Open Security Exchange™

Physical Security Bridge to IT Security
PHYSBITS

Framework and Data Model

The Open Security Exchange is developing a series of specifications for promoting interoperability in security management.

Open Security Exchange – Physical Security Bridge to IT Security (PHYSBITS) is a vendor-neutral approach for enabling collaboration between physical and IT security to support overall enterprise risk management needs.

This document describes: the Open Security Exchange – PHYSBITS framework for the integration of physical security and IT security; a supporting data model; use cases; and best practices for access card selection and log management.

Following documents in the Open Security Exchange – PHYSBITS series will build from this model to define Web services and other interfaces, XML data formats and additional best practices.

Version 1.0
April 14, 2003

INTRODUCTION 3

OPEN SECURITY EXCHANGE – PHYSBITS 3
DOCUMENT SCOPE 3
SPECIFICATION TEAM 4

FRAMEWORK FOR BRIDGING PHYSICAL AND IT SECURITY 4

PHYSICAL AND IT SECURITY TODAY 4
INTEGRATION OF PHYSICAL AND IT SECURITY MANAGEMENT 6
INTEGRATED SECURITY MONITORING 7
INTEGRATING USER ADMINISTRATION 7
OPEN SECURITY EXCHANGE – PHYSBITS FRAMEWORK 8

THE OPEN SECURITY EXCHANGE – PHYSBITS DATA MODEL 9

PURPOSE AND SCOPE 9
DATA FORMATS AND NORMALIZATION 10
DATA MODEL SCOPE 10
DATA MODEL OBJECTS 10
PHYSICAL OBJECTS 12
SUPPORTING OBJECTS 15
APPLICATION OBJECTS 16
EVENTS 19
OTHER VIEWS 22
ASSET-/READER-CENTRIC VIEW 23
APPLICATION-CENTRIC VIEW 24

DATA MODEL USAGE 25

RETRIEVING ENTITY DATA 25

USE CASES FOR OPEN SECURITY EXCHANGE – PHYSBITS 27

CARD LIFE CYCLE MANAGEMENT 29

INTRODUCTION 29
CARD MANUFACTURE AND ISSUANCE 29
PERSONALIZATION 30
IDENTIFIERS FOR HYBRID CARDS 30

ACCESS CARD SPECIFICATIONS 33

INTRODUCTION 33
STANDARDS AND REFERENCES 33
RELATED SPECIFICATIONS 34
CONTACTLESS SMART CARD READER SPECIFICATIONS 34
CONTACTLESS SMART CARD CREDENTIAL SPECIFICATIONS 35
ACCESS CARD 35
ATTRIBUTES OF AN OPEN SECURITY EXCHANGE – PHYSBITS SMART CARD 36

AUDIT TRAIL, INCLUDING MANAGEMENT BEST PRACTICES 37

INTRODUCTION 37
BEST PRACTICES 37
DATA FORMATS FOR NORMALIZED EVENTS 38

Introduction

Open Security Exchange – PHYSBITS

The Open Security Exchange is developing a series of specifications for promoting interoperability in security management. Open Security Exchange – Physical Security Bridge to IT Security (PHYSBITS) is a vendor-neutral approach for enabling collaboration between physical and IT security to support overall enterprise risk management needs. This document describes the Open Security Exchange – PHYSBITS framework for the integration of physical and IT security, along with a supporting data model, which together enables data to be shared between physical and logical access control systems.

Open Security Exchange – PHYSBITS provides a data model for the integration of physical and IT security. Converging these security environments addresses security gaps that fall between these two different security disciplines and helps protect organizations against multifaceted security threats. Because these two industries manage vastly different scopes of security, bridging the two together and addressing their various aspects of security will require sustained industry effort.

While security can be gradually improved and advanced, a complete picture of organizational security is very difficult to assess without a comprehensive data model as a working guideline and basis for a standard. This model is intended to be the foundation for a series of specifications that should focus on the individual disciplines of security and define the interoperability standards that form a more complete security management system and process.

It is intended that the structure of these standards be both flexible and extensible. In addition, its contents should be mapped to applicable industry standards and be specified to enable Web services and other implementations of the interfaces, as well as XML data formats.

Authentication systems and management tools that are Open Security Exchange – PHYSBITS-compliant will be able to integrate into an overall enterprise architecture for provisioning, monitoring, auditing and managing physical and IT security systems.

Document Scope

This document contains:

- Introduction and framework
- Data model and use cases
- Card life cycle management, including manufacture and personalization
- Mandatory and best practices attributes for card selection and provisioning
- Audit trail, including management best practices

Other related documents, which are being developed by the specification team, will include:

- Web services and other interfaces, and XML data formats
- A set of best practices guidelines for system integration and deployment

Specification Team

Company	**Reviewers and Contributors**
BAE Systems	H. Amos
Computer Associates International, Inc. (CA)	D. Batchelder P. McMahon
Gemplus	R. Elmore N. Hislop R. Sharma
HID Corporation	D. Spitler E. Widlitz
Pinkerton Consulting and Investigations Inc.	W. Belmont
SAFLINK Corporation	C. Tilton
Software House (a member of Tyco Fire & Security)	D. Hawkins

Framework for Bridging Physical and IT Security

Physical and IT Security Today

Today, physical security focuses on the protection of assets, personnel and structures against potential assessed risks. In addition, managing the flow of individuals and assets into, out of or within a facility are extremely important aspects of physical security. Managing areas, perimeter intrusion, occupancy, access methods (for example, standard entry and egress vs. the allowances made for a wheelchair user), internal and external facility monitoring, and containment are all issues that must be dealt with by physical security experts today.

IT and network security focuses on authorization for users to access IT services to which they are entitled and helps ensure business continuity. These resources can include: roles on a network; permissions to a database; drive space allocations; email access; Internet/extranet and intranet access; and remote access privileges. Today, IT managers are focused on the enormous task of managing resources that can vary from person to person, and can encompass literally hundreds of thousands of items that all comprise a network user's tools, which allow them to conduct business on a daily basis.

An effective security policy addresses both physical and IT security. In almost every large enterprise, physical and IT security are present but are not often coordinated, or organizationally or operationally functional. In many cases, these two disciplines are completely independent and unaware of the strengths and weaknesses of each other's practice.

This has a number of consequences:

- Incompatibilities between building access hardware tokens and IT access tokens
- Trend analysis and specific forensic investigation struggle to relate physical access logs to IT logs

- Lack of consistent standards for journal and IT log management, indicating that logs may not be of evidentiary quality
- Monitoring systems do not provide a situational awareness of coordinated physical and IT attacks
- Costly, manual processes for new hires and contractors to get building access set up and changed when their access needs to be changed
- Lack of integration of building access and business processes for new hires, and de-provisioning terminated staff — potentially causing security exposures

This document identifies how an integrated approach to business security management can provide an integrated view of physical and IT security management.

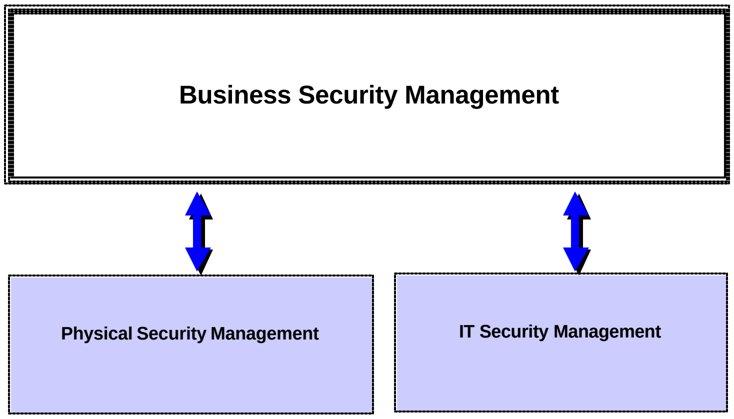

Figure 1 – Integrated Security Management

Integrated business security management can have a number of benefits:

- Reduced administrative overhead through automation of manual processes for provisioning and de-provisioning users
- Enhanced security through a combined view of potential intrusions within the physical and IT environments
- More effective reporting and investigations through a centralized, normalized audit trail
- Support for a forensics process through consistent audit logs of evidentiary quality
- Cost savings through leveraging investment in authentication tokens to work to secure access for both physical and IT systems

Security management must itself be integrated within existing business processes for managing people, facilities and IT systems.

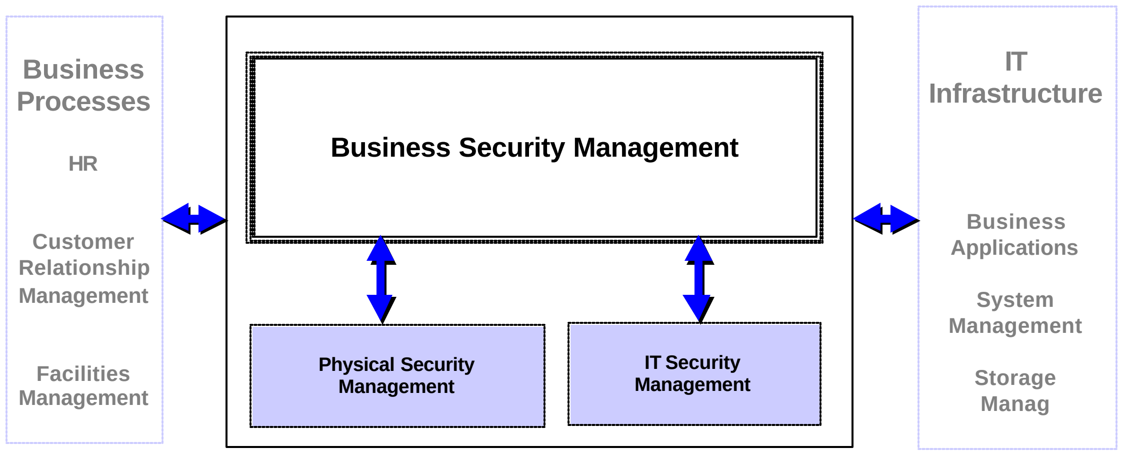

Figure 2 – Business Integration of Security

As shown in Figure 2, this integration can be exploited to enable more effective integration of security within business processes (such as human resources [HR] and telecommunications provisioning). In addition, this integration can also be used to connect security within the IT infrastructure so that vulnerability remediation can be linked within existing system change management processes.

Integration of Physical and IT Security Management

Effective security management requires clear organizational ownership and accountability across a number of critical security management processes, including:

- Enterprise security policy definition
- User provisioning and asset management
- Security monitoring and auditing
- Incident response
- Business continuity planning

Technology can support each of these processes but does not define them. Assuming that an organization has performed an asset inventory and conducted a risk assessment, it is then appropriate to look at where technology can reduce the threat of specific vulnerabilities. Historically, investment in security technology has focused on technologies that can solve particular problems, such as building access systems, firewalls, virus prevention and intrusion detection. Increasingly, large, complex organizations require an integrated approach to security management, which can deliver a consolidated approach for management of security policy and security events, as shown in Figure 3 below:

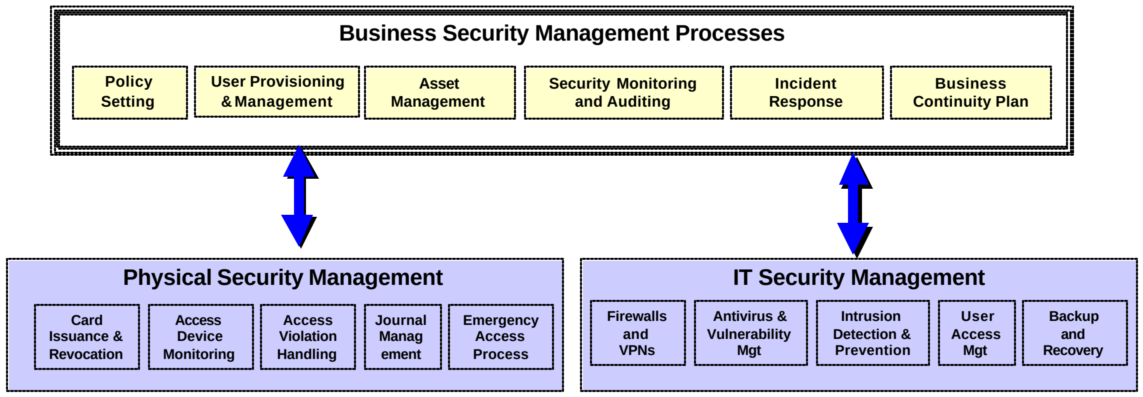
Business Security Management Processes
Policy Setting
User Provisioning & Management
Asset Management
Security Monitoring and Auditing
Incident Response
Business Continuity Plan
Physical Security Management
Card Issuance & Revocation
Access Device Monitoring
Access Violation Handling
Journal Manag ement
Emergency Access Process
IT Security Management
Firewalls and VPNs
Antivirus & Vulnerability Mgt
Intrusion Detection & Prevention
User Access Mgt
Backup and Recovery

Figure 3 – Integrated Control of Security Management Processes

A consolidated approach to enterprise security can enhance security posture and reduce security management costs.

Integrated Security Monitoring

Security monitoring is one of the components within a best practices enterprise security architecture. Experience has shown that this technology is ineffective without both comprehensive coverage and a business process for regularly monitoring its outputs and taking actions.

An integrated approach to event management with a clear supporting business process enables new complex attacks on facilities and IT systems to be detected, correlated and prevented — and enterprise-wide trends to be discovered through trend analysis across logs of both physical and IT systems. This requires data to be consolidated into a normalized format across multiple devices, and to be correlated so that critical events get immediate attention.

Integrating User Administration

User management is a business process. From a best practices perspective, user management needs to be driven by HR, line of business requirements for employee access and business needs for customer/partner management. However, in most enterprises today, user management is often fragmented across multiple systems and applications and physical access.

Business spending associated with user management can be readily identified, such as help desk calls to reset passwords, the amount of time spent on hold while locked out, time before new hires are fully provisioned and so on. One of the most straightforward ways to reduce security management costs is through automation and combination of provisioning of users for physical and IT access.

Automating user provisioning and managing user access requires a number of processes, which need to be established within an enterprise for defining and maintaining internal users' identities and access, as well as provisioning and maintaining customers and partners. These are shown in Figure 4 below:

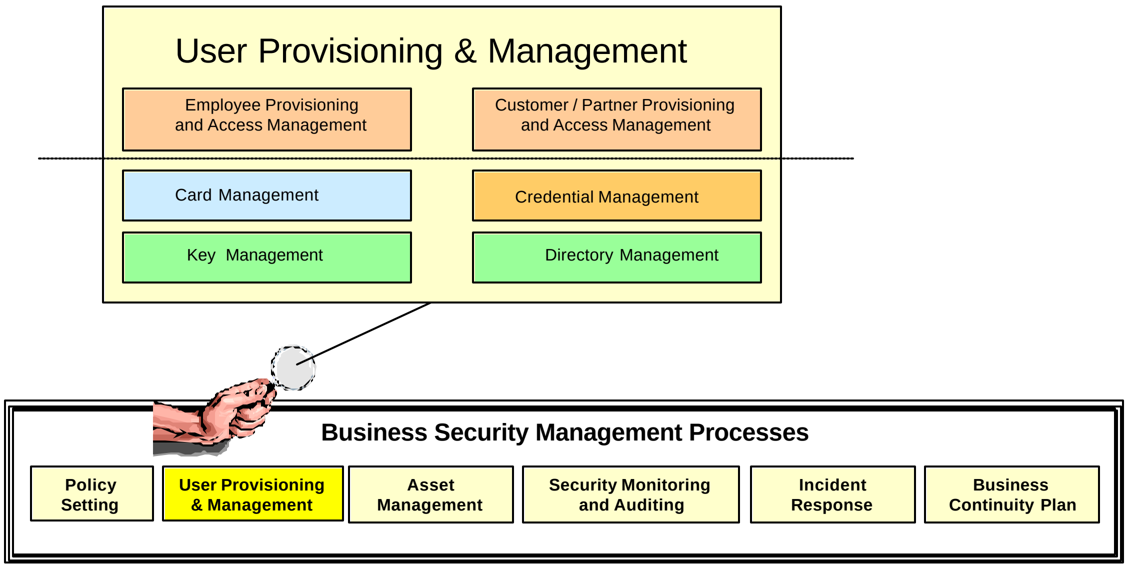
User Provisioning & Management
Employee Provisioning and Access Management
Customer / Partner Provisioning and Access Management
Card Management
Credential Management
Key Management
Directory Management
Business Security Management Processes
Policy Setting
User Provisioning & Management
Asset Management
Security Monitoring and Auditing
Incident Response
Business Continuity Plan

Figure 4 – User Provisioning and Management

User provisioning and management comprises:

- **Employee Provisioning and Access Management** — includes setting up new hires with: system and facility access; self-serve password management; access card management; and employee de-provisioning. Needs to integrate with HR processes and applications.
- **Customer/Partner Provisioning and Access Management** — enables new customers to get registered, and manage their access levels and preferences. Needs to integrate with Customer Relationship Management (CRM) processes and applications.
- **Card Management** — issuance and life cycle management of access cards
- **Credential Management** — system and application authentication information
- **Key Management** — authentication and encryption key generation, distribution and change. For PKI environments, key management includes certificate generation, distribution and revocation.
- **Directory Management** — infrastructure that enables distributed, scalable access to user information and attributes

Open Security Exchange – PHYSBITS Framework

Converging physical and IT security within a consistent framework is a significant challenge. Open Security Exchange – PHYSBITS provides a framework for approaching this challenge using a modular approach.

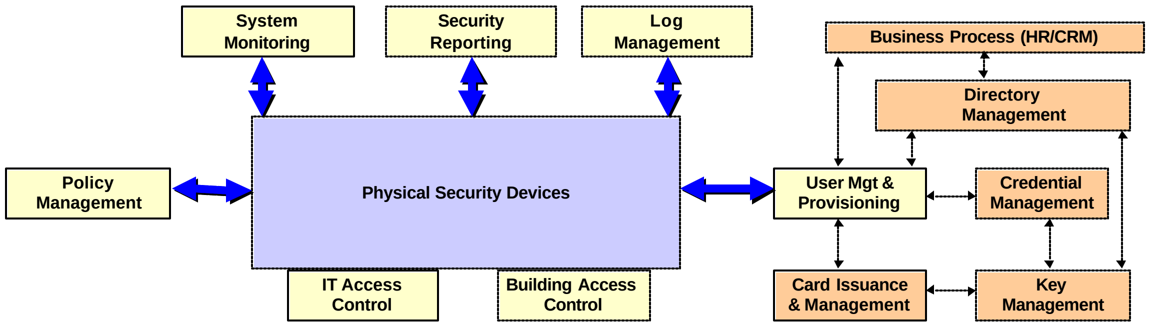

Figure 5 – Open Security Exchange – PHYSBITS Framework

A set of interfaces and data formats are required to enable the integration of physical and IT security management:

- **User Provisioning** — enables users to be set up with access devices with a consistent set of data associated with each user, which integrates with enterprise identity management driven by business processes and also integrates with infrastructure management associated with the access token (such as issuance and revocation)
- **Policy Management** — sets the controls used by the devices (when to raise alerts, what level of logging)
- **Security Reporting** — allows device settings to be reported on
- **Log Management** — enables centralized management of journals and normalization into a consistent format
- **System Monitoring** — allows IT security event management to monitor physical devices and correlate attributes, such as location, with network and other intrusion monitoring systems

The PHYSBITS series of specifications will include Web services as well as other APIs and XML data formats to support each of these functions, together with a series of best practices guidelines for conformance with industry codes, such as ISO 17799 — maintaining evidentiary quality of audit logs.

The Open Security Exchange – PHYSBITS Data Model

Purpose and Scope

This section of the document focuses on a data model specification that describes a method to map physical security applications data (such as events, information and methods) into an IT security framework.

The Open Security Exchange – PHYSBITS Data Model (which can also be referred to as an Information Framework Model) is the root of all information models and structures to be developed as part of the Open Security Exchange development process.

The Open Security Exchange – PHYSBITS standard development process consists of data models developed to describe the requirements and design for interoperable organizational security standards. It includes data diagrams and use case models, interaction models, data type models, terminology models and other types of models to provide as complete a view of the Open Security Exchange standards requirements and design.

Data Formats and Normalization

Using standards-based communication and data exchange formats will best facilitate interoperability between the physical and IT security environments. Standard protocols, such as XML, TCP/IP, ODBC and SQL, will be used as a base functionality for any solution developed with the Open Security Exchange – PHYSBITS approach. Helping to ensure adherence to these standards will foster greater functionality and easier implementation of the components of any Open Security Exchange – PHYSBITS-compliant solution.

As indicated, abstract implementations are the basis for a complete XML format, which will include definition of objects and attributes to enable interworking. A normative vocabulary shall be defined, indicating the mappable data between the disparate security disciplines. Extensibility to add vendor-specific attributes will be permissible, but adherence to the base interfaces and data formats will enable interoperability.

No prescription is made in this data model or subsequent specifications on how intra-system communications or internal storage will be performed — as these will vary from implementation to implementation.

Data Model Scope

By implementing an Open Security Exchange – PHYSBITS interface, physical security vendors will be able to integrate their applications with the following IT security processes and supporting technologies:

- **Event management systems** — auditing and monitoring, log management
- **User provisioning systems** — define, maintain and remove users
- **Vulnerability assessment systems** — verify policy complies with corporate standards
- **Security management products** — policy management and reporting

Data Model Objects

The Open Security Exchange – PHYSBITS Data Model is described as a set of objects. The objects break down into the following groups:

- **Physical Objects.** These objects relate to physical entities within a PHYSBITS application. They maintain state. At the top level, they include Person, Reader, Asset and OtherObject. Persons have Credentials, AssignedRoles and AssignedRights.

- **Supporting Objects.** These objects provide additional information to the Physical and Application Objects. They include Location, Picture, Video and AccessObject. AccessObjects have Roles and Rights. Roles have AssignedRights.

- **Application Objects.** These objects relate to the PHYSBITS application. They include User, OtherObject, PolicyCheck and Report.

- **Events.** These are the events that PHYSBITS applications can emit. They include StatusEvents, ApplicationEvents, PersonEvents and AssetEvents.

Each PHYSBITS object's attributes and methods are described below.

Following the object description, a series of diagrams show different "views" of the objects. These views help put the PHYSBITS objects in context.

Physical Objects

The following diagram shows the PHYSBITS physical objects (derived from PhysObject):

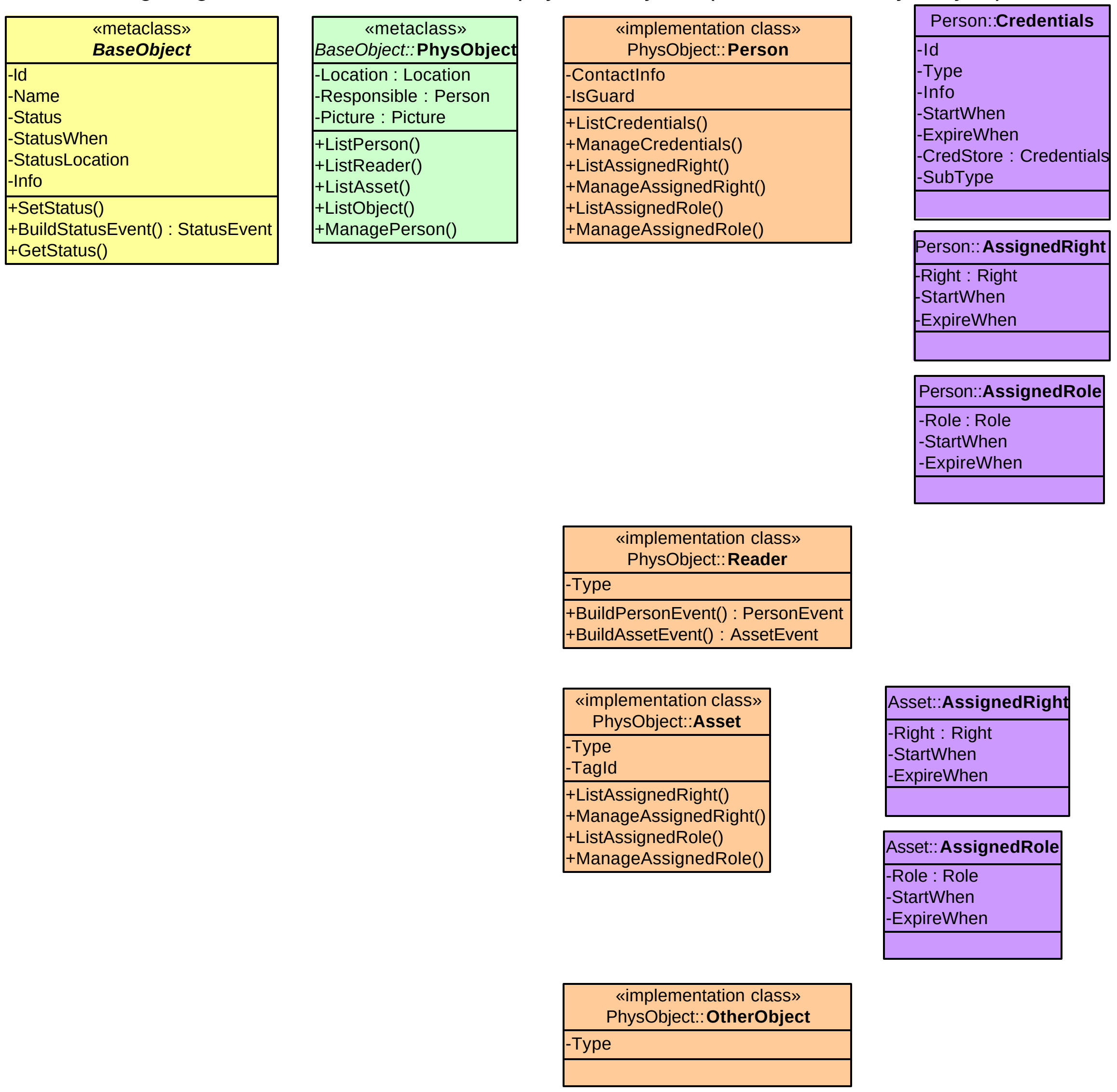

Figure 6 – PHYSBITS Physical Objects

The PHYSBITS physical objects are described below:

Person

BaseObject

Id: unique identifier of the Person
Name: common name of the Person
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: the location assigned to this Person based on the status
Info: additional information about this Person

PhysObject
Location: reference to the Assigned Location of this Person
Responsible: reference to the Person responsible for this Person
Picture: reference to a Picture of this Person

Person
ContactInfo: how to contact this person (address, phone and so on)
IsGuard: Is this person a guard? (and, therefore, can affect “tour” OtherObjects)

Credentials (many per Person)
Id: unique identifier of the credentials
Type: type of credentials (badge, card, certificate, retina, thumbprint and so on)
Info: additional information
StartWhen: when these credentials begin their validity
ExpireWhen: when these credentials expire

AssignedRole (many per Person)
Role: reference to Role object
StartWhen: when this assign becomes valid
ExpireWhen: then this assigned role expires

AssignedRight (many per Person)
Right: reference to Right object
StartWhen: when this right becomes valid
ExpireWhen: when this right expires

Reader (Readers build and submit PersonEvents and AssetEvents)
BaseObject
Id: unique identifier of the Reader
Name: common name of the Reader
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: the location assigned to this Reader based on the status
Info: additional information about this Reader

PhysObject
Location: reference to the Assigned Location of this Reader
Responsible: reference to the Person responsible for this Reader
Picture: reference to a Picture of this Reader

Reader
Type: type of reader (badge, card, retina, certificate and so on)

Asset

BaseObject

Id: unique identifier of the Asset
Name: common name of the Asset
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: the location assigned to this Asset based on the status
Info: additional information about this Asset

PhysObject

Location: reference to the Assigned Location of this Asset
Responsible: reference to the Person responsible for this Asset
Picture: reference to a Picture of this Asset

Asset

Type: Type of Asset
TagId: Tag identifier of the Asset (used in AssetEvent)

OtherObject

BaseObject

Id: unique identifier of the OtherObject
Name: common name of the OtherObject
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: the location assigned to this OtherObject based on the status
Info: additional information about this OtherObject

PhysObject

Location: reference to the Assigned Location of this OtherObject
Responsible: reference to the Person responsible for this OtherObject
Picture: reference to a Picture of this OtherObject

OtherObject

Type: Type of OtherObject

Supporting Objects

The following diagram shows the PHYSBITS supporting objects:

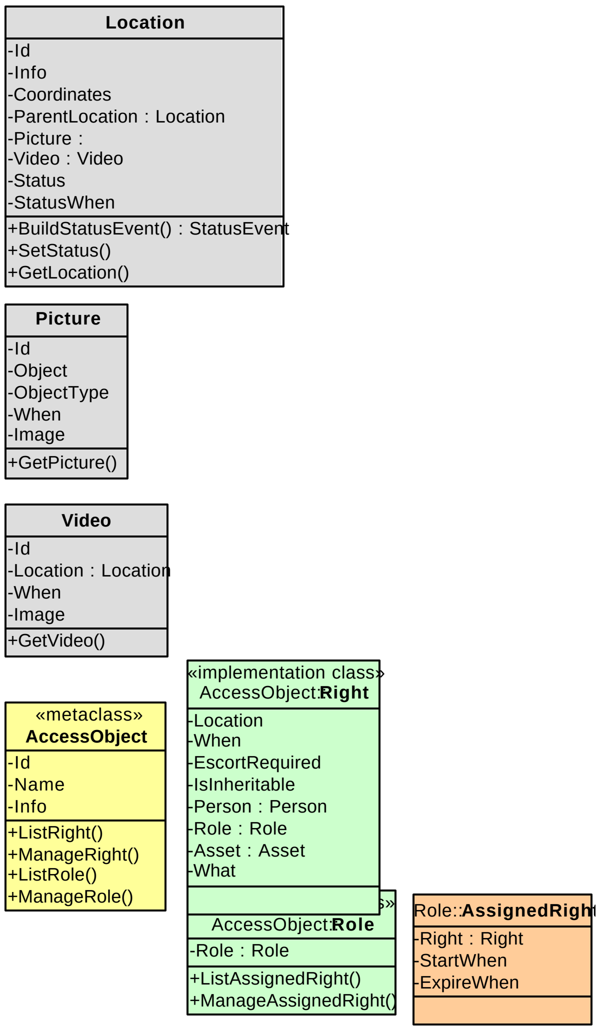

Figure 7 – PHYSBITS Supporting Objects

The PHYSBITS supporting objects are described below:

Location

Id: unique identifier of the Location
Info: additional data
Coordinates: spatial coordinates
ParentLocation: reference to Location of the parent; hierarchical
Picture: reference to a Picture of the Location
Video: reference to a Video object that views this Location
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported

Picture

Id: unique identifier of the Picture
Object: reference to the object that this Picture is of
ObjectType: type of object
When: when the picture was taken
Image: the image

Video (Videos are dynamic objects, retrieved by GetVideo)

Id: unique identifier of the Video
Location: reference to the Location of the Video
When: when the Video was taken
Image: the Video image

Right (Rights are tied to Locations and Time)

AccessObject

Id: unique identifier of this Right
Name: common name of this Right
Info: additional information

Right

Location: reference to Location this Right applies to
When: When the Right is valid (days of week and so on)
EscortRequired: Is an Escort required by Persons accessing this Right?
IsInheritable: Is this Right inheritable by the Location's children?
Person: associated person
Role: associated role
Asset: asset to which this Right grants access
What: what the Rights are

Role (Roles always inherit Rights of their parent Roles)

AccessObject

Id: unique identifier of this Role
Name: common name of this Role
Info: additional information

Role

Role: reference to parent Role

AssignedRight (many per Role)

Right: reference to Right object
StartWhen: when this Right becomes valid
ExpireWhen: when this Right expires

Application Objects

The following diagram shows the PHYSBITS application objects:

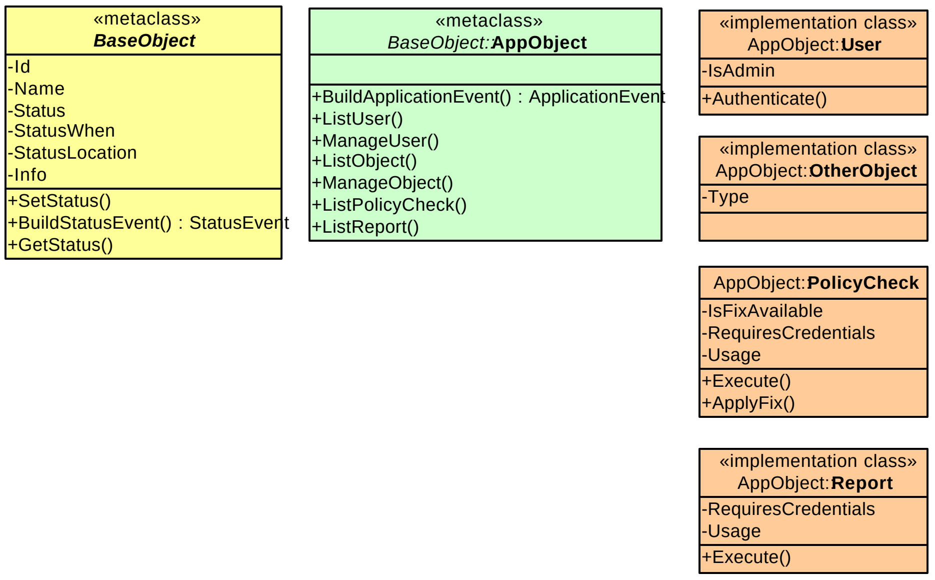

Figure 8 – PHYSBITS Application Objects

The PHYSBITS application objects are described below:

User

BaseObject

Id: unique identifier of the User
Name: common name of the User
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: the location assigned to this User based on the status
Info: additional information about this User

User

IsAdmin: Is this User an administrative user?

OtherObject

BaseObject

Id: unique identifier of the User
Name: common name of the User
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: (not intended for use)
Info: additional information about this User

OtherObject

Type: Type of object

PolicyCheck

BaseObject

Id: unique identifier of the PolicyCheck
Name: common name of the PolicyCheck
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: (not intended for use)
Info: additional information about this PolicyCheck

PolicyCheck

IsFixAvailable: Does this PolicyCheck support “Fixing” a deficiency?
RequiresCredentials: Does this PolicyCheck require Admin credentials?
Usage: the usage (parameters, frequency) of this PolicyCheck

Report

BaseObject

Id: unique identifier of the Report
Name: common name of the Report
Status: last reported status on this object (from a StatusEvent)
StatusWhen: when the last status was reported
StatusLocation: (not intended for use)
Info: additional information about this Report

Report

RequiresCredentials: Does this Report require Admin credentials?
Usage: the usage (parameters, frequency) of this Report

Events

The following diagram shows the PHYSBITS event objects:

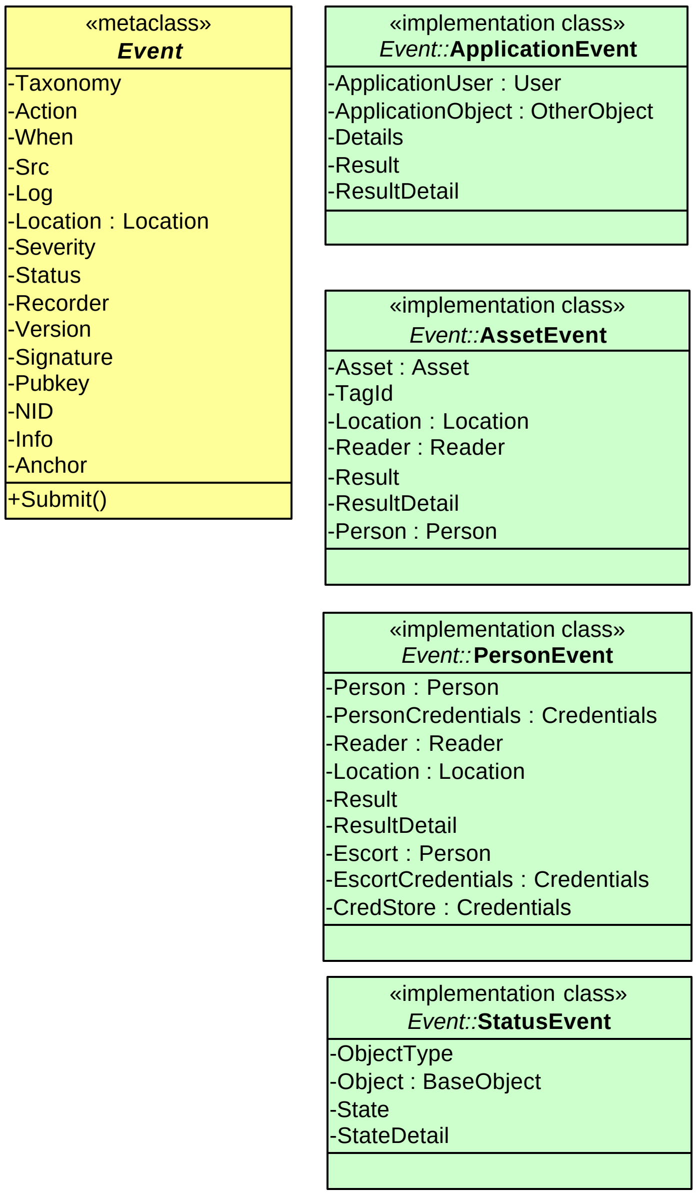

Figure 9 – PHYSBITS Event Objects

The following are descriptions for PHYSBITS event objects:

Base Events

All events have the following attributes:

Event

Taxonomy: event classification
Action: event definition
When: date/time of the event
Src: source of the PHYSBITS-enabled application
Log: well-known name of the event recorder that is providing the events
Location: reference to the Location of the event; blank in ApplicationEvents
Severity: Severity of the event (Info, Warning, Critical, Fatal)
Status: Status of the event (Success, Fail)
Recorder: well-known name of the event recorder (same as log)
Version: Version of the recorder
Signature: a signature on the event; provides tamper-proof events
Pubkey: public key used to verify the signature of the event
NID: native event id in the PHYSBITS application
Info: additional Information
Anchor: Anchor “thrown” by PHYSBITS application to refer back to system

ApplicationEvent

Event

Action: event types, including “Login,” “Logout,” “Access”
Taxonomy: set to “PhysicalSecurity.Application.{Action}”

ApplicationEvent

ApplicationUser: reference to User
ApplicationObject: reference to OtherObject
Details: application-specific details
Result: PERMIT/DENY
ResultDetail: additional information

AssetEvents (AssetEvents are emitted by Readers tracking Assets)

Event

Action: event types, including Enter/Exit/Access/Proximity
Taxonomy: set to “PhysicalSecurity.Asset.{Action}”

AssetEvent

Asset: reference to Asset
TagId: TagId tied to this Asset (read by Reader)
Person: reference to Person (if exists)
Reader: reference to Reader emitting this event
Location: reference to Location related to this event
Result: PERMIT/DENY
ResultDetail: additional information (attempted, authorized, unauthorized, manual and so on)
CredStore: credentials

PersonEvent (PersonEvents are emitted by Readers tracking Persons)

Event

Action: event types, including Enter/Exit/Access/Proximity/Arm/Disarm/Reset

Taxonomy: set to “PhysicalSecurity.Person.{Action}”

PersonEvent

Person: reference to Person
PersonCredentials: reference to Credentials used to validate Person
Reader: reference to Reader emitting this event
Location: reference to Location related to this event
Result: PERMIT/DENY
ResultDetail: additional information (unknown card, wrong clearance, wrong PIN, tailgated, expired, disabled, faulty device, door jammed, more info required, manual and so on)
Escort: reference to Person who was the Escort
EscortCredentials: reference to Escort’s credentials

StatusEvent (StatusEvents are emitted by any BaseObject and Location)

Event

Action: set to the value of StatusEvent::ObjectType
Taxonomy: set to “PhysicalSecurity.Application.{Action}”

StatusEvent

ObjectType: type of object (Person, Asset, Reader, Location and so on)
Object: reference to Object
State: Status based on ObjectType (For example: tour could be start/stop/suspend/resume/cancel)
StateDetail: additional details of the State (manual and so forth)

Other Views

The following views provide other ways to look at the relationships between the PHYSBITS objects.

Person-Centric View

The following diagram shows a person-centric view of the PHYSBIT objects:

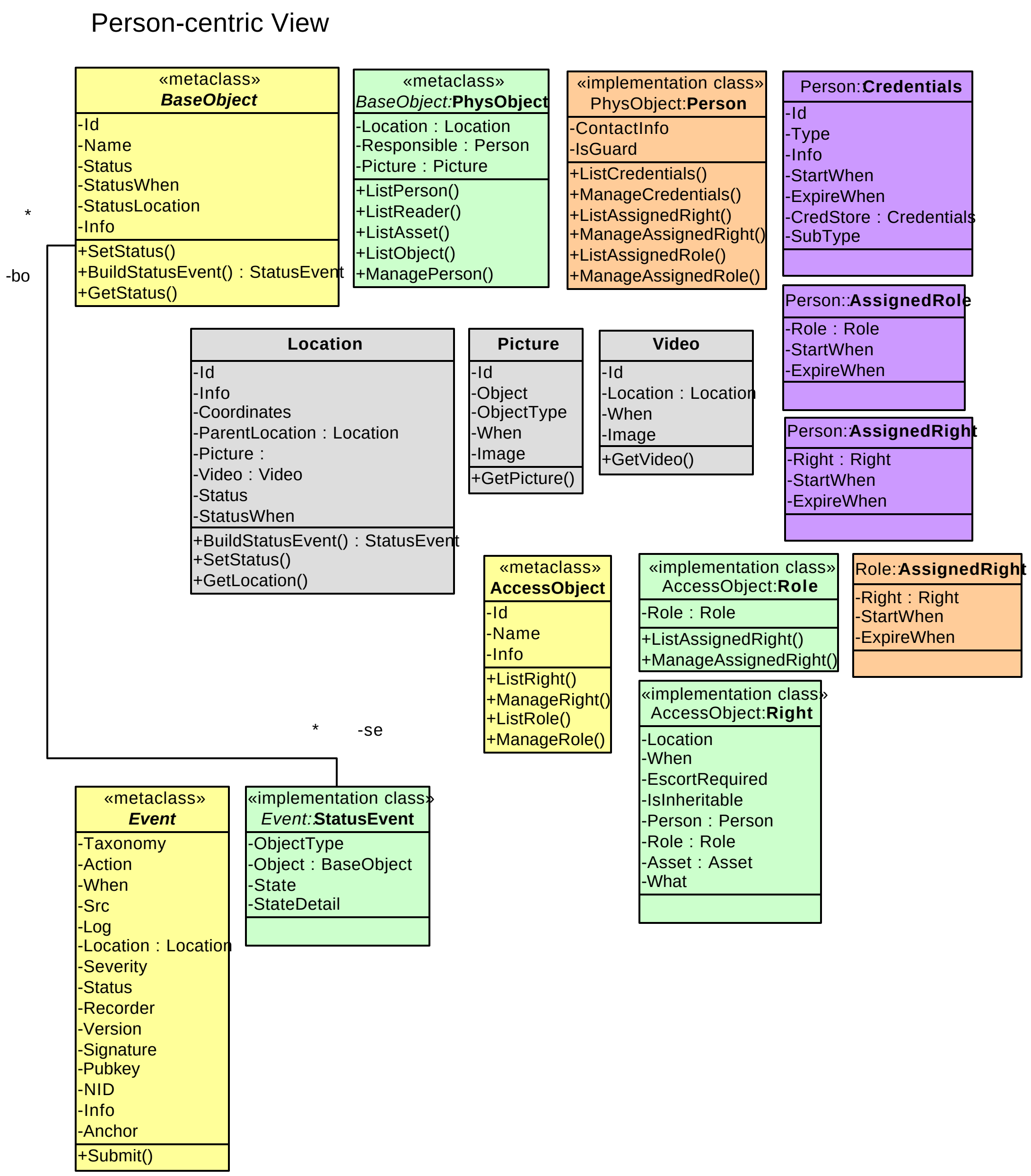

Figure 10 – Person-Centric View

Asset-/Reader-Centric View

The following diagram shows an Asset-/Reader-Centric view of the PHYSBITS objects:

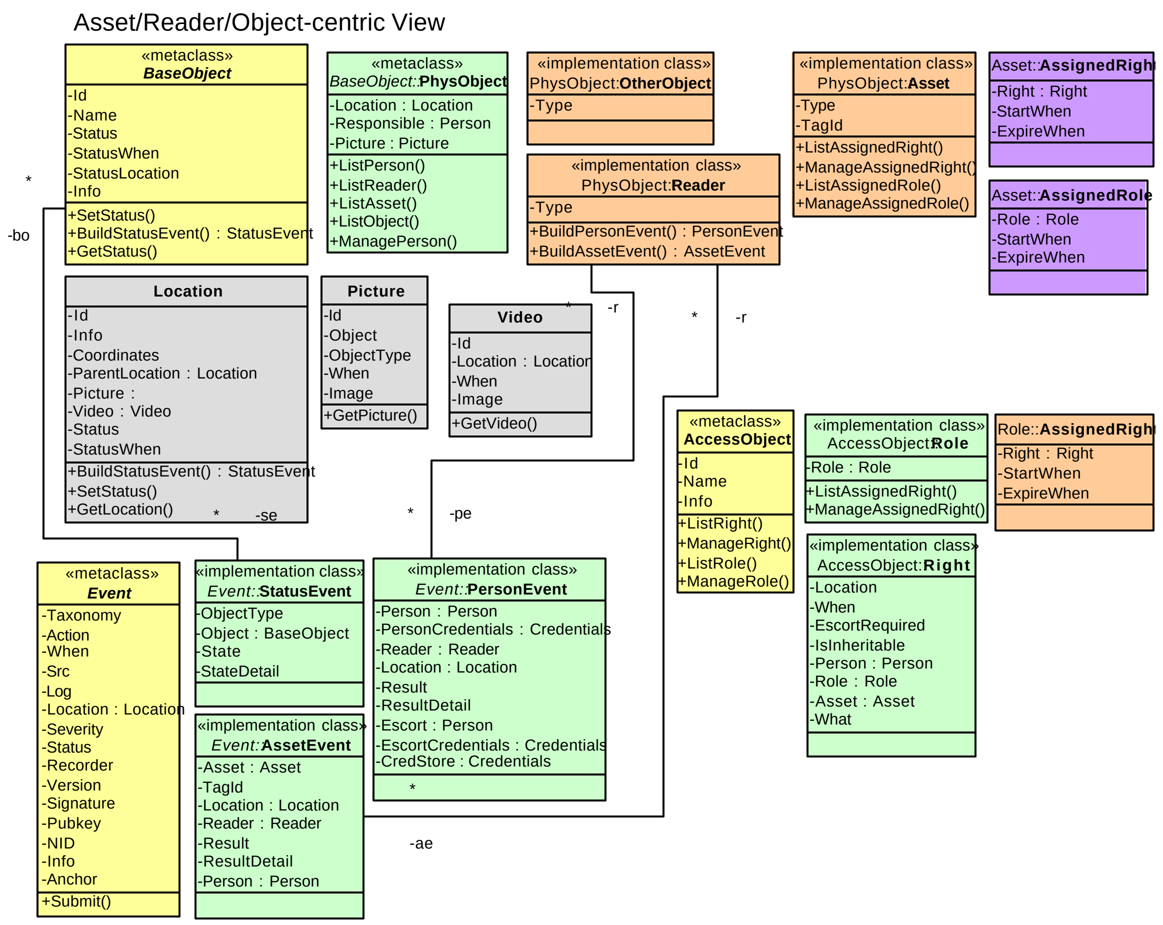

Figure 11 – Asset-/Reader-Centric View

Application-Centric View

The following diagram shows an application-centric view of the PHYSBITS objects:

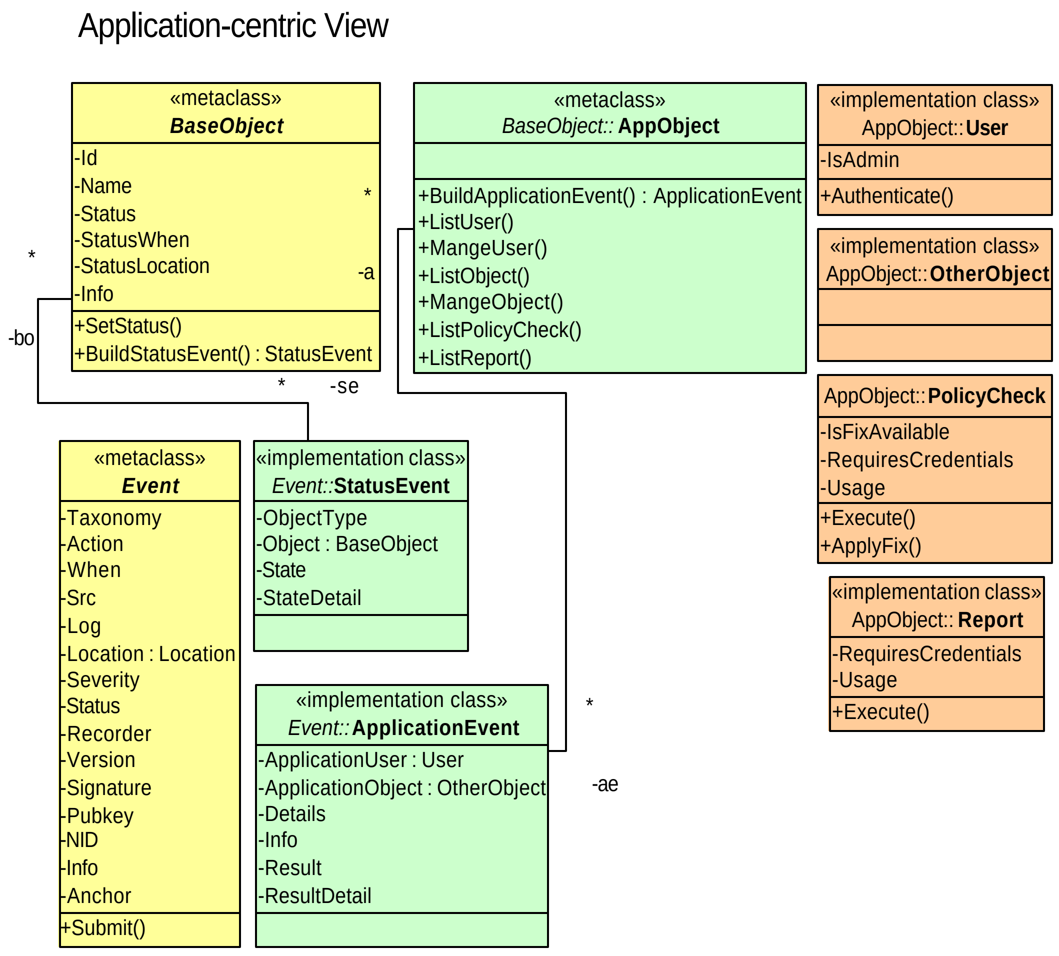

Figure 12 – Application-Centric View

Data Model Usage

Retrieving Entity Data

As described in the data model, the following PHYSBITS entities are available for applications to access:

1) Person (*PhysObject::ListPerson*)
2) Reader (*PhysObject::ListReader*)
3) Asset (*PhysObject::ListAsset*)
4) Other Physical Objects (*PhysObject::ListObject*)
5) Application Users (*AppObject::ListUser*)
6) Other Application Objects (*AppObject::ListObject*)
7) Role (*AccessObject::ListRole*)
8) Right (*AccessObject::ListRight*)
9) Support Data (*Video::GetVideo, Picture::GetPicture, Location::GetLocation*)

Each of the *ListXxx* methods will accept "filters" for limiting the scope of the data returned. The filter format will be defined in a separate API and Data Formats specification. Security management integration requires a conformant Software Development Kit (SDK) for supporting these entity retrieval requests, and for parsing and using the "filters."

User Provisioning

As described in the data model, the following PHYSBITS entities can be viewed/managed by external user provisioning applications:

1) Person (*PhysObject::ListPerson, PhysObject::ManagePerson*
2) Credentials (*Person::ListCredentials, Person::ManageCredentials*)
3) Roles (*AccessObject::ListRole, AccessObject::ManageRole*)
4) Rights (*AccessObject::ListRight, AccessObject::ManageRight*)
5) Assigned Roles (*Person::ListAssignedRole, Person::ManageAssignedRole*)
6) Assigned Rights (*Person::ListAssignedRight, Person::ManageAssignedRight, Role::ListAssignedRight, Role::ManageAssignedRight)*
7) Application Users (*AppObject::ListUser, AppObject::ManageUser*)
8) Other Application Objects (*AppObject::ListObject, AppObject::ManageObject*)

Each of the *ListXxx* and *ManageXxx* methods will accept "filters" for limiting the scope of the data returned. Each of the *ManageXxx* methods will accept "actions" for describing the action to perform (for example, Insert, Delete, Modify, Rename).

The format of the filters and actions will be defined in a separate API and Data Formats specification. Security management integration requires a conformant SDK for supporting these provisioning functions, and for parsing and using the "filters" and "actions."

Event Handling

As described in the data model, PHYSBITS events are built by the following objects:

1) Any object derived from BaseObject may generate a **StatusEvent** by invoking *BaseObject::BuildStatusEvent*.
2) Readers may generate a **PersonEvent** and **AssetEvent** by invoking *Reader::BuildPersonEvent* and *Reader::BuildAssetEvent*
3) Any object derived from AppObject may generate an **ApplicationEvent** by invoking *AppObject::BuildApplicationEvent*

Once events are built, they are submitted by calling *Event::Submit*. The syntax of events will be defined in a separate API and Data Formats specification. Security management integration requires a conformant SDK and associated standardized APIs for sending these events.

Policy Checks

As described in the data model, the following PHYSBITS entities can be used by vulnerability assessment applications:

1) PolicyCheck Listing (*AppObject::ListPolicyCheck*)
2) PolicyCheck Executing (*PolicyCheck::Execute*)
3) PolicyCheck Fixing (*PolicyCheck::Fix*)

Each of the *ListXxx* methods will accept “filters” for limiting the scope of the data returned. The format of the filters will be defined in a separate API and Data Formats specification.

Security management integration requires a conformant SDK for supporting these vulnerability assessment and conformance methods, and for parsing and using the “filters.”

Reporting

As described in the data model, the following PHYSBITS entities can be used by security management applications for running security reports:

1) Report Listing (*AppObject::ListReport*)
2) Report Execution (*Report::Execute*)

Each of the *ListXxx* methods will accept “filters” for limiting the scope of the data returned. The format of the filters will be defined in a separate API and Data Formats specification.

Security management integration requires a conformant SDK for supporting these reporting methods, and for parsing and using the “filters.”

Use Cases for Open Security Exchange – PHYSBITS

It is assumed here that the organization needs to define a set of access rights and roles, and define which roles grant access to which rights.

However, no assumption is made here about implementation details. For example, the issuance of the physical access ID could be done by HR or by a security officer following a workflow process, and require a separate identity check.

Credentials may be stored on- or off-card. User information may be stored in a common distributed directory or federated across multiple data stores.

Provisioning

```
// Set up new hire with a physical access token and cert
// Starting point: new hire is in HR; first day on the job + factory-issued badge with physical access ID
// create/manage Person and set the address, social security number (and so on), and job function
// ( driven by HR application )
personname = new Person
personname::ManagePerson (set HR attributes)
personname::ManagePerson (set ADA requirements)
personname::ManagePerson (set photo)
personname::ManageAssignedRole (set role(s))

// Set building access rights
personname::ManageAssignedRight (access physical location)

// Select a blank identity card for personalization
badge = new personname::Credentials

// Associate the card with the person (FOR RIGHTS NOT ASSIGNED THROUGH ROLES)
badge::ManageCredentials (set physical access ID from badge)

// create cert
cert = new personname::Credential (Badge) …. subcredential stored on badge credential
cert::ManageCredentials ( generate key pair )
cert::ManageCredentials ( generate cert request )
cert::ManageCredentials ( cert issued by CA )
```

Lost Token or Compromised Credential

```
// Locate the badge credential for the user
badgeid = personname::ListCredentials (type=badge)
personname::ManageCredentials (revoke badgeid)

// Issue a new badge
ReplacementBadge = new personname::Credentials
ReplacementBadge::ManageCredentials (set physical access ID from badge)
```

PIN Change

```
// Update credential attribute
Badge = personname::ListCredentials (type=badge)
Badge::ManageCredentials (PIN=value)
```

User on Extended Vacation

```
// Suspend all credentials
credentiallist = personname::ListCredentials
for all cred in credentiallist
   cred::ManageCredentials (suspend cred)
   cred::ManageCredentials (StartWhen = sabbatical end date)
```

User Leaves the Organization

```
// Deprovision user
personname::ManagePerson ( set user as terminated or retiree )

// Revoke all credentials
credentiallist = personname::ListCredentials
for all cred in credentiallist
   cred::ManageCredentials (revoke cred)
```

Delegated Admin (Temporary Employee Take on Role/Rights of Manager)

```
// Set building access rights
managerroles = managername::ListAssignedRole
personname::ManageAssignedRole (managerroles)
personname::ManageAssignedRole (ExpireWhen = date manager returns)

// Set building access rights
managerrights = managername::ListAssignedRight
personname::ManageAssignedRight (managerrights)
personname::ManageAssignedRight (ExpireWhen = date manager returns)
```

Use Physical Token Used to Access a Restricted Area

```
reader::BuildPersonEvent ( action, badgeid, time )
```

Reporting

```
// Find all events for a specific user at a specific reader
badgeidlist = personname::ListCredentials (type=badge, current or expired )
for all badgeid in badgeidlist
    eventlist += reader::ListEvent ( badgeid )
```

Card Life Cycle Management

Introduction

This section addresses some of the major issues that should be kept in mind when selecting cards to deploy in an environment that integrates physical and IT security. Additionally, consideration is given to the requirements associated with *card issuance* and *life cycle management.*

Card Manufacture and Issuance

Cards are manufactured and delivered to the enterprise in a *personalized* state, but without valid *credentials*. The manufacturer will initialize the card so that it can be *issued* in conjunction with an appropriate Smart Card Management System (SCMS).

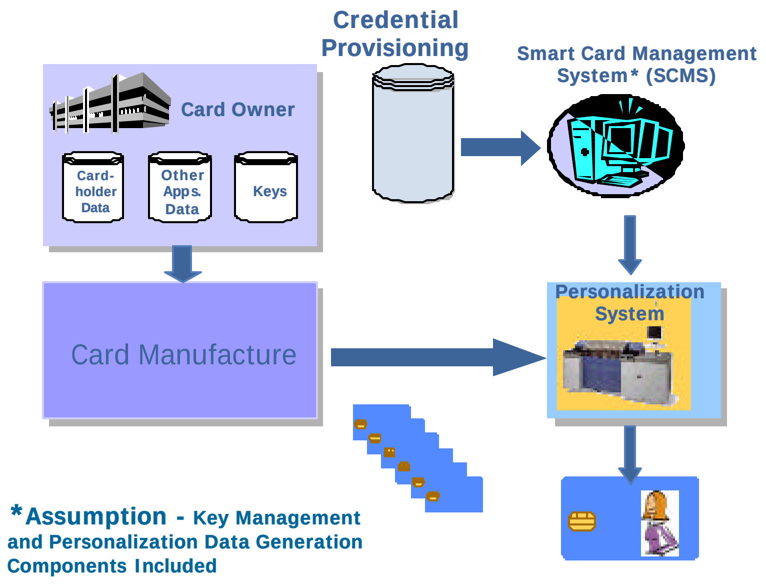
Credential Provisioning
Smart Card Management System* (SCMS)
Card Owner
Card-holder Data
Other Apps. Data
Keys
Card Manufacture
Personalization System
*Assumption - Key Management and Personalization Data Generation Components Included

Figure 13 – Card Provisioning

In addition to managing the activation of applications, the SCMS is typically also responsible for interfacing with the appropriate *Certificate Authority* to obtain appropriate digital credentials that are uniquely associated with the end user and loaded onto the card.

Personalization

Personalization processing consists of three functional areas:

- Data preparation
- Personalization device processing
- Smart card application processing

Each of these areas is summarized below:

- **Data Preparation.** Data preparation is the process that creates the data that is to be placed in the smart card application during personalization. Some of the data created may be the same across all cards, and other data may vary by card. Some data, such as keys, may be secret and may need to be encrypted at all times during the personalization process.

  Data preparation may be a single process or it may require interaction between multiple systems. Much of the definition of data preparation is application-specific.

- **Personalization Device Processing.** The personalization device is the terminal that sends the personalization data to the smart card application. For most smart card applications personalized using this common approach, this device must have access to a security module to create a MAC on commands sent to the application, and to decrypt and re-encrypt secret data.

  All the processing is the same regardless of the application being personalized.

- **Smart Card Application Processing.** The smart card application receives the personalization data from the personalization device and stores it for later use when the smart card application becomes operational. The actual processing of the smart card application prior to personalization will vary by application and must be defined in application-specific documentation.

Identifiers for Hybrid Cards

Hybrid cards are capable of supporting multiple applications, such as contactless chip for physical security, contact chip for logical security and so on.

Different chips and applications on the card might be independently controlling physical and logical security application. Hence, there is a need to tie all these different security credentials to one common unique identifier — managed by a card issuer — which can be directly linked to the cardholder at the time of data creation and/or card issuance.

In view of the above, three different identifiers can be used to effectively manage the card data and link it to the smart card management system, along with credential management systems. The following are three key identifiers, as defined by Global Platform:

Card Identifiers

- **Issuer Identification Number (IIN).** An international identifier for Card Issuers allocated on behalf of the International Organization for Standardization (ISO) and complying with ISO/IEC 7812.
- **Card Image Number (CIN).** A "logical" number controlled by the Issuer to uniquely identify a card (for that issuer only, for example, unique for an IIN) regardless of the target chip and used for deriving final Card Manager keys in back-office key management systems. It is six bytes long and is also defined as the Card Issuer Data field.
- **CAMS Reference Number (CRN).** The CRN is generated by the Card Issuer and is independent of any chip reference, such as the Card Serial Number (CSN) of the contact/contactless chip. The Application Loader in pre-issuance mode to collate the Personalization Data from different Application Providers to be loaded on the same card uses the CRN.

NOTE: *The SCMS generates the CIN and must be linked to the CRN for a specific card. After personalization, the Personalization device returns the CSN, the CIN and the CRN in the log data; the SCMS then links the CRN, CIN and CSN to a physical chip.*

Card Usage — Physical Access

For physical access applications, the unique ID number encoded within the smart card is provided to the access control system via a contactless interface over a small distance. Typically, the user holds the card near a proximity reader, which validates the ID of the card against an Access Control System Database before granting access.

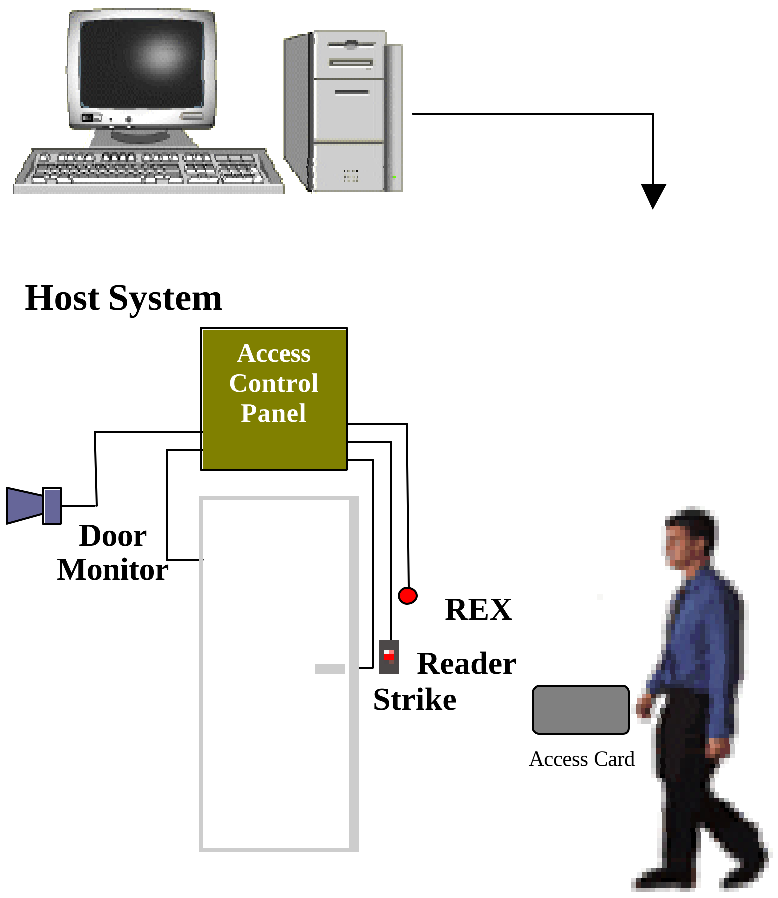

Figure 14 – Physical Access Control

Card Usage — Logical Access

For logical access applications, the user inserts his/her card into a physical coupler. The application software and the reader communicate using a well-defined set of industry standard protocols (ISO 7816, PC/SC).

The application software interfaces with the data contained within the smart card via either of two well-defined token interface standards: PKCS#11 and/or MSCAPI.

The smart card will typically not perform sensitive cryptographic functions unless it has been unlocked by use of a PIN or password, which is input by the user and compared with a known value contained within the protected memory of the smart card.

Therefore, cryptographic operations are not performed unless the user has provided *two factors* for authentication:

- Something he/she *has* (such as the card itself)
- Something he/she *knows* (such as the secret used to unlock the card)

An example logical access use case for a smart card is as follows:

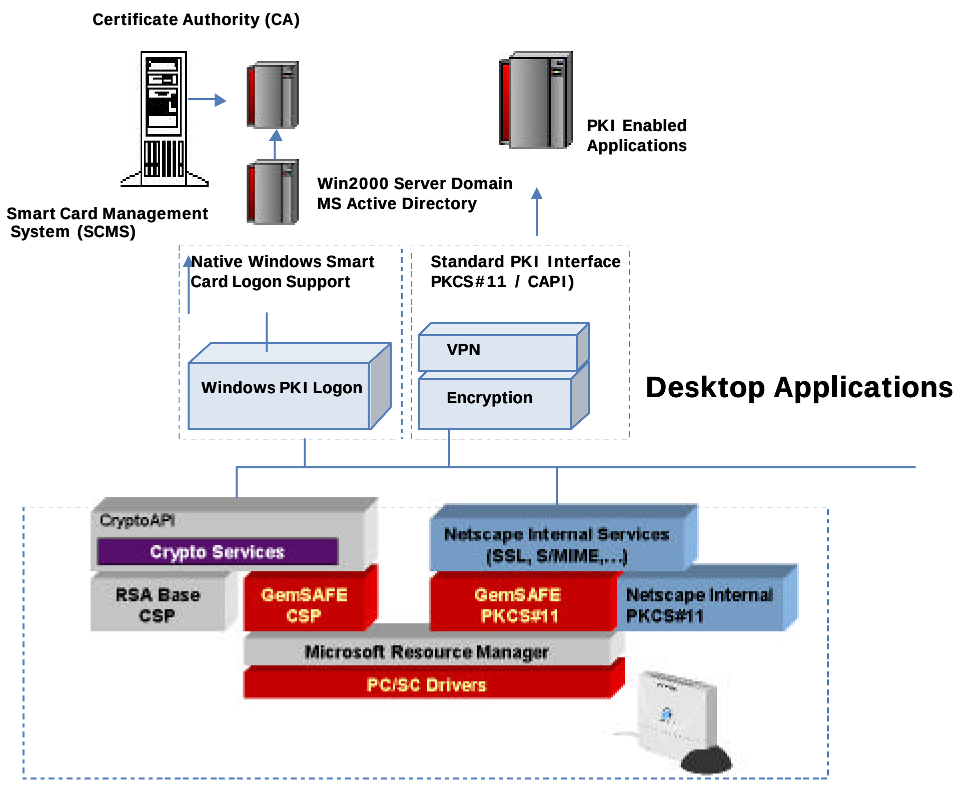
Certificate Authority (CA)
PKI Enabled Applications
Win2000 Server Domain MS Active Directory
Smart Card Management System (SCMS)
Native Windows Smart Card Logon Support
Standard PKI Interface PKCS#11 / CAPI)
VPN
Windows PKI Logon
Encryption
Desktop Applications
CryptoAPI
Crypto Services
Netscape Internal Services (SSL, S/MIME,...)
RSA Base CSP
GemSAFE CSP
GemSAFE PKCS#11
Netscape Internal PKCS#11
Microsoft Resource Manager
PC/SC Drivers

Figure 15 – Logical Access

Access Card Specifications

Introduction

This section provides a set of attributes that are regarded as mandatory for the purposes of card selection and provisioning. Additionally, best practices are identified, where appropriate.

Standards and References

1. Contact Cards
   a. ISO 7816 Parts 1,2,3,4,15
   b. JavaCard 2.1.1
   c. Global Platform 2.0.1'
   d. Visa Open Platform Card Production Guide
2. Contactless Smart Cards
   a. ISO 15693
   b. ISO 14443B
   c. ISO 7810 specifications for length, width, thickness, flatness, card construction and durability, and shall be in a form suitable for direct, two-sided dye-sublimation or thermal transfer printing on the specified badge printer
3. Application and Public Key Certificate
   a. PKCS#1, 7, 11, 15

   b. MSCAPI
4. Cryptography
   a. RSA 512. 768, 1024 bit keys
   b. DES / 3DES
   c. SHA-1
   d. MD-5
5. Contact Smart Card Readers
   a. PC/SC (Serial or USB)
6. Contactless Smart Card Readers
   a. ISO14443 Type B
   b. ISO 15693
   c. UL/C 294

Related Specifications

- NISTIR 6529, Common Biometric Exchange File Format, January 2001. Available from http://www.nist.gov/cbeff
- ANSI/INCITS 358-2002, BioAPI Specification, Version 1.1. Available through the INCITS online standards store, http://www.techstreet.com/ncitsgate.html
- ANSI X9.84-2002, Biometric Information Management and Security for the Financial Services Industry. Available at http://www.x9.org
- NISTIR 6887: Government Smart Card Interoperability Specification, Version 2.0, available from http://smartcard.nist.gov
- FCD 2002 ISO/IEC 7816-11 Information technology – Identification cards — Integrated circuit(s) cards with contacts Part 11: Personal verification through biometric methods
- Biometric API for Javacard – Javacard forum — subject to ballot in INCITS as an ANSI standard

Contactless Smart Card Reader Specifications

Card readers shall be "single-package" type, combining controller, electronics and antenna in one package, in the following configurations:

a. Provide “single-gang” mounting style contactless smart card readers for wall mounting, vehicle stanchions and pedestals
b. The reader shall be of potted, polycarbonate material, sealed to a NEMA rating of 4X (IP65).
c. The reader shall contain an integral magnet for use with an external magnetic reed switch to provide tamper protection when connected to an external alarm system.
d. The reader shall be UL/C 294 listed, and shall be FCC and CE certified, and shall conform to the following ISO Standards: 15693 (read-only or read/write), and 14443B2 (read-only or read/write)*.
e. Transmit Frequency — 13.56 MHZ
f. The reader shall have an approximate read range of 1”– 4.5” when used with the compatible access card.
g. The reader shall require that a card, once read, must be removed from the RF field for one second before it will be read again, to prevent multiple reads from a single card presentation and anti-pass-back errors.
h. The reader shall be capable of reading access control data and transmit that data in SIA standard Wiegand format.
i. The reader shall have a Wiegand output and shall operate under internal control for read-only Access Control Applications*.

j. The reader shall have separate terminal control points for the green and red LEDs, and for the audible indicator.
k. The reader shall have multiple LEDs for increased visibility.
l. The reader shall have an audio transducer capable of providing unique tone sequences for various status conditions.
m. The reader shall have a configurable hold input, which when asserted shall either buffer a single card read or disable the reader, until the line is released. This input may be used for special applications or with loop detectors.
n. Access control data shall be protected using 64-bit diversified security keys, encrypted RF data transmission and mutual authentication.
o. Security keys in the cards and readers shall be required to match and may be customized for individual sites.
p. The reader shall have flash memory to allow future feature enhancements to be added in the field.
q. The reader shall have a lifetime warranty against defects in materials and workmanship.

[*] Alternative acceptable reader configurations:

1. Some Access Control applications may require additional levels of security. To achieve this, a reader as stated above can be used with an integrated keypad.
2. For read and/or read/write applications for connection to PCs or dedicated microcontrollers, a reader as stated above can be used with an RS232 or USB Port.

Contactless Smart Card Credential Specifications

Access Card

a. Access cards shall be used with access readers to gain entry to access controlled portals (such as doors, gates and turnstiles) and to hold information specific to the user.
b. The card shall be available in single technology or multiple technology configurations. Single technology cards shall meet the following criteria:
    i. The card shall meet the following standards for contactless smart cards: ISO 15693 and ISO 14443B.
    ii. The card shall meet ISO 7810 specifications for length, width, thickness, flatness, card construction and durability, and shall be in a form suitable for direct, two-sided dye-sublimation or thermal transfer printing on the specified badge printer.
    iii. Unique 64-bit, fixed card serial number, used for anti-collision and key diversification
    iv. The card shall support read/write capability, with two or 16 Application Areas to support future applications. Data retention shall be 10 years, nominal. Wiegand card data up to 144 bits in length shall be factory programmed in Application Area 1 for use with access control systems.
    v. Each Application Area on the card shall be secured with a unique 64-bit, diversified security key, such that data stored in that area cannot be accessed or modified until the card and reader have completed a mutual authentication process.
    vi. The card shall be capable of completing any write operation, even if the card is removed from the RF field during that operation.
    vii. The card shall have a lifetime warranty against defects in material or workmanship.

viii. The card shall be capable of accepting a slot punch on one end, allowing it to be hung from a strap/clip in a vertical orientation.

c. Multiple technology cards shall meet one of the following additional criteria:

i. 125 kHz Proximity Chip and antenna
ii. Magnetic Stripe
iii. The card shall support 13.56 MHz contactless smart chip and antenna plus an Embedded Smart Chip Module and any or all simultaneously.

Attributes of an Open Security Exchange – PHYSBITS Smart Card

Contact Chip Module Attributes		
Feature	**Mandatory Attributes**	**“Best Practice”**
Protocol	ISO 7816 T=0 and/or T=1	
Memory	Sufficient to store user data	>32K bytes free
RSA Key Length	1024 bit	
Operating System	Any	VM-Compliant with JavaCard 2.1.1
Key Management	Any	Global Platform 2.0 Compliant
Physical Security	Tamper Resistant	CCEAL5, FIPS 140-1 –2 or better
Application Interface	PKCS#11 **and** MSCAPI	
Biometric Support	Not Required	Ability to support Match on Card (MOC)

Contactless Smart Card Attributes		
Feature	**Mandatory Attributes**	**“Best Practice”**
Protocol	ISO 15693 and 14443B	
Transmit Frequency	13.56 MHz	
Key Length	64-bit diversified key structure	
Reader Output	Wiegand or RS232	Wiegand
Plastic Composition	PVC	
Biometric Support	Not Required	Ability to support Match in Reader

Multi-Technology Card
The Open Security Exchange – PHYSBITS solution should use a single credential for ease of use, containing both contact and contactless technology.

Audit Trail, Including Management Best Practices

Introduction

Proper audit trail management is an important element to maintaining any organization's network security. Audit trail management involves the collection, analysis, storage and retrieval of logging information, such as the network traffic, security logs or audit data from the key components of the information technology environment — including applications, operating systems, network routers, firewalls and intrusion detection systems. When correctly managed, audit trails can be key in successfully detecting and preventing attempts by outsiders or insiders to compromise the integrity, confidentiality and availability of an organization's systems and information.

Best Practices

This section contains high-level management best practices to be used when managing audit trail information.

- Develop a clearly defined policy and supporting procedures for managing audit logs. Example policies and procedures include:
    - Audit log retention policy
    - Log monitoring procedures
    - Incident handling and reporting procedures

- Audit log retention: The log retention period varies widely from company to company as well as across industries. It is recommended that each company make its own determination as to what fits its particular environment, based on a risk assessment or industry regulations. As a general rule, most systems should have a retention period of at least six months for audit logs. If a system is deemed mission-critical or highly sensitive, audit logs should be kept for one to two years.

- Log monitoring: The log monitoring procedures should specifically address events to monitor. At a minimum, the following types of events should be included in the log monitoring procedure:
    - Authorized access — attention should be paid to the user ID, date and time of key events, types of events, files accessed and programs used
    - Privileged operations — examples of privileged operations to look for include the use of a supervisor or administrator account, system startup and stop, and I/O device attachment/detachment
    - Unauthorized access attempts — examples of unauthorized access include failed login or resource access attempts, policy violations, and firewall and network intrusion detection alerts
    - System alerts — system alerts to monitor include console alerts, system log exceptions and network management alarms

  The frequency of log monitoring will depend on each of the risks involved and will vary between companies. Risk factors involved in determining frequency of log monitoring include:

    - Criticality of the application to the business
    - The value and sensitivity of the information involved

- Proximity of the server to public networks, such as the Internet

  A general guideline for log monitoring frequency is at least daily for critical or sensitive systems, and once per week for all other systems.

- Establish a central collection point for all audit logs. This allows for an enterprise-wide view of system activity, as well as correlation between disparate systems. The central collection point can be one server or several networked servers. The servers must be secured such that no unauthorized access or disruption could occur to the integrity of the logs. Controls should be put in place to protect against:

  - The logging server or servers from being deactivated
  - Audit logs being altered or deleted
  - Hardware failures to the server or log storage media

  The central collection point should have the ability to archive significant events to a separate, non-networked computer or to read-only media.

- An organization's computer clocks should be set to the proper time and date to help ensure the accuracy of the audit log information. Proper time and date settings are key when performing forensic analysis of events and providing evidence to law enforcement agencies should an incident occur. It is recommended that a centralized timeserver or servers, depending on geographic area, be used to provide the correct time. The central timeserver should synchronize itself regularly with a trusted time source, such as the National Institute of Standards and Technology (NIST).

- Regular backups should be performed of the audit log collection server. Offsite rotation of tapes should be considered for added protection of audit logs.

- Perform a twice-yearly test of audit log management functions. These tests should correspond with a test of incident response and handling procedures.

Data Formats for Normalized Events

Defining events between physical and IT security systems requires an event structure that fits both roles. Physical security accounts for states and functions that exist in the real world, which cannot be mapped directly to IT security environments.

Conclusion

The Open Security Exchange – PHYSBITS approach brings about the convergence of physical and IT security within a consistent and modular framework.

By implementing interfaces as defined in the Open Security Exchange – PHYSBITS Data Model and following specifications, physical security vendors and integrators can combine their applications with IT security processes and supporting technologies.

The framework, data model, access card selection criteria and associated best practices defined in Open Security Exchange – PHYSBITS enable integrated security management solutions. This helps organizations reduce costs and improve their security postures through integrated provisioning, monitoring, auditing and management of physical and IT security systems.

Following documents in the Open Security Exchange –- PHYSBITS series will define Web services and other interfaces, XML data formats and additional best practices. Open Security Exchange – Physical Security Bridge to IT Security (PHYSBITS) will build from existing standards and enable integration of security management within security management, based on secure Web services and XML-based interoperability.

For more information, visit opensecurityexchange.com