SECRET assets at risk.

Therefore it is possible to describe a network both before SAMSON has been deployed and the subsequent additions and modifications to the environment that were instantiated as part of a SAMSON deployment. The needed changes to the environment to support a SAMSON data-centric solution:

- Do not interfere with the existing processes, practices or safeguards of the originating environment; and
- Can be encapsulated and described as a formal change, that is, a detailed statement of the additional software services that are to be deployed to the target environment.

This change to the target environment to support data-centric security forms the basis of the SAMSON security overlay, which includes:

- The addition of new security services that provide SAMSON capabilities in the form of a service oriented architecture; and
- The redirection of network traffic to pass through SAMSON data processing modules for the application of data protection services.

It is significant to note that the security overlay can be deployed to an existing environment without forcing the use of new applications, changing data processing routines or disrupting the current user experience. As such, the SAMSON solution must be able to protect, without modification of, existing applications, their data formats or the data exchange protocols that are used. SAMSON achieves this transparent nature through the use of an application level proxy architecture that is able to process per-application data protocols. In this context, however, proxies are not meant to protect applications or services, but to be the sole point in the information request/response cycle where content can be released. It is only at the application proxy that the security policy is enforced and that cryptographic transformations are applied to data that allows the data to be released to the requestor. In no other context is information released rendering the data protected from all unauthorized forms of disclosure.

The SAMSON architecture must be modular in that existing services should be able to be replaced with components that provide equivalent capabilities without the need to re-architect the solution or deployment. Each SAMSON service can be viewed in three ways:

1. The SAMSON core security services are an information service that can be called upon by other SAMSON components for the delivery of specific data or transformational activities.
2. The SAMSON core security services provide an interface between the SAMSON overlay, where security messages are exchanged over the SAMSON SOA and the external component that is delivering the actual service.
3. The service is the backend capability that is already present in the environment and is leveraged by SAMSON to fulfill its function in the SAMSON architecture

This can be best described as an example. The SAMSON SOA requires information about users' identities in order to evaluate information requests in the context of the current security policy. In many cases, the target environment will already have a COTS IdM solution in place that is used to manage user attributes. All SAMSON components that require identity information will call the SAMSON Identity Attribute Service, which will, in turn, call out to the target environment's IdM solution for the needed information.

In this example, the message handling through the SAMSON SOA over a security messaging layer and the bridging function that links SAMSON services to their external counterpart are SAMSON components. The external component itself, in this case the COTS IdM, is not part of the SAMSON architecture although SAMSON leverages the external component to fulfill its own information requests and work with the existing enterprise security solution.

As a security overlay, SAMSON uses the pre-existing security software that is present in the environment. This leads to two significant observations about SAMSON:

1. SAMSON is a secure communications infrastructure that allows many different security solutions to be integrated in a comprehensive data-centric security architecture; and
2. Since SAMSON bridges to external security software for the delivery of services, it is possible to change the back end security software without disrupting the SAMSON SOA deployment. The architecture itself remains agnostic to the actual services that are being leveraged to deliver SAMSON data-centric security.

Based on the description of SAMSON data-centric security and the example of a SAMSON Identity Attribute Service it is appropriate at this point to describe those core services that form the basis of the SAMSON architecture.

The following diagram illustrates the layers of protection in the form of a suite of security services that a SAMSON-enabled application will leverage to apply information protection to its information handling functions. A SAMSON-enabled application thus becomes a policy enforcement point where SAMSON security services are used to achieve the data-centric information protection goals.

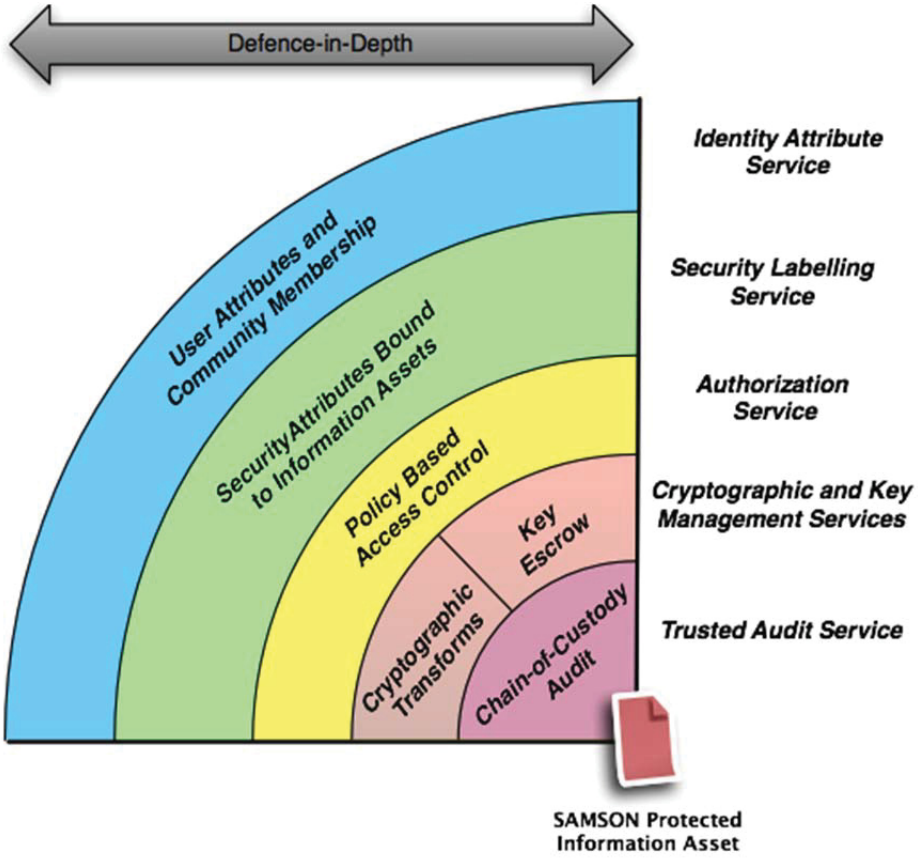

Figure 2: The SAMSON Defence-in-Depth Model

These six services can be individually described with respect to the role they play in both protecting data assets and enabling the sharing of information that is in accordance with the organization's mandate and operational requirements for collaboration.

Identity Attribute Service: A user's identity is enhanced with attributes defining that user's membership in specific Communities of Interest (COIs). This service allows SAMSON component to query these attributes, which are stored within the organization's enterprise IdM repository

Security Labelling Service: Information objects are also defined by security attributes encapsulated within the security label that has been bound to and stored with the data. This service applies, extracts, verifies and validates labels on data objects that are protected by the SAMSON security architecture.

Authorization Service: Drawing from the attribute information of both the subject (user) and the target (object), and including ancillary environmental information, the Authorization Service can apply sophisticated decision functions to instantiate complex security policies that are the norm in operational environments.

Cryptographic Transformation Service: SAMSON protected assets are stored in an encrypted form. When a user has the policy right to access an object, SAMSON decrypts the asset prior to delivery. In this way, the only means to access a SAMSON protected object is through SAMSON services which will require a policy check to be performed and an audit records to be generated for each data request. The CTS is the SAMSON component that provides these encryption and decryption capabilities.

Key Management Services: This service is responsible for the creation, storage, and retrieval of cryptographic keys. Each SAMSON protected asset is encrypted with its own unique key and the KMS provides all symmetric key management functions that are needed by SAMSON cryptographic services.

Trusted Audit: A tamper-resistant record of policy decision activities is made in support of short term Security Incident and Event Monitoring (SIEM) and long-term forensic activities.

When viewed in this context, the SAMSON services work together to support a defence-in-depth principle: a series of security components that provide a layered approach to protecting information assets. (i.e. policy protections, cryptographic protection and audit activities). As a security overlay, this architecture leverages environmental protection to have an even stronger defence-in-depth protection that includes:

- Systemic hardening at the host level;
- Network segmentation and network path protection; and
- Cryptographic protection of information at rest and in motion.

It is also important to recognize that these six services represent only the initial selection of security capabilities that are provided by SAMSON and new services can be added to meet new or emerging security concerns.

2.2 SAMSON Information Protection

The SAMSON design philosophy calls for the protection of data objects as they are processed or delivered by applications within the operational environment and to do so in a manner that minimizes the required customization of these applications to leverage SAMSON protection services. Modification of the applications themselves is deemed to be counter to the SAMSON design philosophy as it ties the architecture to a specific application implementation, forces integration with proprietary solution components, requires that application support become part of the SAMSON development cycles, and deviates from the leveraging of industry standards.

The SAMSON design philosophy calls for data object access mediation to occur along the communication path between the subject and the target application service. At this interception point, SAMSON security services, including its Authorization Services, are called upon to enforce the security policy in a reliable and trusted manner.

In summary, the SAMSON architecture can be examined using the NEAT model: a construct that is used to define a high assurance system.

SAMSON information protection components and processes are:

- **N**on-bypassable: SAMSON intercepts traffic between the workstation and the target data service. Information requests that comply with the security policy are allowed to proceed. While traversing the interception point, data is cryptographically transformed. Only valid requests can traverse the intercept and any attempt to access the data directly will only disclose an encrypted object.
- **E**valuatable: Each SAMSON component is implemented as a well designed, well specified, well implemented, small, low complexity module that is accessible through a well defined, open protocol. It is possible to do assurance testing against each SAMSON interface through the use of validation and verification harnesses
- **A**lways-invoked: Every SAMSON-relevant data request is checked by the appropriate security monitors and information protection services. The selection of what constitutes a SAMSON-relevant data request is entirely defined by the implementation. SAMSON does not place restrictions on what actions can be policy validated.
- **T**amperproof: The system generates an audit trail for all security relevant events that is established through a chain-of-custody. This capability detects the addition, deletion or modification of audit trail information. While this does not provide proof against tampering, it does support the detection of unauthorized modification of the audit records in support of incident handling and forensic activities.

3.0 The SAMSON Architecture

Architecturally, SAMSON mediates access to information assets as a security overlay over an existing operational network. SAMSON leverages the concept of a reference monitor in the form of a policy engine that receives information access requests from other SAMSON components, evaluates the request against the domain's stated security policy and returns a policy decision for enforcement.

This reference monitor is implemented based on a standard called eXtensible Access Control Markup Language (XACML), from Advancing Open Standards for Information Society (OASIS). Through this policy engine and application-specific policy enforcement points, SAMSON gates access to information assets since all actions against protected data are first vetted and audited by the SAMSON security components.

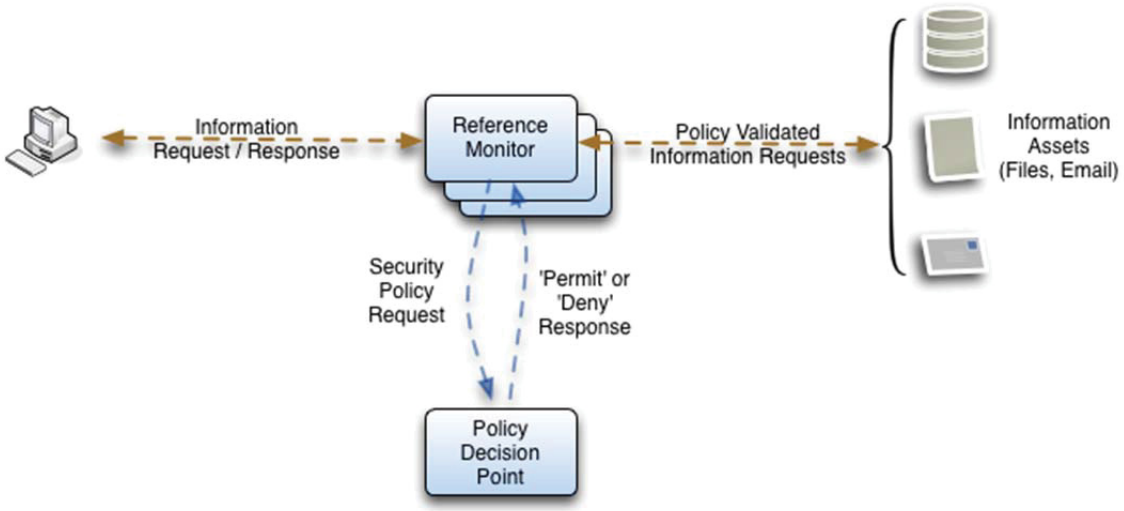

Figure 3: The SAMSON Reference Monitor

Achieving this policy based, data-centric security model is achieved through the mapping the defence in depth security services onto the XACML model and leveraging of ancillary security services in the target environment that each have a specialized role within the security architecture. The major XACML 2.0 components: Policy Decision Point (PDP), Policy Enforcement Point (PEP), Policy Information Point (PIP) and Policy Administration Point (PAP) and ancillary services are shown in Figure 4: XACML Components.

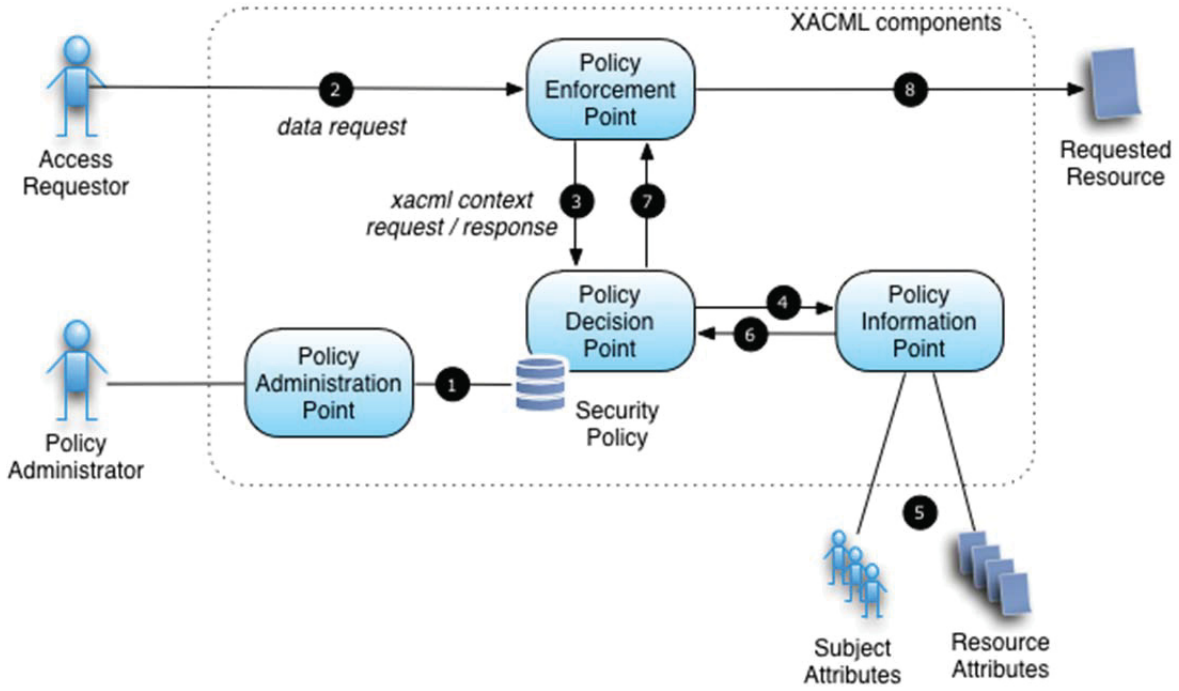

Figure 4: XACML Components

A clear alignment can be seen between the SAMSON information protection philosophy and the XAMCL security policy expression and enforcement standard.

XACML Component	SAMSON Architectural Component
Policy Decision Point (PDP)	The SAMSON *Authorization Service* provides an interface to a PDP to apply a single unified security policy against information access requests.
Policy Information Point (PIP)	The SAMSON *Identity Attribute* and *Security Labelling Services* act as policy information points, retrieving security attributes bound to users and data, respectively, that can be used as supplemental and context-specific information to assist in the determination of a policy decision.
Policy Enforcement Points (PEPs)	SAMSON Policy Enforcement Points intercept data requests between the user and the target service. These PEPs apply the policy decision returned from the Authorization Service and perform information protection transformation on data to protect the data against unauthorized disclosure by leveraging the *Cryptographic Transformation* and *Key Management Service*.

XACML Component	SAMSON Architectural Component
Policy Administration Points (PAPs)	SAMSON, as a security overlay, is designed to leverage existing COTS-based security products such commercially available PDPs, audit and identity management solutions. As a result, a target environment would use the administrative services provided by those products. However, SAMSON can supplement the security protections on administrative interfaces by applying the reference monitor protection model to those interfaces. In the same way it protects data assets, a PEP can gate access to all administrative interfaces including the PAP.

SAMSON goes beyond the traditional XACML model by incorporating *Trusted Audit Services* and *Cryptographic Transformation Services* into the design of the PEP architecture, ensuring that a tamper-resistant audit trail is kept of all user initiated transactions.

The Cryptographic Transformation Services protects information against unauthorized disclosure through cryptographic protections that occur as part of the data handling routines as information passes through SAMSON policy enforcement points as shown in Figure 5 SAMSON overlay on to an Existing Network.

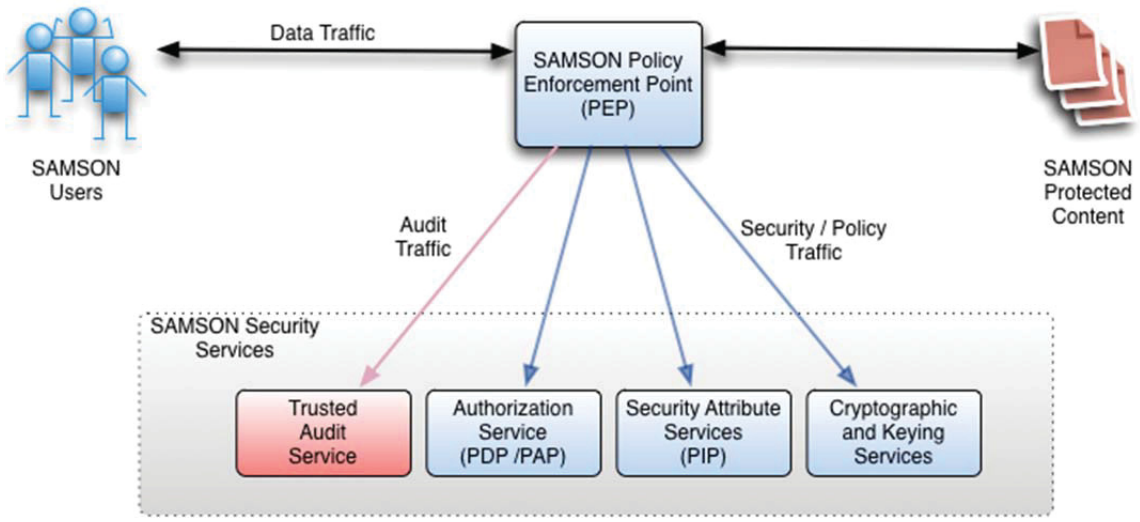

Figure 5 SAMSON overlay on to an Existing Network

3.1 The SAMSON Service Oriented Architecture

The SAMSON architecture is implemented as a Service Oriented Architecture (SOA) whereby all security requirements are met by independent services that are accessible through open, well-defined interfaces. The information exchange formats within this architecture utilize industry accepted, open standards that are based on the eXtensible Markup Language (XML).

It is important to recognize that the architectural goal of the SAMSON infrastructure is to provide a common set of security interfaces and the ability for data handling applications to leverage those services for a universal application of the domain's security policy. Conceptually, the SAMSON security services act as security gateways with the ability to route an internally generated SAMSON security service information request to the actual external service or process that will handle the request. SAMSON is, therefore, not tied to any specific vendor solution and can replace any vendor solution with another product that provides similar capabilities.

In this way, all SAMSON security interfaces conform to a common set of design goals, including:

- They can be made to work with any external vendor solution, product or implementation;
- They can be extended to include any SAMSON-specific capabilities that are not reflected in the chosen standard;
- They are appropriately secured in order to ensure the confidentiality and integrity of these information exchanges; and
- New security services can be added to the architecture without the need to redesign or redeploy the entire security overlay.

The SAMSON architecture achieves its data protection requirements through the use of three core architectural components:

1. **A Secure Messaging Service Bus (SMSB)**: The ability for SAMSON components to exchange data in a manner that is secure, protocol agnostic, and reliable.
2. **Security Services Gateways**: The ability to bridge between the SAMSON security architecture and the back end (non-SAMSON) applications that provide the needed security functionality. These services include authorization for adherence to policy, cryptographic services for protection of data and audit for the creation of a trusted chain of evidence.
3. **Policy Enforcement Points (PEPs)**: The ability to link external application and security services to the SAMSON infrastructure in a manner that adheres to the SAMSON security protection principles.

These components can be seen in their proper context in Figure 6: SAMSON Core Components.

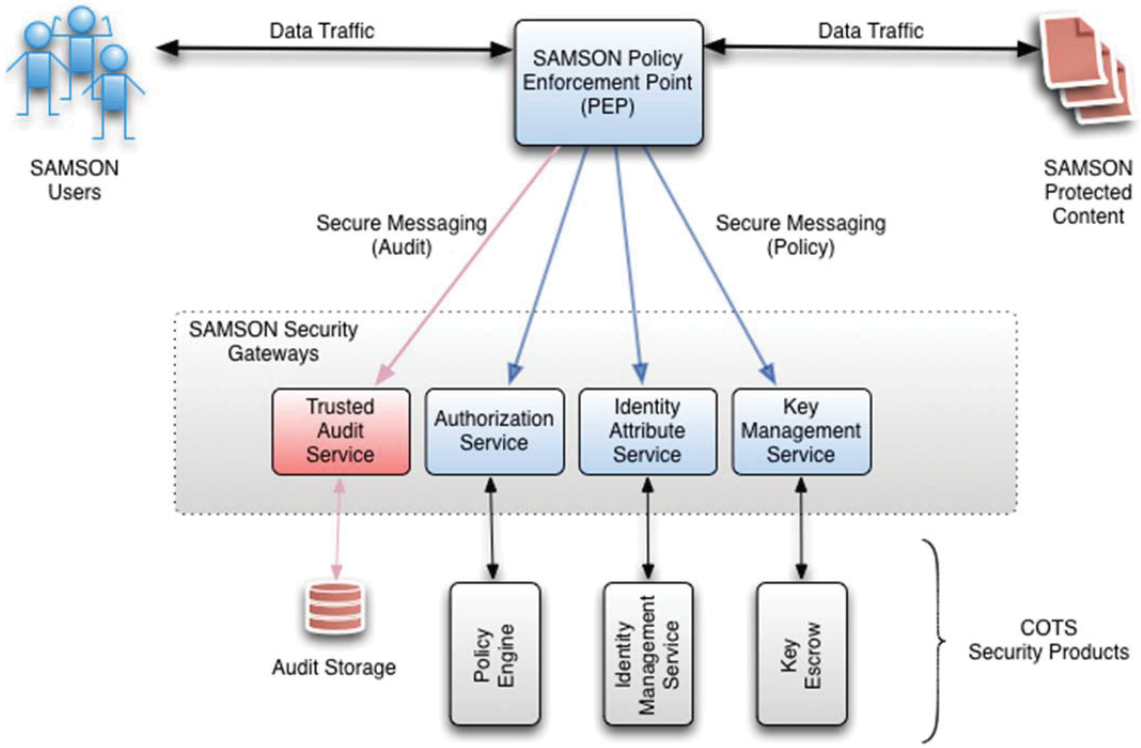

Figure 6: SAMSON Core Components

The SAMSON policy engine: and the security policy it enforces work with the security gateways and PEPs to ensure that only information to which a user has a right to access is delivered to the endpoint. As illustrated in Figure 3: The SAMSON Reference Monitor, policy decisions are based upon the user's community memberships and the caveats contained within the security labels that are tied to the information that has been requested. If a user does not have the policy right to perform actions on the data, the operations on data will not be performed and the information remains protected against disclosure.

The SAMSON project is an enabling infrastructure that allows security services to be leveraged by applications through a trusted messaging system. While there are specific SAMSON components that are mandatory to a SAMSON deployment, additional services can be added to the SAMSON capability set, allowing SAMSON to be an effective paradigm for protecting information assets in a variety of environments.

The following sections discuss these architectural components in detail. Section 4.0 describes the SAMSON security services that have been implemented as Security Service Gateways.

3.2 SAMSON Messaging Infrastructure

Within the SAMSON infrastructure, any entity that communicates, either to request or provide security services, does so using industry-recognized protocols. While the protocol messages themselves form the necessary protection context to secure information in the SAMSON environment, it is the responsibility of the SAMSON messaging core to provide the delivery of these messages between SAMSON components. Although the specific protocol or format of the message content will depend on the nature of the entity or service being leveraged, all messages are delivered through the same communications mechanism. As a result, the architectural specification differentiates between the message transport mechanism and message content itself.

All components that communicate SAMSON security messages do so via the message delivery mechanism. Every component that must send or receive security messages will be a participant on this messaging infrastructure, including:

- The Application interfaces that intercept data access requests, formulate policy queries based on the requested information, submit the policy requests and respond appropriately to the policy decision;
- The Authorization Service which responds to information access requests and returns decisions based on the domain security policy;
- The ancillary Security Services which add additional context and processing to the policy enforcement process such as the retrieval of identity information and the use of cryptographic services; and
- The Trusted Audit Service which provides a record of all information request transactions.

With the responsibility to ensure robust, secure and trusted delivery of security messages between SAMSON components, the delivery mechanism is a critical core of the SAMSON architecture.

The SAMSON architecture uses the eXtensible Messaging and Presence Protocol (XMPP) as the delivery mechanism for the exchange of SAMSON security messages. XMPP is designed to support low-overhead, message-based communication over persistent sessions and is suitable for delivering encapsulated messages to authenticated entities in a secure and robust manner. XMPP is designed to be agnostic about the message payload, allowing it to carry any SAMSON message content as necessary to support the SAMSON security services. Additionally, XMPP provides the following capabilities, which are necessary to achieve trusted message delivery.

- **Authentication**: Prior to joining the messaging infrastructure, a component must be authenticated using the chosen authentication service. XMPP authenticates its participants using Simple Authentication and Security Layer (SASL). SASL supports strong authentication using public-key based credentials.
- **Persistent Communications**: XMPP is a connected synchronous communications protocol and is, therefore, resistant to man-in-the-middle or hijacking attacks.

- **Low Overhead**: XMPP is designed to handle XML messages with very low overhead that will allow SAMSON to scale to large deployments.
- **Presence Detection**: XMPP facilitates resource discovery since presence detection is built into the protocol. Resource discovery is essential to making the system scalable and reliable once operational.
- **Encryption**: XMPP sessions can be encrypted with Transport Layer Security (TLS). Encrypted sessions will not only protect the delivery details, but the message content as well. The TLS session between the XMPP clients and servers will be established using single key pair credentials issued for this specific purpose. The XMPP server will perform Certificate Revocation List (CRL) checking as part of the client authentication process.

XMPP's inherent capability for establishing presence makes it an ideal choice for supporting an SOA as it natively provides identification, monitoring, and delivery of information services within an open standard format. XMPP is complementary to the SAMSON design philosophy as it is an open standard and is XML based. Open XML standards also make it possible to extend existing XMPP capabilities to include features that are needed by SAMSON in an operational context.

3.2.1 *Messaging Architecture Guidelines*

While the SAMSON architecture makes no requirement on where message traffic is hosted, security best practices including: data isolation, network zoning and load balancing would recommend a deployment of SAMSON traffic along the following physically or logically segregated networks. The use of multiple Secure Messaging Service Bus (SMSBs), that is, multiple XMPP message domains, can be used to separate SAMSON traffic in the following manner:

<u>Security Policy SMSB</u>: An XMPP domain that supports the exchange of security information traffic such as attributes, policy requests and decisions and cryptographic key information between SAMSON components.
<u>Audit SMSB</u>: An XMPP domain that supports the delivery of audit messages between SAMSON components and the Trusted Audit Service.

Components only connect to the domains for which they have an architectural need to exchange messages. While the use of separate XMPP domains provides a logical separation of traffic, these domains can also benefit from physical separation. Specifically, a SAMSON deployment can attain a strongly defensible security posture through the use of separate physical networks for:

1. *Management Traffic*: The connection used by system administrators to manage SAMSON components.
2. *Data Traffic*: The network over which users connect to gain access to data services. This is the pre-existing operational network in a target deployment.
3. *Security Traffic*: The network that hosts the Security Policy SMSB.

4. *Audit Traffic*: The network that hosts the Audit SMSB.

It should be observed that an additional benefit of using separate networks for SAMSON security messages is that a SAMSON security overlay does not incur a large increase in the amount of traffic on the existing data or operations network.

3.3 SAMSON Messaging Components

All SAMSON components that exchange data on the SAMSON messaging network follow the same architectural design from which all application PEPs and security gateways are based. Using the baseline design, each component, as part of its processing of SAMSON messages, is able to:

- Act as a bridging element between internal and external services;
- Translate protocols;
- Transform the data as needed, including cryptographic transformations;
- Perform policy checking against the SAMSON security policy; and
- Submit trusted audit records.

The selection of the ability defined above is, however, very dependant on the requirements of the component being implemented. For example, policy evaluation is a function of an application PEP whereas protocol translation is only done by security service gateways.

3.3.1 *Security Service Gateways*

Security Service Gateways are participants on the SAMSON security message bus where security messages such as: policy requests, caveat information requests, and cryptographic transformation service requests are exchanged between components. These components also extend an interface outside of the SAMSON infrastructure, leveraging non-SAMSON security elements to deliver SAMSON security messages and services. For example, SAMSON might leverage the services of an Identity Management (IdM) Service that resides outside the SAMSON architecture. Calls from inside the SAMSON environment to acquire information from the IdM would traverse a gateway which bridges the gap from the originating request to the IdM itself.

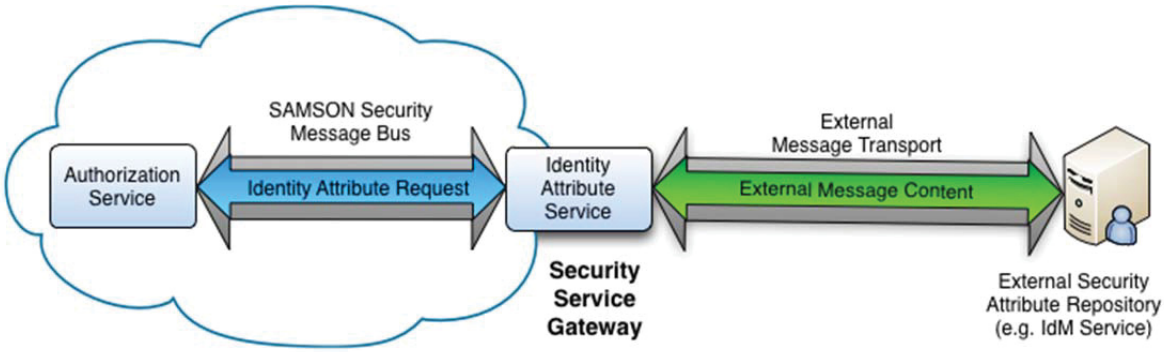

Figure 7: A Security Service Gateway (SSG)

In Figure 7: A Security Service Gateway (SSG), we see that component that exchange messages over the security message bus, that is to say the component that are using the SAMSON SOA, messages are exchanged using the data format that has been defined for internal communication between services. An SSG accepts information requests from inside the SAMSON SOA and redirects those requests to the eternal service that is providing the service. For example, as a SSG it is the responsibility of the IAS to broker identity attribute request messages between the SAMSON environment and the external security service that is providing the needed functionality. This behaviour is shared between all SSGs, although the message format used by each SSG will be specific to the service it is providing.

Additionally, as a bridging component between internal and external interfaces, the security gateway can act as a translator, converting the internal, open (non-proprietary) SAMSON message content formats into the format that is used by the external component for accessing its information or security services. As a result, any external service can be utilized in the SAMSON processing environment, regardless of the protocols that are being used and the external service protocol can be changed without requiring changes to the internal SAMSON messaging architecture.

An alternate view of the bridging and translating abilities of a security service gateway is to examine the protocol that is being processed by each function.

The *bridging* function acts on the *message delivery protocol* (e.g. a transport protocol like HTTP) and since external services are not part of the SAMSON messaging infrastructure the bridging function is a *mandatory* element for security service gateways.

The *translation* function operates on the *message content* itself (e.g. SPML) mapping the internal information request format to the format supported by the external service. In cases where the external service is using the same security data protocol as that used within the SAMSON architecture, message translation is *not necessary in all cases*.

3.3.2 *Policy Enforcement Points*

Whereas security service gateways bridge and translate protocols between SAMSON and external security services, PEPs operate on data requests by ensuring the transactions adhere to the security policy, transforming the data as it is delivered and auditing the actions that have been performed against the data request.

Application PEPs are composed of two primary subcomponents:

- An application data network service proxy (discussed below); and
- A SAMSON messaging client that connects the PEP to the XMPP domain(s) to allow the PEP to draw upon and utilize SAMSON security services.

Since the fundamental goal of SAMSON is to provide information protection for data in a network environment with a pre-existing set of deployed applications, SAMSON must be able to operate on the information request formats that are currently used by those applications. In the SAMSON architecture this is achieved with application proxies that intercept and apply SAMSON information protection logic. This logic is the order in which SAMSON services must be leveraged to adequately protect the information, such as: authorization services, cryptographic services, audit services. This logic will depend on the type of data being protected and the operation on the data is being requested.

A SAMSON application proxy is placed between an end user workstation and the information resources that user is attempting to access. Note that when accessing data through a SAMSON protected resource, the user still retains use of the existing applications and the user's experience, in terms of the application activities and processes that are performed, remains unchanged. What defines a user as a SAMSON user is the fact that a set of identity attributes has been assigned to that user. These attributes are used as part of the policy checks that determine what actions the user can perform against information assets.

As an application proxy, the SAMSON PEP has access to all information that is needed to submit a policy request to the Authorization Service and to submit an audit record to the Trusted Audit Service. This information set includes, but is not limited to:

- Who is asking for the information asset;
- What are the users access control privileges
- What is the caveat on the data being accessed;
- What operation on the data is being requested; and
- What is the general environment. (e.g. what time of day, physical location)

In Figure 8: PEP Data Mediation Process, a user is requesting information from SAMSON protected data store. The data requests go through an application PEP that intercepts the request and submits a policy decision request to the Authorization service.

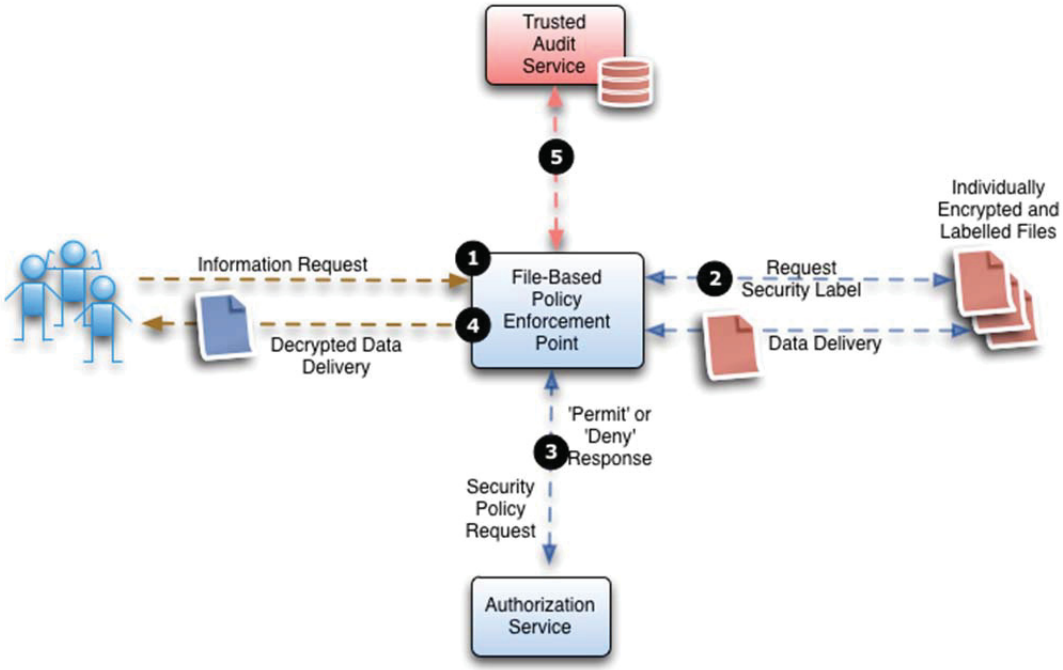

Figure 8: PEP Data Mediation Process

The PEP will (1) examine the originating request to determine the user's identity and the action requested and will also (2) call the SLS to look ahead to the data being requested to acquire the security attributes in the security label that have been bound to the target information. Combined with any environmental data needed to make the policy decision, the PEP has sufficient information to (3) submit a policy request to the SAMSON Authorization Service. Depending on the policy decision that is returned to the proxy, the data is either made available to the user or denied.

As an application proxy, the PEP calls the CTS to transform the data in accordance with the SAMSON information processing logic. SAMSON transforms all information assets to individually encrypted objects that can then be stored on the native host-network server; each protected using a unique symmetric key. When access to an information asset has been granted, the object must be (4) decrypted by the CTS prior to release to the end user. The decryption process is requested by the PEP by calling upon the SAMSON Cryptographic Transformation Service.

The final step in the information protection logic dealt with the (5) creation of an audit record. Being the data interception point, the point where policy is enforced and the point where transformation on data is performed makes the PEP the most appropriate place to generate an audit event regarding the transaction. A failure of any SAMSON component to provide a valid response during the processing of this transaction would similarly be recorded with this transaction request.

When presented with an information access request, the PEPs exercise *information protection logic*: a series of security transactions that leverage SAMSON security components that are performed in a specific order to ensure that the processing of the request is done in a manner that both ensures that the security policy is applied and that the end user receives the information should the policy allow it. The information protection logic is different for each PEP given the nature the data being requested. For example, when a user requests a file, the file handling PEP will execute the following functions:

1. Call the Security Labelling Service to obtain the security attributes of the target;
2. Call the Authorization Service for a policy decision;
3. Call the Cryptographic Transformation Service to decrypt the object for the end user; and
4. Send an audit record to the Trusted Auditing Service (TAS) to create a permanent record of this transaction.

At any point along the information processing logic, the PEP can abort the request, for example, in response to an error received by another SAMSON component. The PEP would in this case, call the TAS to record this failure and inform the end user that the transaction cannot be performed. The error returned to the endpoint software application depends on the flexibility of the software at the endpoint. For example, with email, an error email is generated by SAMSON describing the error generated. In addition, depending on the nature of the failure, a message is also directed to the environment SIEM solution to alert security or operational staff.

The PEP will audit the full transactional history of an information request:

- What was the original request;
- What SAMSON components were leveraged in the processing of this request;
- What were the results of intra-SAMSON security service requests;
- What result was returned to the PEP; and
- What decision was enforced.

From this information, it is possible to determine and re-create a rationale for why a particular data request was permitted or denied.

SAMSON Audit messages are similar to other SAMSON message content: XML-based records that may be transmitted by the same security message bus, or have its own dedicated audit bus depending on the security posture of the target infrastructure.

As previously stated, the security architecture can benefit from the use of two separate and isolated Secure Messaging Service Bus's: one for security policy messages and one dedicated exclusively to hosting the audit information. This physical or logical messaging bus provides a separation of traffic that enhances user privacy and the robustness of the audit forensic activities. The decision to use one or multiple SMSB's is a deployment decision; the SAMSON architecture does not dictate the need for multiple messaging networks. However the use of logically or physically separate networks for management, data, policy and audit traffic greatly enhances the deployment security posture.

4.0 Core Security Services

The SAMSON core services, those services that are implemented as security service gateways and provide defined security capabilities according through a specific data protocol interfaces, are defined below:

- Section 4.1:Authorization Service
- Section 4.2:Identity Attribute Service
- Section 4.3:Key Management Service
- Section 4.4:Cryptographic Transformation Service
- Section 4.5:Security Labelling Service
- Section 4.6:Trusted Audit Service

The application-centric security components are discussion in subsequent sections, including:

- Section 5.0:Application Protection: Policy Enforcement Points
- Section 6.0:Self-Protecting SAMSON Services

Each of these core security services as shown in Figure 9: SAMSON Architecture are implemented as part of the SAMSON TD program and are aligned with the SAMSON design philosophy of:

- Modular component integration;
- Universal and consistent service use through the SAMSON solution; and
- Service access through open, non-proprietary interfaces.

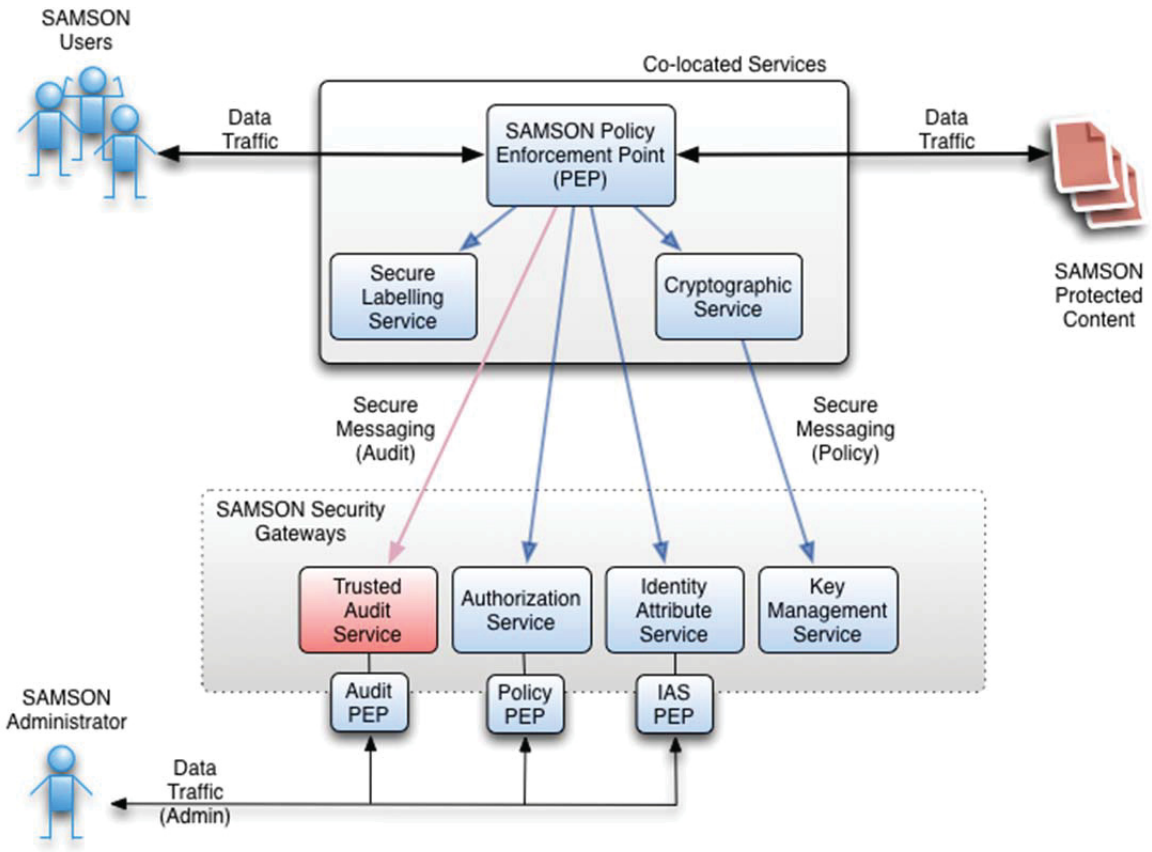

Figure 9: SAMSON Architecture

Other security services are present in the SAMSON operational environment and contribute to the overall SAMSON solution architecture while not necessarily following, at this stage, the SAMSON design philosophy as detailed in the SAMSON Functional Specification. For example, user authentication is a security service that is leveraged directly by SAMSON and it not accessed through a security service gateway. Since SAMSON is currently targeted for Windows-based security domains, it leverages the operating system, application-level and single sign on (SSO) authentication capabilities that are already present in the target environment.

The SAMSON TD uses the Windows Domain Authentication Service for account authentication at the desktop. In order to gain access to applications services that are hosted in the domain, all users are required to log in to their local workstation. The credentials acquired from the Windows domain controller are used for all information access requests that are mediated by the SAMSON Authorization Service. The use of the Windows Domain Authentication Service supports the SAMSON design principles of:

- Supporting the unmodified DND workstation baseline configuration; and
- Providing a Single Sign On capability.

In the future, a fully SAMSON compliant authentication service could be created that would leverage any backend authentication service such as LDAP, NIS or NIS+.

The remainder of this section discuss each of the core SAMSON security services.

4.1 Authorization Service

The central core security service for the SAMSON demonstrator is its Authorization Service. This service is a reference monitor, a software module that mediates all software access to data objects and verifies that the access requests are allowed by the access control policy. Data objects, in the SAMSON architecture, can be information assets, data services, session objects, or real-time sessions.

The Authorization Service module dictates the conditions under which an entity is authorized to access a protected object by interpreting a single, universal security policy in the context of a policy decision function.

Information access requests are intercepted by the application PEPs. The information request details at the PEP is formulated into a policy request message and transmitted to the Authorization Service that is acting as a PDP. The Authorization Service applies its policy engine logic to derive a policy response that is returned to the PEP for enforcement. This policy decision message exchange is transmitted via the SAMSON security message bus using an appropriate message content format, such as the eXtensible Access Control Markup Language (XACML).

At a minimum, the Authorization Service is composed of:

- The Authorization Service SAMSON messaging client; (the security service gateway)
- The Authorization Service Policy Engine;
- The Policy Storage facility; and
- An administrative interface for modifying the policy.

These four components (in blue) are to be deployed in the following configuration.

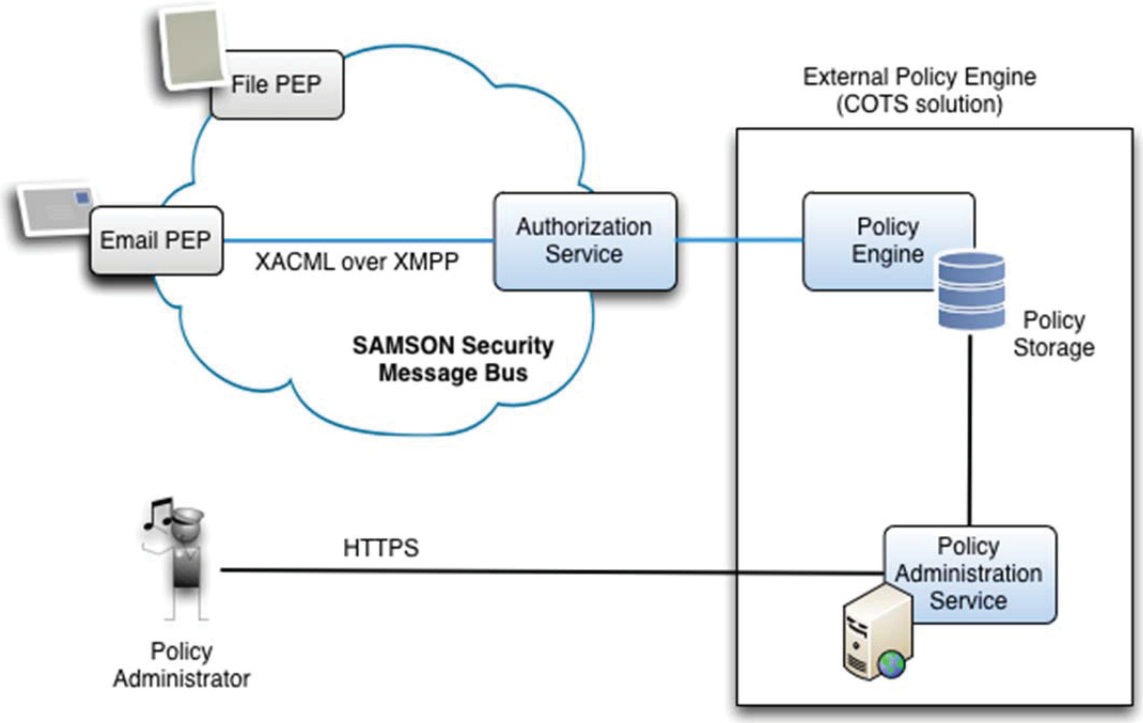

Figure 10:SAMSON Authorization Service

The SAMSON Authorization Service is designed to work with pluggable components: it is possible to replace the policy engine, storage facility and administrative tools with any security policy solution so long as the policy gateway is able to bridge to the back-end third party policy (e.g. COTS) product. This ability to work with a back-end security product is inherent to the SAMSON design philosophy for SAMSON security service gateways.

While not explicit in the SAMSON design, proper security architecture practices call for security modules to exercise the principle of least privilege, returning an explicit denial unless there is a specific policy right to access the information.

4.2 Identity Attribute Service

The SAMSON Identity Attribute Service (IAS) manages the security attributes of SAMSON users within the deployed environment. At a minimum, the SAMSON IAS Service is composed of:

- The IAS and its messaging Interface;
- The COTS-based back end IdM service or application;
- The user security attribute storage facility; and
- An administrative interface for modifying the user caveat information.

These four components (in blue) are to be deployed in the following configuration.

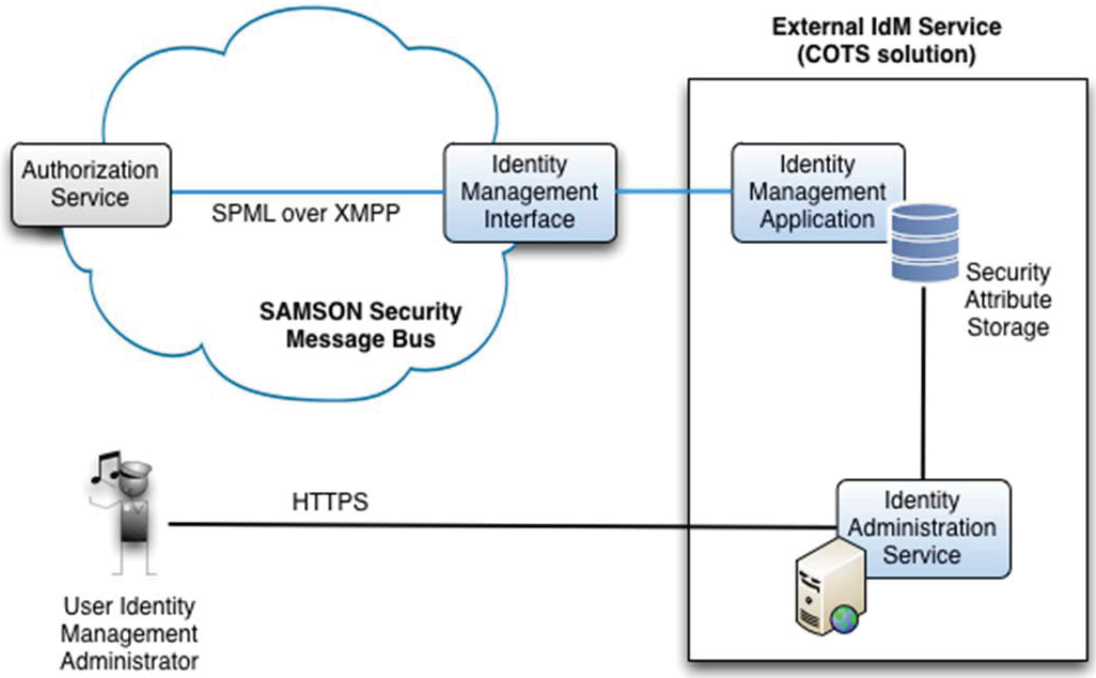

Figure 11: SAMSON Identity Attribute Service

The Security Attribute Storage facility is the central repository for all SAMSON security information associated with users in the SAMSON environment. This may be the primary IdM repository for all user information or a separate repository used exclusively for SAMSON security information that is maintained and synchronized with the user identity authoritative source. However, the user security attribute storage facility is the de facto authority for the determination of the security attribute information used during the evaluation of SAMSON protected data requests.

The interface presented by the IAS is used by other SAMSON components to acquire user attribute information from the user security attribute repository. This is most notably seen in the Authorization Service's need to obtain community of interest membership information from the IAS as part of the policy decision process. All SAMSON components that require information from the IAS formulate a request using the IAS message format (SPML and DSML) and transmit this request over the SAMSON security message bus.

A web-based interface is used to set and maintain user's security attribute information. In a deployment scenario where there is a separate repository for user security attribute information versus user identity information, synchronization between repositories will be required. While this synchronization would be performed by the IdM itself, outside of the SAMSON messaging services, the authoritative nature of the various sources must be maintained.

4.3 Key Management Service

The SAMSON Key Management (KMS) Service manages the generation, storage and retrieval of symmetric keys that are used to cryptographically protect individual information assets. At a minimum, the SAMSON KMS is composed of:

- The KMS Service messaging client; (the security service gateway)
- A Key Escrow Service;
- A Key storage facility; and
- A Key generation facility.

These four components (in blue) are to be deployed in the following configuration.

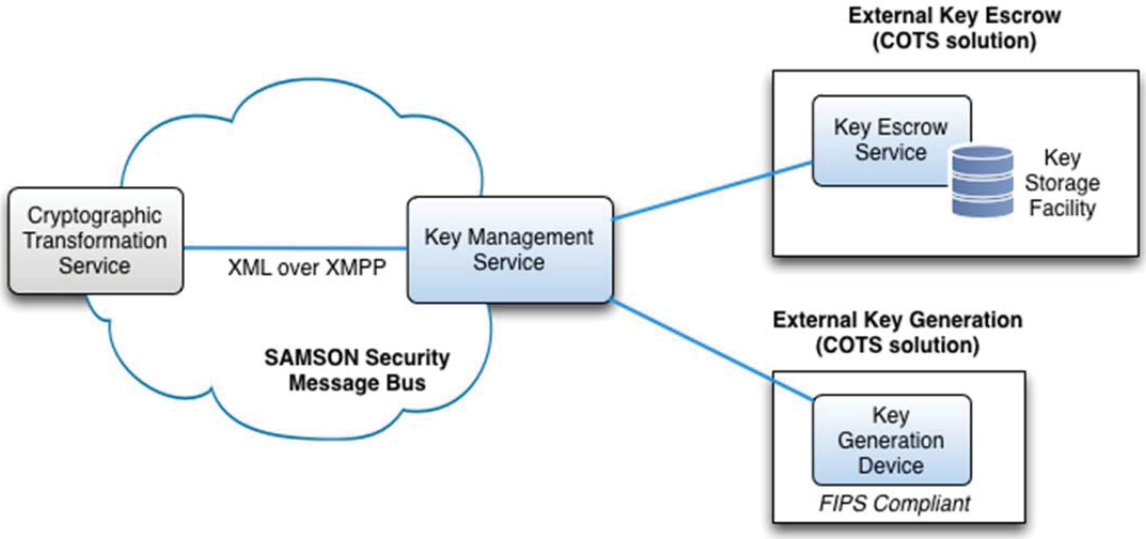

Figure 12: Key Management Service Architecture

This service is called by any component that requests access to the cryptographic keys that are used to protect information assets. Typically, the only components that would require keying information are the locally deployed instantiations of the Cryptographic Service at the various PEPs in the deployed SAMSON security overlay. The KMS receives requests to generate keys and can act as a gateway to reform and redirect that request to a key generation facility (e.g. a FIPS compliant library). The KMS also receives requests to store and retrieve keys, based on a unique token lookup value. This token value is stored with the protected data asset so that the PEP is able to ask for the correct key when requesting a decryption. Outside of the key storage facility, there is no linkage between the token and the key that could be used to derive the key through non-SAMSON means.

4.4 Cryptographic Transformation Service

The SAMSON Cryptographic Transformation Service (CTS) is responsible for performing the actual transformations on data assets response to a directive from SAMSON to protect or unprotect the asset. The CTS is the only client service of the KMS. The CTS is composed of:

- The CTS Service messaging client; (the security service gateway)
- A Cryptographic library or module.

These two components (in blue) are to be deployed (in multiple instantiations) in the following configuration.

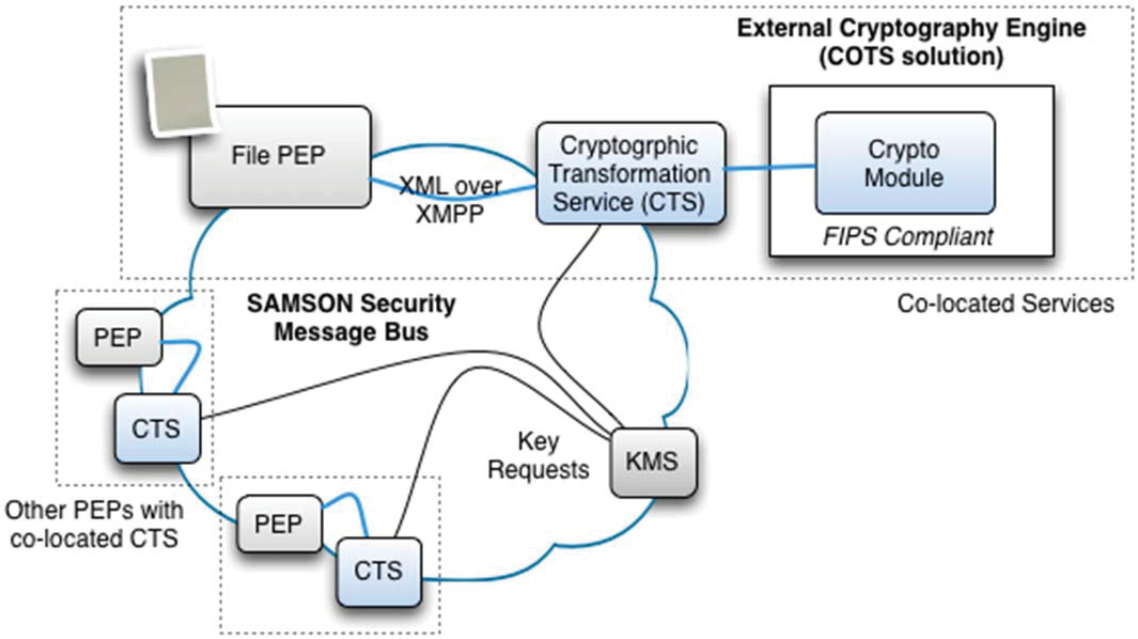

Figure 13: Use of Cryptographic Services by PEPs

This CTS service receives an instruction, a reference to a data asset, a reference token for a key and target location for the result of the transformation. The instruction will dictate whether this is an encryption or decryption action. The CTS will use the key retrieved using the token to perform the targeted action on the data asset and place the result at the designated location.

Unlike the previously described core services, the CTS must be co-located where the originating data asset is stored. For example, an application PEP that must cryptographically protect a data asset will call, using the SAMSON secure messaging bus, the locally deployed CTS using a local reference to the file. In this way, SAMSON follows a principle whereby *services are moved to the data* and not vice versa. There is no need to transmit a file to the CTS, the local CTS is able to see and protect the local copy of that file.

This results in less exposure for the file since it does not need to be sent, there are never multiple copies of the file present and strict local access control can be exercised over the file where it resides. This also means that a SAMSON deployment will have one AS, one IAS and one KMS, but will have multiple instantiation of the CTS; one for each deployed application PEP.

The CTS can bridge to any cryptographic library, including libraries that have a FIPS certification. The actual selection of the cryptographic library will depend on the attitude towards risk and certification in the target environment.

4.5 Security Labelling Service

The SAMSON architecture has the ability to operate on data assets that have security labels attached to:

- Files;
- Email messages; (and, individually, files attached to email messages)
- Instant Message Chat Rooms; and
- TLS protected web sessions.

These labels are attached to data objects either through explicit assignment of a security label by the data owner or through an automated assignment of an appropriate caveat by a context aware labelling function. Given the different formats that a data object can take, there is no single SLS that can handle all data formats, however, all SLS and SLS functionality will have a consistent behaviour. Taking the file-based SLS as an example, the SLS instantiation that works on files, the SLS is called with the name of a data asset for which the security label is needed and with specific instruction on what information is needed from the label on the data. Specifically, the SLS must be able to:

- Extract the label;
- Verify that the label has not been tampered with; and
- Validate that the label is correct for the data to which it is bound.

Verification of a label is a check to ensure that the cryptographic binding that ties a security label to a document has not changed. The validation of a label is achieved by applying heuristic analysis against the content to ensure that the label is appropriate for the content, for example, a scan for specific sensitive words.

Each of these calls can be done separately since they involve more sophisticated processing. For example, a when listing a directly, it may not be necessary to verify the integrity on labels, but when a file is requested, the label must not have been tampered with prior to its release.

As with the CTS, the SLS must work directly on the data itself and, therefore, must be co-located where the originating data asset is stored. Each PEP will, therefore require an SLS deployed as part of the PEP installation.

4.6 Trusted Audit Service

The SAMSON demonstrator includes an audit service that receives and stores, in a tamper-resistant and therefore trusted manner, audit event information from the deployed SAMSON security overlay. The SAMSON Trusted Audit Server (TAS) manages the collection, processing, storage and analysis of the audit records that are generated as SAMSON processes a request for SAMSON protected information assets.

At a minimum, the SAMSON TAS is composed of:

- The TAS Service messaging client; (the security service gateway)
- The TAS audit event processing logic;
- The TAS notification processing logic;
- The TAS audit store;
- The TAS integrity analysis tools; and
- The TAS analysis interface.

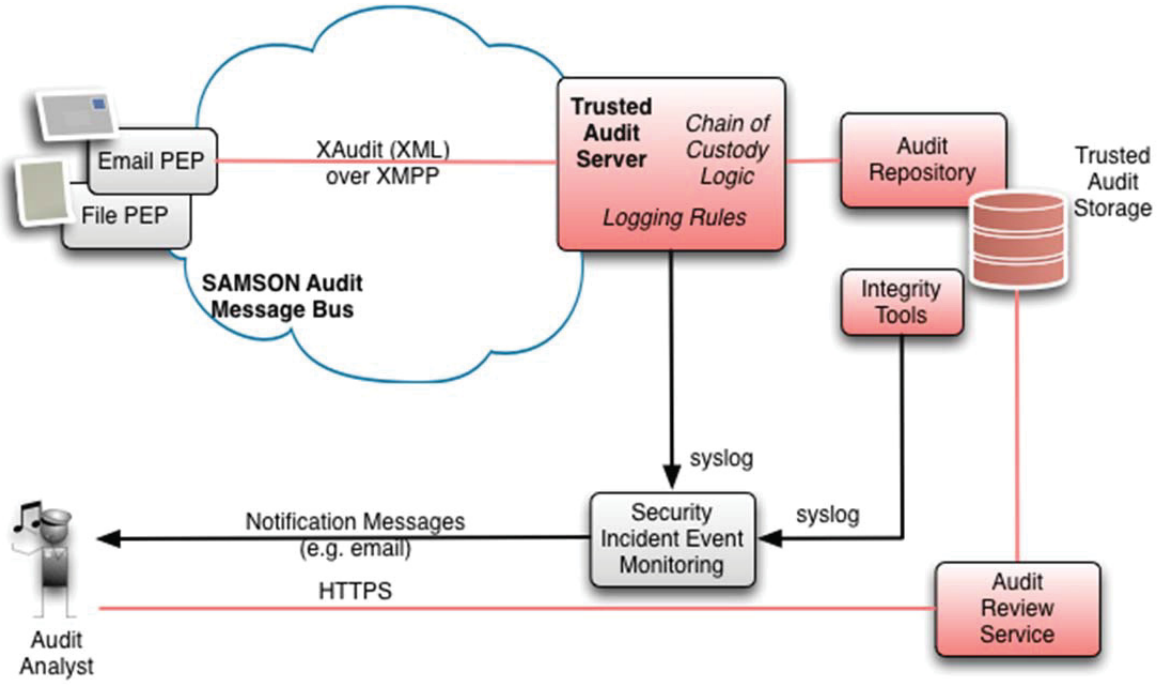

Figure 14: Trusted Audit Service Architecture

The TAS service receives the audit messages from the application PEPs, which includes the full description of the request, the policy decision result and the intermediary results from all SAMSON components that were leveraged in order to derive the result that was returned to the application PEP. Each recoded transaction is a snapshot of the policy decision as well as a rationale for why a particular decision was made; a rationale that can be examined in a forensics context.

The TAS audit event processing logic expands upon the audit data to include the record and block chaining data that form the chain of custody. This transformational logic also includes the ability to detect security incidents that need to be flagged to domain security officers through a standard event logging mechanism. In terms of preventing malicious activity against the target environment, SAMSON is uniquely placed to detect attempts to send illegal instructions, tampering of data, and other suspect activity. Processing of notification messages is not part of the SAMSON architecture, but SAMSON, by leveraging known interfaces, can work within enterprise SIEM solutions.

Audit records are stored, in their chained format, in an external repository. In this way, modification, deletion or addition of audit records can be detected where it resides in the storage repository. Supplemental tools that are needed to review the audit records include:

- Integrity tools that can be run periodically against the audit records to detect tampering; and
- Review tools that can search, sort and filter audit records in support of incident investigation and forensic activities.

5.0 Application Protection: Policy Enforcement Points

A network environment that is protected by the SAMSON security overlay will have SAMSON enabled application proxies positioned between the user workstation and the back end data service that is serving data to the user's application. In that location, the application proxies are able to monitor the information request/response cycle. By intercepting the information flow, SAMSON can be invoked to mitigate access to data and only release information when the request is compliant with the security policy. The software components that intercept information request are part of the application PEP architecture in that they:

- Interpret the information that is being requested;
- Formulate the information request in terms of a policy decision request;
- Send the policy request to the Authorization Service;
- Respond to the decision that has been returned in a policy decision response
- Perform any needed transformations on the data; and
- Audit the actions that were taken in the course of processing this information request.

Since the communication protocols and data formats vary depending on the type of application that is to be protected, the proxy portion of the application PEP will vary in its implementation. However, the general application proxy architecture for intercepting data, the leveraging of the defined core security services and information protection logic that defines how SAMSON processes data requests, leads to the complete picture of the SAMSON security architecture.

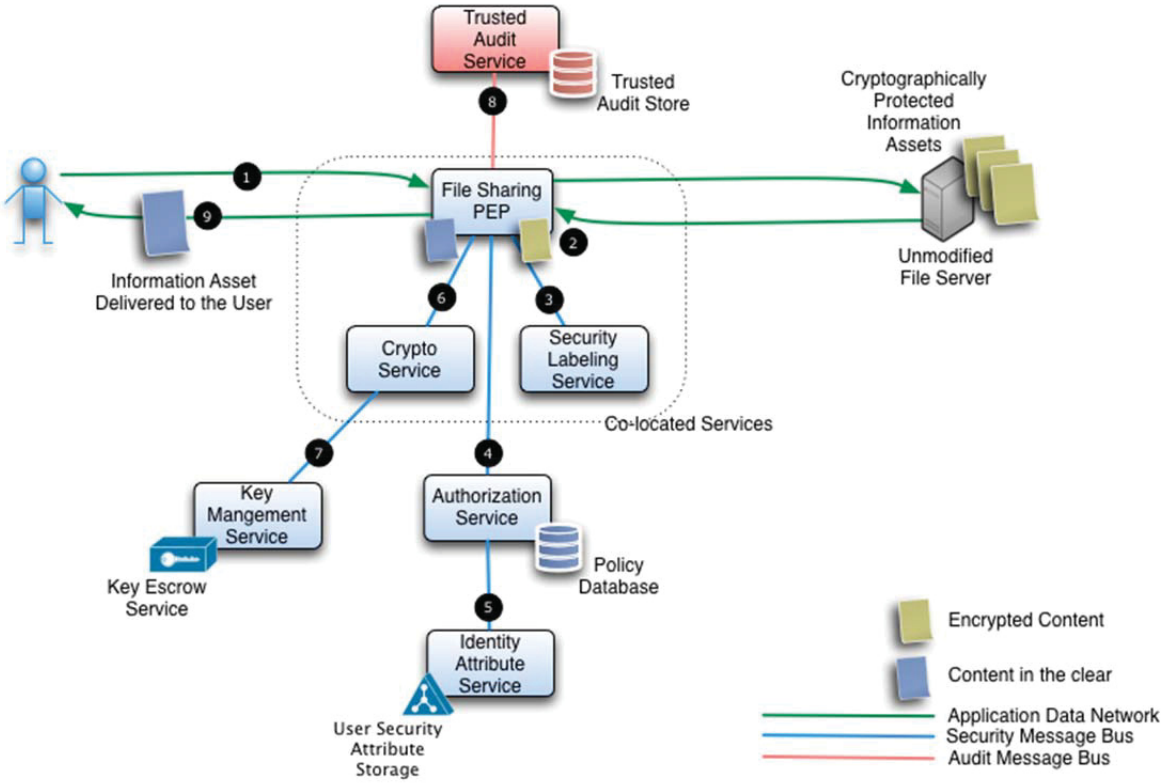

Figure 15: Detailed SAMSON Information Handling Process

The above diagram provides a detailed interpretation of the SAMSON processing steps involved in an attempt to retrieve a file from a SAMSON protected file store.

1. Over an authenticated session, the SAMSON user submits a request to retrieve the file to the SAMSON file sharing PEP using the file management tools that are present in the target environment (e.g. Windows File Sharing)
2. The SAMSON File Sharing PEP retrieves a copy of the file, in encrypted form, to the PEP for local processing. Any temporary or working files are removed and the associated memory locations zeroized at the end of the transition processing cycle.
3. The SAMSON labelling service, co-located with the File Sharing PEP with access to the local copy of the encrypted file, is called to extract the security label on the target file
4. With the identity of the user, the security attributes of the file and the requested action on the file now known, the PEP calls the Authorization Service to determine if the transaction is permitted as per the domain security policy
5. The Authorization Service calls upon the identity attribute service to retrieve security attributes of the user, including the user's membership in communities of interest. This information is returned to the Authorization Service which then evaluates the request against the policy and returns a decision to the PEP.
6. Where the transaction is permitted, the PEP calls upon the Cryptographic Transformation Service to decrypt the file prior to delivery.

7. The CTS calls upon the Key Management Service using the token that is stored with the data asset to retrieve the key that was used to protect the file as it was originally protected by SAMSON.
8. After performing the decryption process on the data asset, the file is ready to be delivered to the end user. However, prior to delivery, an audit record of the transaction is submitted, along the Audit Message Bus, to the Trusted Audit Store and, subsequently, to the Audit Store.
9. The originally requested file is then delivered to the end user.

Unlike the SAMSON security services, which are exclusively connected to the SAMSON security or audit message bus, the SAMSON application services must maintain multiple independent communication channels:

1. On the DATA network, the application proxy communication channel that monitors the traffic between the source and target of the information request (i.e. the user and the data); and
2. On the SAMSON security message bus, the PEP is able to leverage the SAMSON security services for data mediation and protection. If the chosen deployment architecture calls for security and audit messages to be hosted on separate networks, a third connection is needed to connect the PEP to the Audit Secure Message Bus.

Using this generalized architecture, the SAMSON security overlay is able to provide information protection for a wide range of applications as defined below.

File Sharing: SAMSON intercepts file access attempts, including directory listings and operations on individual files. The SAMSON Application file sharing service will make policy decisions based upon the identity of the user requesting access to the file, the label on the targeted file, and the operation that is being attempted.

Instant Messaging: SAMSON intercepts the creation and joining of chat rooms and only allows those rooms to which a user has a policy right to create or access, respectively. Messages sent via the IM server remain protected while awaiting delivery and are only accessible to users that are leveraging the SAMSON security overlay. Chat room history is similarly stored in encrypted form. Individual messages can be sent in a sub-channel within the chat room, that is, encrypted with a unique key and restricted to a smaller community of users within the chat room

Email: SAMSON intercepts email messages in transit and enforces the security policy to ensure that the originator has the policy right to send the message and its attachments. SAMSON will, for each recipient, ensure that the message and its attachments can be received. Email messages will not be delivered to recipients for whom access to the information will be a policy violation. Messages are stored in encrypted form while awaiting delivery.

Web Services: SAMSON protects web services by gating access to the web interface, only allowing users with a policy right to access the service to submit web requests via that interface.

6.0 Self-Protecting SAMSON Services

Based on the description of the SAMSON architecture to date, two items are significant to note:

1. Core Services, such as the Identity Attribute Service, the Authorization service and the Trusted Audit Service, each have web-based administrative interfaces for the assignment of user security attributes, the expression of security policy and the ability to review audit records, respectively.
2. SAMSON has the ability to protect web sessions through the deployment of an application PEP that can gate access to secured web interfaces.

From these two points it can be seen that SAMSON is able to protect it own administrative interfaces using standard policy enforcement processes. For example, by placing a PEP between the user community and the administrative interface, these interfaces become additional information assets that are protected by SAMSON. In such a case, administrative functions are restricted to users that hold the security attributes for administrative roles.

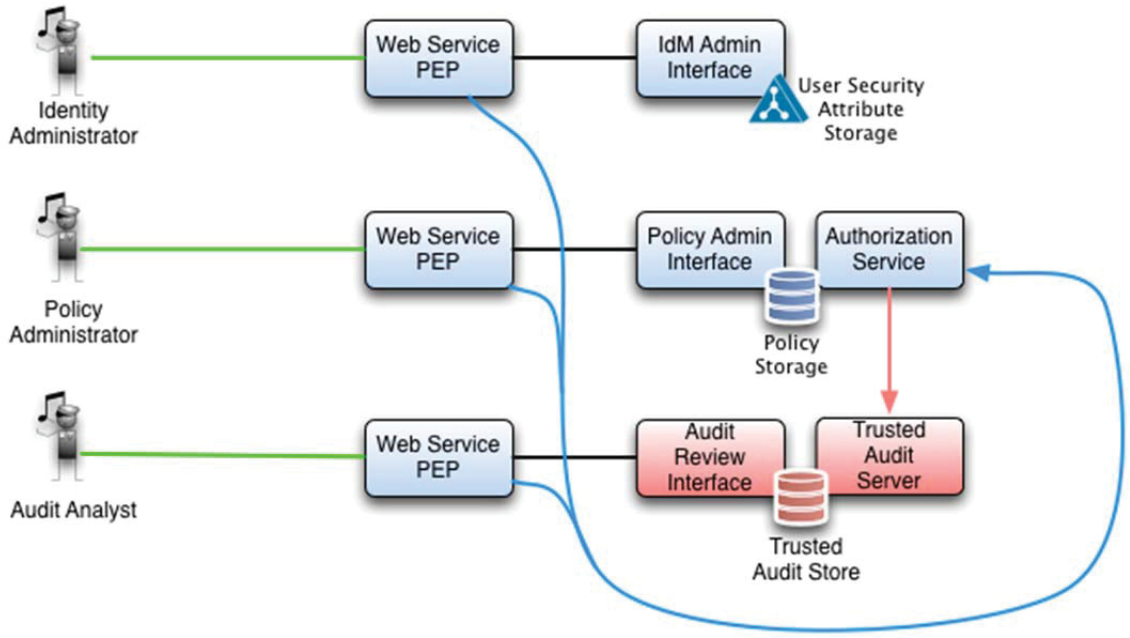

Figure 16: SAMSON Self-Protecting SAMSON interfaces

It is interesting to note that the PEP protecting the policy administrative interface will perform a policy check against the very security policy it is protecting. Similarly, since each policy enforcement action will result in the creation of an audit record in the Trusted Audit Store, an

Audit Analyst will see an audit record generated for the very policy decision that allowed the analyst access to the audit record review interface.

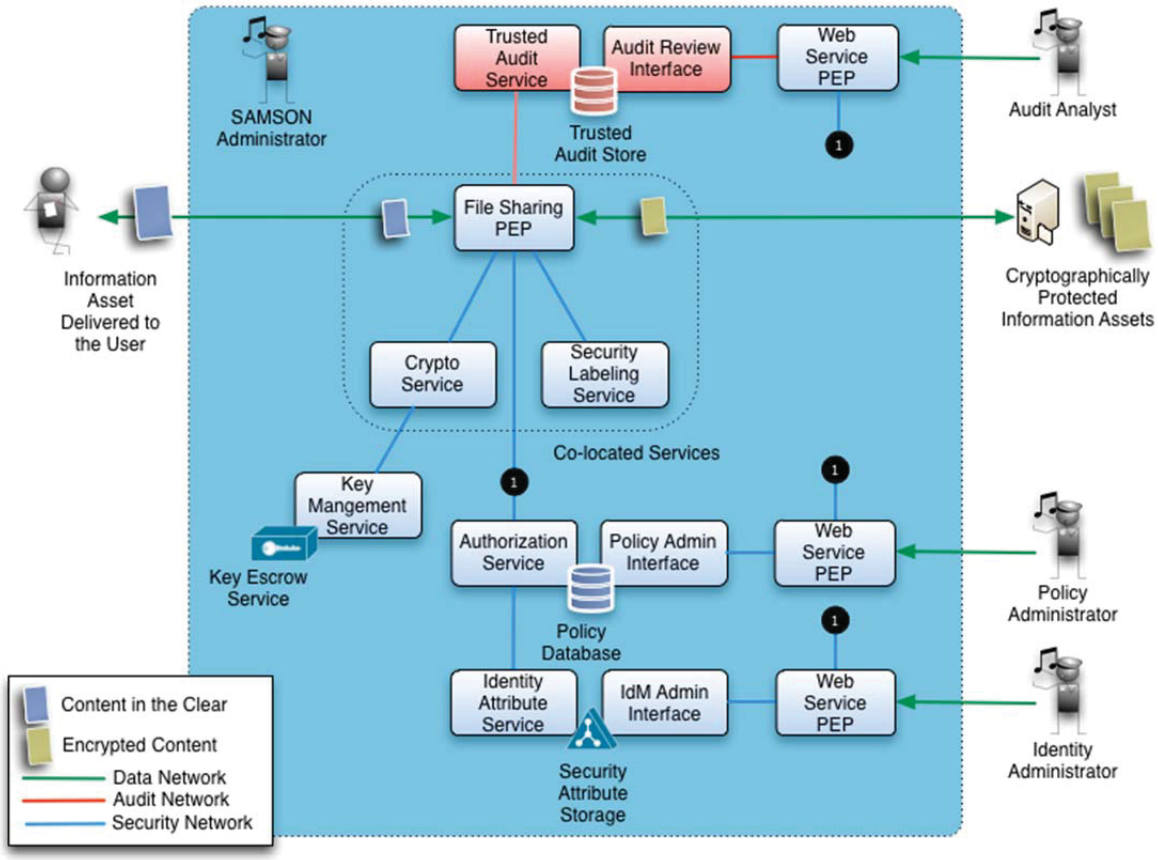

Figure 17: Self-Protecting SAMSON interfaces

When viewed in this context, the power of the SAMSON architecture as a flexible, yet self protecting architecture is clear. It is possible to envision expanded capabilities of the self-protected interfaces that grant all users a varying degree of control over SAMSON behaviour and, therefore, the ability to share information. If administrative interface data requests were subject to policy enforcement, rather than just access to the interface itself, then greater freedom of action could be granted to individual users. For example, SAMSON could allow users to create their own communities and policies with which they could control the release of information assets. This would allow SAMSON to support Discretionary Access Controls in addition of the mandatory controls expressed in the domain security policy.