Section 4: Stucturing and Placement of Security Services

4.1 Introduction

This chapter applies the security concepts introduced in {xxx} to the development of a generic approach to the definition of a security architecture.

Firstly, the perception of an IT system as a set of security domains is developed, each deploying a set of countermeasures specific to the identified threats to its activities and elements. Such countermeasures are generally a composition of discrete security services. The countermeasures applicable to the IT system domain as a whole is the composition of the countermeasures of all its constituent subdomains.

Secondly, the structuring of security services within an IT system is described, both in terms of the layering of services and the interdependency of security services within different subdomains of an IT system domain.

Finally, security services have both an operational and a management aspect and APIs are required to be defined for both aspects. In addition security is a cross-category service. Within a security architecture it is not sufficient to describe only the security services. It is also necessary to describe their interaction with the other categories of services within an IT system, for example system services and communication services). With these objectives this security framework includes a security service model which addresses the interaction with other categories of service.

4.2 Generic Security Architecture

An information system security architecture is a top level specification that is included within the overall architectural specifications for a system. It covers the high level identification, specification and deployment of a set of interdependent and co-operating security relevant elements (IT services and components of IT services). These elements are collectively responsible for enforcing the system security policy in the system domain.

The security functional elements are variously and collectively responsible for providing: identification and authentication, authorisation, security audit and security administration services.

One of the principal objectives of this framework is to establish a generic system security architecture that supports a range of instantiations. These security architectures will vary in:

System Security Policies
The types of system security policies reflect the different types of security functional and assurance requirements demanded by different market sectors.

System Type and Scale
System architectures range from standalone systems to large scale distributed systems required to operate within enterprises and, increasingly, across enterprises. The distributed systems are likely to be heterogeneous, using different vendor platforms.

4.3 Security Domains within an IT System

To help bring the various architectural aspects of information system security down to manageable proportions this framework employs the concept of security domains, introduced in xxx. The concept of nested and interacting security domains and their accompanying security policies may be used to:

• Capture many of the perspectives in terms of requirements and responsibilities.

• Handle the scaling of distributed systems, particularly those that cross enterprise boundaries.

• Define boundaries for security responsibility and jurisdiction within systems as well as between them.

An example of subdomains within an IT system is illustrated in Figure xxx. The system is subdivided into 3 subdomains; user session, database and print spooler services. Each subdomain has a security policy applicable to the particular functions it performs. Each subdomain includes security functionality to support its own operations. The security authority for each subdomain installs administrative security information to support the enforcement of policy within that domain, including the identification of the principals authorised to use the domain's services.

A user interacts with the user session security domain by creating an association with it. A user is permitted to use the services of that domain once the operational security information the user supplies has been validated by the user session domain (for example his identity is authenticated by means of his password) against the administrative information configured by the security authority.

A user may also require to use the services supported by the database security domain, or the print spooler security domain. Each of these security domains

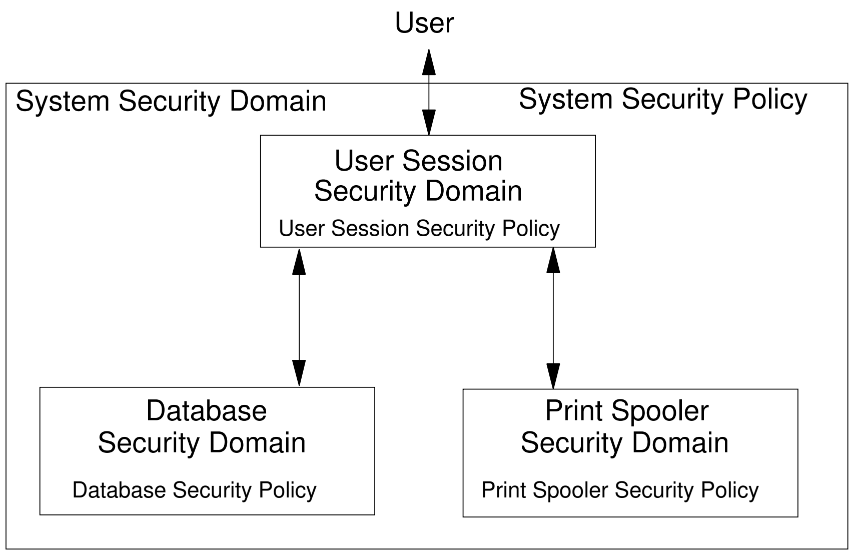

Figure 4-1 – Subdomains within a System Domain

require the creation of an association with the user session security domain to access their services. This is achieved by the exchange of further operational security information. In the absence of any provision (distributed security services) for the reuse or secure propagation of operational security information supplied to the user session security domain within the database or print spooler security domains further user authentication is required to access their services.

4.3.1 Distributed Security Service Domain within an IT System

In Figure xxx a distributed security service domain has been included within the system domain. The distributed security service domain is a subdomain of the system domain trusted by the other subdomains to provide operational security information. On user authentication, operational security information is issued by the distributed security service to the user session domain, for example a combination of authenticated identity and authorisation information. This operational security information may be used within the user session domain and also exchanged with the database and print spooler domains to establish associations to use their services. The database and print spooler domains now verify the origin of the operational security information instead of re-authenticating the principal it represents.

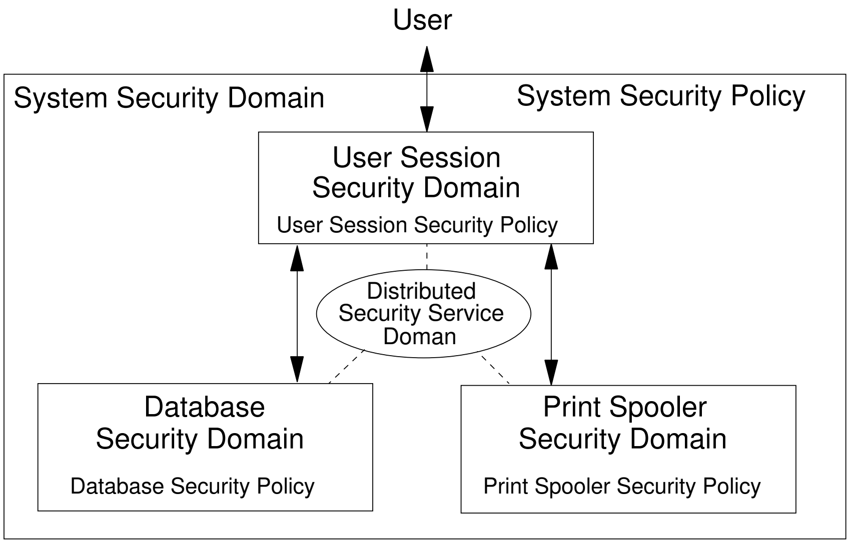

Figure 4-2 – Distributed Security Service Domain within an IT System

4.3.2 Platform and Service Domains

At the highest level, the framework separates the major architectural functional areas into the following types of security domain:

• *Platform domains* provide the services that control and structure the physical, processing and communication resources of an IT system into elements that may be used by the service domains.

• *Service domains* provide information processing services to principals in the context of enterprise operations (applications). The service domains may also comprise a set of *middleware* services that together with the platform services constitute the system infrastructure services.

The infrastructure services collectively provide the system environment that facilitates: communication, distributed processing, interactions with users, and system management and control.

Platform domains structure the physical resources of an IT system, processor memory, data storage and communication media into identifiable elements with assigned security characteristics and which mediate the flow of data between these elements. To enforce data flow mediation successfully the services of a platform domain are presented at a protected, non-bypassable interface.

A platform domain is typically composed of operating system and communication system elements. The actual interfaces and services presented by a platform domain are dependent upon the implementation. They may be represented by an interface equivalent to the POSIX.1 system service interface, such as a traditional operating system kernel implementation, or by a microkernel interface with POSIX.1 system services implemented as service domains.

Service domains are formed from the elements of platform domains, for example processes, files, and communication channels. Thus inter and intra service domain operations involving interactions and data flow between platform domain entities are subject to the security policy enforced by the platform domains.

The inter-relationship between service domain components and a platform domain is that of subdomain to superdomain. The platform domain enforces a basic policy on the activities of the service domain but may surrender some aspects of policy enforcement to a particular service domain component assigned that responsibility, for example a process that possesses a particular authorisation (capability).

A service domain relies upon the platform domain to enforce policies that provide for its protection, and may rely upon a platform to control its interactions with other service domains.

It is common for a service domain to provide services in support of other service domains, in particular some service domains may provide specific security services such as authentication or security attribute services. The system security architecture needs to identify such dependencies between service domains.

The point of separation between platform and service domains is based on the current realities of most open system implementations. Typically, a hardware supported, separation and protection boundary is to be found at the interface to the operating system kernel. The framework does not prescribe which parts of the infrastructure services lie below that protected interface. Nor does the framework prohibit the use of hardware-supported multi-ring protection domains. In this case the higher layers of the infrastructure and application service domains can be separated and structured using a stronger protection mechanism.

Where the operating system services of several processors are closely coupled and they are all enforcing the same security policy, then they can be considered to constitute a single platform domain.

4.4 Distributed Systems and Security Domains

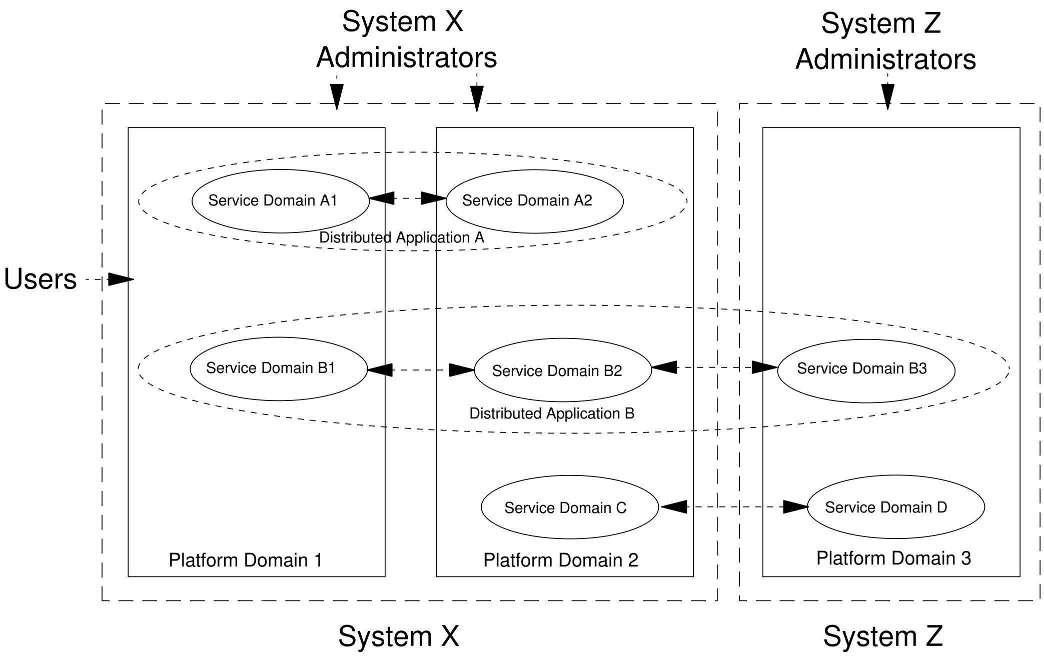

Figure 4-3 – Security Domains in Distributed Systems

Figure xxx provides an example of the structuring of security domains within a simple distributed system. Here the relationship between service and platform security domains, that of subdomains to superdomain, is illustrated.

A distributed application may span more than a single platform with elements of the application represented by service domains on more than one platform. Each service domain enforces a security policy common to the distributed application. However, the functions of each service domain are subject to the security policy enforced by the platform on the platform domain elements that comprise it. The interactions between the service domain elements of the distributed application are subject to any interconnection policy enforced between the platforms.

A distributed system may therefore be constructed so that some security services are implemented either within the platform domains, or within the service domains, or within both.

4.5 Trust and Assurance

4.5.1 Trust

Trust is a fundamental security architecture objective. Trust is defined as:

Element x trusts element y for some classes of activity in the context of a security policy if and only if element x has confidence that element y behaves in a well-defined way that does not violate the security policy, see

The administrative actions that establish a secure interaction policy between two security domains form a trust relationship between them. Each trusts the other to behave in a manner that does not violate the secure interaction policy. An example of this form of relationship is a secure interaction policy permitting the exchange of information whose privacy is to be protected. Regardless of any security measures implemented within or over the communication services transferring the information the originator must trust the recipient to uphold the privacy security policy once the information is delivered into the recipient's custody.

Similarly a trust relationship exists between a subdomain and a superdomain and is established by the administrative actions integrating the components of an IT system into a whole. For example this could be the assignment of security attributes, including capabilities, to service domain components, that is, program and data files.

Particular trust dependencies arise between a service domain and the platform domains that substantiate it in order to maintain the integrity of the security services of the service domain. As derived from ECMA-138 these are:

• Protect the security services and objects against external corruption. That is provide protection for the elements substantiating the service domain against unauthorised access by elements not constituting part of the service domain. An example is the privacy of process address space.

• Provide the communication security services (for example peer entity authentication) requested when forming associations or indicate that it is unable to do so.

• Provide the necessary initial security information to security services and objects. Example are cryptographic master keys and identities for service domains.

• Enforce use of an appropriate authorisation service for each access request. That is, the mediation of access to platform entities cannot be bypassed.

• Direct all initial human user contact to a subject sponsor and secure this link according to the local security policy. This is may include the provision of a trusted path. Within this framework this requirement extends to require any initial contact by a principal with a security domain to be directed to a component acting as sponsor for the security domain.

4.5.2 Assurance

The security-relevant and enforcing elements are sometimes known as the *trusted* elements. Here *trust* implies that the required degree of confidence (assurance) in the correct and continued working of these elements is established and maintained with respect to the security policy in effect.

Confidence in the overall system security solution is established by designing, implementing and proving a system security architecture against these general objectives and criteria:

- The trusted elements are closely integrated and can thus be relied on collectively to enforce the system security policy.
- The trusted elements are adequately separated and protected from the *untrusted* elements, so that the opportunities for the latter to interfere with the correct working of the former are minimal.
- The trusted elements are deployed such that they can appropriately mediate the security relevant operations of the untrusted elements, and that this mediation is not easily bypassed.

Confidence in the continued correct working of the trusted elements is addressed by the system security architecture design and through the application of well-defined security operating procedures and strong operational configuration management and administration to the system.

Any delegation of responsibility for the enforcement of security to the information system must be based on some measure of confidence in the correct and continuing working of the security relevant measures and functionality.

The procedures for the rating, measurement and achievement confidence (assurance) are referred to as evaluation. The schemes for specifying and evaluating system security assurance continue to evolve. However, the basic the basic engineering principles and objectives necessary to support assurance requirements need to be addressed in the development of system security solutions and architectures.

In summary, assurance establishes confidence in the correctness and effectiveness of security solutions and their components at each stage in the system life cycle:

- architectural design
- component and product specification
- development
- supply
- deployment and integration
- maintenance and management
- disposal.

4.6 Layering of Services

A service is a combination of functional and data elements that are exercised and accessed through a well defined interface. The range of operations performed by the service functional elements on the service data elements or the passed data elements is specified by the interface.

The layering of services is a common characteristic of information system functional architectures. For services located at all but the lowest levels within the system, both service functional and data elements are often formed out of the more primitive elements of underlying services through the use of the interfaces to those underlying services. The term substantiate is used to cover this service formation through functional and data element abstraction.

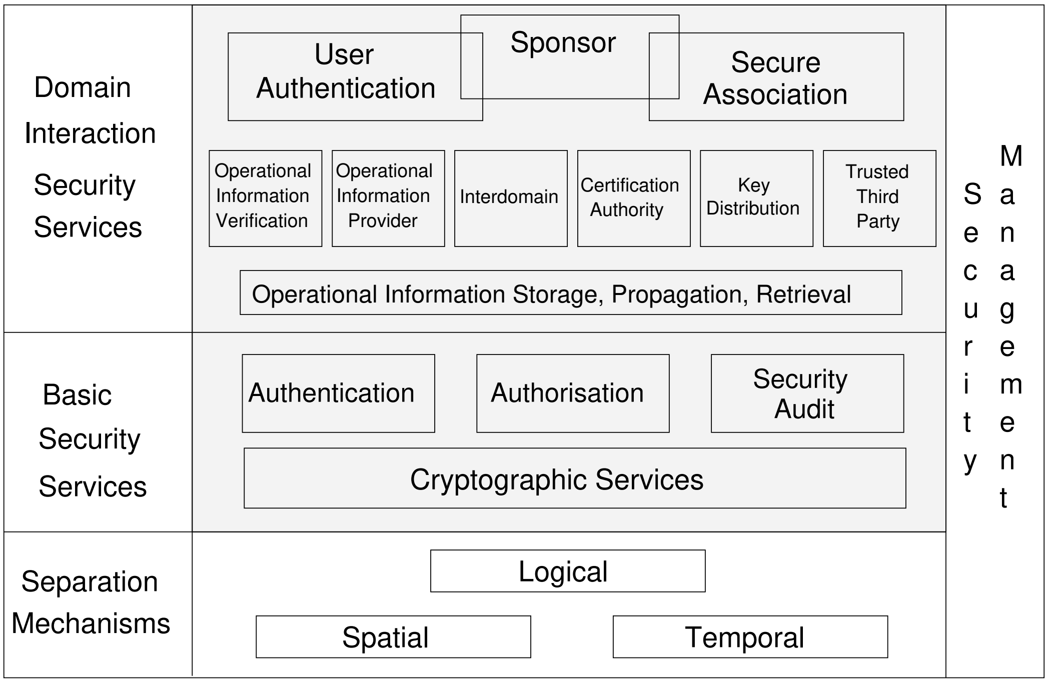

Figure 4-4 – Generic Security Services

The security services within a security domain may be considered to be based on a layered model as illustrated in Figure xxx. This comprises:

• at the lowest level, security mechanisms responsible for the segregation of the system resources that are used to substantiate the IT security domain elements and activities,

• at an intermediate level are the basic security services used to enforce primitive elements of security policy,

• at the highest level are domain interaction security services that control the security information environment in which the basic security services operate. These services are responsible for the distribution and exchange of security information necessary to support interactions between security domains (including users in the physical domain interacting with the IT system).

• applicable to all levels are requirements for security management.

4.6.1 Separation Mechanisms

In order to protect the integrity and confidentiality of the activities and elements of the domain, a security domain is required to provide for the segregation of the domain's elements and activities from any other domain's activities and elements.

In addition, the maintenance of integrity of those activities and elements of a security domain that comprise the security services of a security domain, termed the Trusted Computing Base of the domain, depends upon the segregation of those activities and elements from the other activities and elements of the domain.

The elements and activities of security domains within an IT system are substantiated by the physical and processing resources of the IT system. The segregation of domain elements and activities therefore depends upon techniques for the assignment of IT system resources to support domain elements and activities and the protection of the segregation of the resources assigned.

The provision of such segregation may be based upon three protection and separation techniques.

Physical

As a segregation mechanism, this is the simplest and crudest in that it is based on the physical separation of information processing resources. A trivial example is a standalone single user system. This is a simple and effective technique, but has restricted applicability to the segregation of component processing environments within the context of distributed system objectives.

However, physical protection of processing and communication components is an important factor in determining the nature, strength and placement of security measures within a distributed system.

Temporal

Temporal separation techniques operate at different levels. At the coarsest level of granularity, an information processing system is only used to process information of one sensitivity at any one time and is totally purged before being used again. This technique when applied at a finer level of granularity to targets, such as transient files, buffers and memory address space, is used to counter disclosure through garbage collection and similar attacks. As such it is known as the *object reuse* measure. Another dimension to temporal separation is object locking as used by logical integrity mechanisms. (Single threading and atomic transaction techniques are important to the preservation of the integrity of security-sensitive information.)

Logical

Protection and separation techniques of this type are based on the combination of hardware and software functionality to create separation environments and properties at different layers within an information system. At the lower levels, the operating system, usually through the use of specific hardware features, separates and protects itself from the vagaries of higher level functionality. At varying levels of strength it provides separation of the different services and resources it offers, for example devices, directories and files. It also separates and protects the higher functional components from each other by establishing a multiprocessing environment, controlling the mapping of memory to each process, and mediating the different ways processes can communicate with each other. These operating system protection and separation features are fundamental to the implementation of any reasonably robust system, let alone a *secure* one. Based on their sound implementation at the operating system level, logical separation and protection techniques are used, where required within higher level functional components.

Deficiencies, or inefficiencies, in the support provided by the above segregation techniques, for example in the use of communication media, may be countered by the use of cryptographic services deployed within higher layers of service, (see However, it should be noted that the security provided by cryptographic services depends upon the above segregation techniques being deployed at critical places within a distributed system to protect the cryptographic algorithms and keys.

4.6.2 Basic Security Services

The basic security services are security service primitives that provide a discrete security service. These include:

— cryptographic and key management — providing services, amongst others, for the encypherment and decipherment of information

— authentication — providing for the validation of claimed identities

— authorisation — providing for the authorisation of the use of activities and access to elements

— audit — providing for the detection, recording and analysis of the occurrence of security relevant events.

The basic security services are described in detail in

4.6.3 Domain Interaction Security Services

The operation of the basic security services depends upon the association of security information with the activities and elements of a security domain and with the principals that use a domain's services.

In order to use the services of a domain, an individual needs to form an association with the domain. The domain interaction security services are responsible for the authentication of a principal's identity and establishment of a set of security information representing the security context within which the domain's services may be provided. This necessitates the propagation of that security context with the tree of execution, the execution path, taken in performing any operation.

Such operations may require the invocation of services within other security domains through the formation of associations between the security domains. This necessitates the exchange of security information in order to propagate the security context within which the association is created and services invoked.

The services to support the formation of such associations between individuals and the IT system, and between security domains within the IT system are described in detail in

4.6.4 Security Administration Services

Each security service has both operational and management aspects. The definitions of each security service in definitions of the management aspects of each service required for configuration and control of the service.

Additional administrative responsibilities may include:

- Controlling the starting and stopping of security services.
- Detecting and handling security service failure and ensuring recovery to a secure state.
- Facilitating and managing the creation, maintenance and consistency of distribution of security management information for each security service. This may include the distribution of cryptovariables (keys) used by the cryptographic services.
- Maintaining historic (archived) records of system security configuration. This information is usually required to support historic audit trail analysis, and may therefore, be included as part of the function of the accountability and audit services.
- Collection and management of the current security operational state information from the system. Here there is an overlap in responsibility between access control and management services. But as this security operational state information is not exclusive to access control services, it has been allocated as a security management responsibility.

4.6.5 Relationship to Platform and Service Domains

The model of the layering of security services described here does not support a single mapping to platform and service domains.

The logical separation mechanisms available are generally enforced by platform domain mechanisms, but service domains are also responsible for applying separation measures to entities within their own domain, for example, object reuse.

The basic security services are deployed within both platform and service domains and some domain interaction services may be supported by service domains, platform domains, or both.

4.7 Security Service Integration

4.7.1 Introduction

Systems are composed of many different categories of service that in combination provide facilities to perform information processing. These categories may generally be considered independently of each other, for example system services and communication services. However, two categories of service, security and system administration, are cross-category services. That is, they are inherent within all the other service categories.

The security requirements to be met by any particular category of service within a given system are derived from the system's security policy and are mapped to the system's services by the system's security architecture.

With respect to these security requirements, services can be classed as:

Security Enforcing
The service is required to meet some specified security requirements in the way in which it operates, that is, enforce some aspect or aspects of the system security policy. For example, a database application may be required to ensure that the system access control policy is applied with respect to the access to database data elements (tables, records and fields). Operating system services are usually required to enforce those parts of the system security policy appropriate to operating system services.

Security Enforced
Here the service is not required to enforce some aspect of the system security policy. Instead, that aspect of the policy is enforced on the operations of the service by the information processing environment in which it runs, that is, the relevant security enforcing services that provide that environment, typically the operating system services within the platform domain.

Within the layered functional architecture, the higher-level security enforcing services or elements of those services, are often dependent on particular security functions of the underlying services to help them meet their security requirements. For example, a message passing application service, required to protect the confidentiality of messages, may expect to use message confidentiality services provided by the communication services, if available, and be able to test for their existence.

The behaviour and properties of security enforcement performed by a service are defined by parts of its interface.

The callers of these services can be categorised as:

Security Aware

Here the caller is aware of some or all of the security functionality provided by a security service. Security aware callers typically are engineered to make use of the security services available to them. They may also be required, in certain circumstances, to take responsibility for some part of the security policy normally enforced by the security services. For example, some elements in a service domain may be assigned the capability, as specified by the operating system interface, to *override* the normal authorisation policy enforced by the operating system. That is, the operating system service surrenders responsibility for policy enforcement to a caller assigned the capability to assume such responsibility.

Security Unaware

Here the caller is unaware of some or all of the security functionality provided by a security service. In these cases the caller is unaware of either the security functionality available or the way that security-functionality is controlling the security of the environment in which it is operating.

4.7.2 Portability and Interoperability

The dimensions of security services and their interfaces, outlined above, have to be addressed in the specification of APIs. In particular, they have to be taken into account in the way the APIs are specified to support the engineering of portable and interoperable system security solutions.

For any one system, which services are to be security enforcing and which services are to be unaware of aspects of security enforcement, is determined by the system security architecture and security policy. In turn the definition of the system security architecture is dependent upon the security service capabilities available.

This framework endeavours to identify and place generic security service APIs. However, the security services will generally be invoked indirectly by other security enforcing services. From the perspective of callers of these security enforcing service APIs the interaction with the security services may be explicit or implicit. That is, the caller may be unaware of any aspect of security as it affects the behaviour of a security enforcing service, or the caller may be aware of some or all aspects of security enforced by the security enforcing service and be required to manipulate the security services. It is therefore insufficient just to define security service APIs for portability and security information exchange formats to support interoperability. APIs to services which are required to be security enforcing need to be structured and extended to acknowledge the requirements to support the underlying security services and meet the different types of usage.

4.7.3 Security Service Model

In order to support the description of the characteristics of the different security services, their placement within a system, and the potential impact on other service APIs a security service model is defined. This model of interaction with security services and their APIs is applicable to services at all levels within a system.

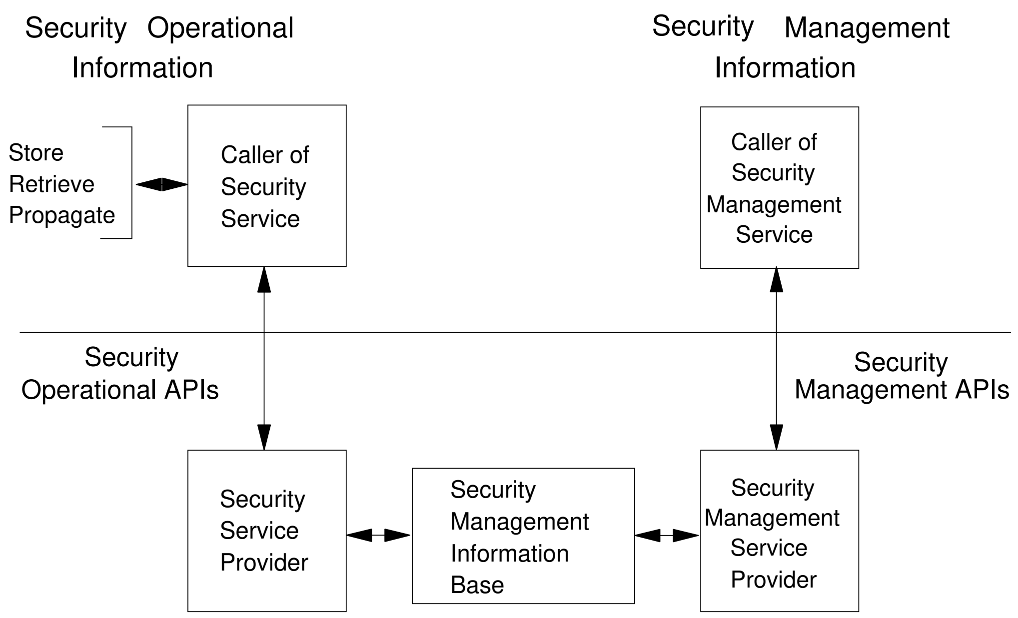

Figure 4-5 – Security Service Model

A security service is based on the provision and use of security information. The security information is of two types: firstly security management information established by administrative action, secondly security operational information established by operational actions. Each security service provides two sets of APIs, a set of security management APIs which handle the security management information and a set of security operational APIs by which the actual security service is provided. This model is illustrated in Figure xxx.

Security Management APIs

The security management information is manipulated by a set of security management services and accessed by security management APIs. The security management information can be considered to be held in a Security Management Information Base (SMIB) which is accessed by the security service provider as required. The SMIB requires protection from access by any functions other than the security service provider and the security management functions. That is, the integrity and confidentiality of the SMIB must be maintained.

The SMIB is a set of security information associated with the elements of the security domain. The Management Services relate to the installation, modification, listing of the security information and the enabling and disabling of a particular security service. That is, the security management APIs are used to control and configure a particular security service.

Each security service has a specific set of security management APIs and a specific set of security information associated with each element within a security domain. That is, each element may have several sets of security information associated with it, potentially one set for each of the individual security services.

Security Operational APIs

The security operational information is handled by the entities that call the security services. The security operational information is passed between the invoking functions and the security service provider across the operational interfaces to the security services. The Operational Related Services are the means of delivery of the actual security service.

The security operational information is associated with a sponsor and principal pairing and represents the security context established to service the principal. The security context has to be generated initially as a result of principal authentication. It then has to be stored and propagated with the chain of execution (the sponsor of the principal) within the system. That is, it has to be associated with every process or thread created and executed as part of the session servicing the principal and it must be propagated to any independent service that provides services to that principal. The integrity and confidentiality of the operational security information must also be maintained, including its binding with the sponsor/principal pairing.

The current security state of a security domain at a point in time is represented by the combination of the SMIB and the current set of security operational information maintained by that security domain. The security state of a system is represented by the combination of the security states of every security domain within the system.

Security and Primary APIs

For any service that is required to enforce some security functionality, it is necessary to specify a service interface that accommodates security unaware and security aware callers. The term *primary interface* is used for the service interface visible to both types of callers. The term *security operational interface* applies to a secondary interface which may be accessible to security aware callers.

Security Unaware Caller

In this example the callers of the primary service are unaware of security and the API to the primary service makes no provision for the control of the security context of the primary service. The implementation of the primary service itself has to include functions to invoke the appropriate security services, authorisation for example, and to acquire and handle any security operational information required by the security service.

The primary service API operates within whatever default security context it is invoked. This default security context has to have been previously established by security aware functions, for example a login program. In this case the primary service API includes no provision for security information within its input and output parameters. However, its behaviour such as success or failure has to reflect the implicit security policy and its return values or error status codes may include security related failures, for example authorisation failure. It may also be required to generate audit records.

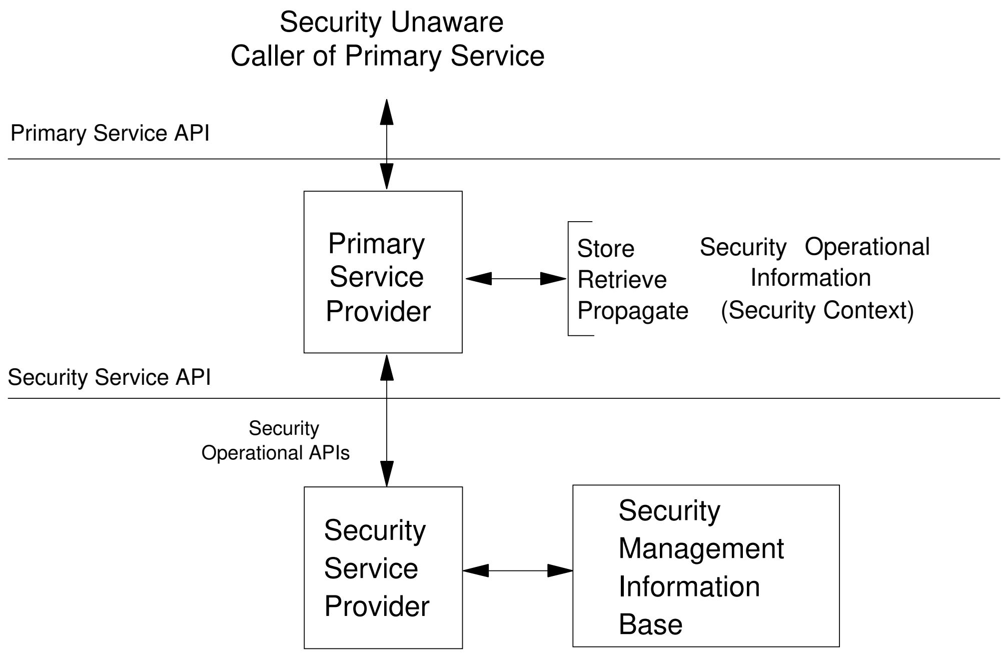

Figure 4-6 – Security Unaware Caller

Security Aware Caller

In this case the caller of the primary service is aware of one or more security services and is be responsible for handling security information related to those security services.

It may interface to these security services in two ways:

- *Directly* — by invoking the security service API to configure default values applicable to the primary service API. In this case the primary service API may be identical to the security unaware case, or
- *Indirectly* — by passing the security information as parameters to the primary service APIs. In this case the primary service API specification must include the security information, which may be optional parameters.

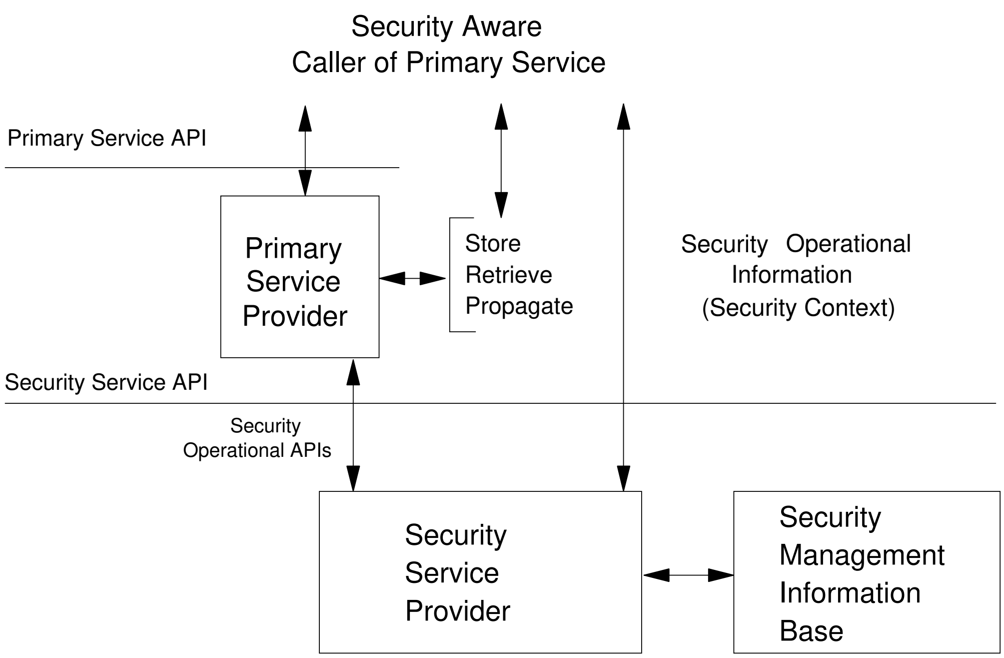

Figure 4-7 – Security Aware Caller