



Security by Design

**DESIGN SECURITY INTO THE ENVIRONMENT,
APPLICATIONS AND INFRASTRUCTURE**
September 2005 version 1.5

Nicholas Vennaro
info@AegisSecurityWorks.com

Issue Overview:

Designing security into applications and infrastructure is faster, cheaper and more comprehensive when done at the beginning of the life cycle than treating it as something to be added at the end of the project. This paper outlines various processes and tools that designers, developers, and architects can use to build security into critical applications from the ground up.

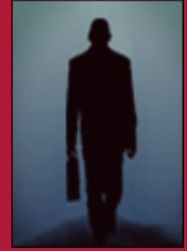


Table of Contents

Summary.....	3
Risk Levels.....	4
“Perimeter” Security.....	4
Application Security Frameworks.....	5
Security Reference Architecture.....	5
Security and the SDLC.....	7
CLASP.....	7
Design Patterns.....	7
Governance Models.....	8
Closing.....	10
References.....	10

Summary

In the government regulated and networked world in which we work it is quite clear that security of your software applications must be treated as a base requirement and can no longer be considered an after thought or a feature that can be added later. The question becomes – and what this article addresses is what can be done across the enterprise to make it easier and more efficient for security to get incorporated into applications.

In Microsoft's IT Department documentation on application security best practices it is required that application security issues be addressed at the beginning of the development lifecycle. Microsoft's research has shown that there are many more high-risk vulnerabilities that result from poor application design than from problems in hardware configurations. It has been my experience that organizations spend more time "locking down" hardware than changing development practices. The reasons for this hardware focus is because device security is easier to implement and less disruptive. The bottom line is, to truly make your applications more secure development practices have to change.

This article explores the following ways to increase your organization's proficiency in the way it builds security into applications:

- **Risk assessment** – it almost seems too obvious to mention but it is often overlooked; the enterprise must first determine what the critical assets are and the threats against those assets before security can be built into the process.
- **Perimeter security** – changes in perimeter security design can affect application development.
- **Security framework** – A design model with tool choices that provide a layer of abstraction between the application developer and the security infrastructure.
- **Security reference architecture** – how the pieces of the security strategy fit together will help communicate enterprise strategy and vision to the various constituencies in the organization.
- **SDLC changes** – modifying your methodology to take security into account from the beginning of the software development life cycle.
- **Governance model** - Implementation of a governance model that leverages architecture principles, design review sessions and the development and usage of design patterns.

Risk Levels

An enterprise risk assessment is the first step for an organization to determine its critical assets, threats against those assets, and a plan to mitigate the threats where appropriate. All too often organizations are spending funds protecting non-critical assets from threats that are not well understood. Following a standard risk assessment methodology will help organizations understand their risks and threats in a structured manner. For a detail analysis of enterprise risk assessment, its value to the enterprise, and a review of the assessment methodologies; the reader is encouraged to review: www.goasw.com/files/EnterpriseRiskAssessmentOverview.pdf.

Once a risk assessment is conducted you can move forward more confidently with your security decisions. You will know what your critical assets are and be assured that the decisions you make will have the desired impact across the enterprise.

“Perimeter” Security

Defense in depth is a security design principle that advocates a layered defense against intruders. Defense in depth similar to concentric growth rings of a tree. Each concentric ring represents a different security zone with varying levels of trust. More sensitive data and applications are kept at the core where security and trust is greatest.

One very typical instance of this defense in depth approach is the use of a Demilitarized Zone (DMZ). The DMZ provides a buffer zone between your internal (more trusted zone) and the outside world (Internet). In this configuration the external requests access the web servers inside the DMZ while the critical data resides on backend devices walled off from direct Internet access. This walling off approach has been and in most cases continues to be the standard approach to network security design.

The model of building an impenetrable shell around your key assets is no longer in favor, as the concept of clear border between internal and external resources, processes, and people has become blurred. Web services, extended channel partner relationships, wireless devices, and remote workers are all adding to the need for a “perimeterless” strategy. For example, web services are problematic for today’s perimeter technologies like firewalls, because the data comes in encrypted and passes through the firewall uncontested.

This change in the security model will have obvious affects on the infrastructure and architecture groups within information technology departments as they grapple with new vendor products, the integration requirements between products, and the design changes that deperimeterization requires. It will become necessary to provide the application development groups with a level of abstraction within your security framework to shield them as much as possible from the changes taking place around the perimeter security.

Application Security Frameworks

The application security framework is an architecture (design) and a set of tool(s) that provide a layer of abstraction between the applications and the underlying systems and infrastructure. The framework affords the organization with many benefits:

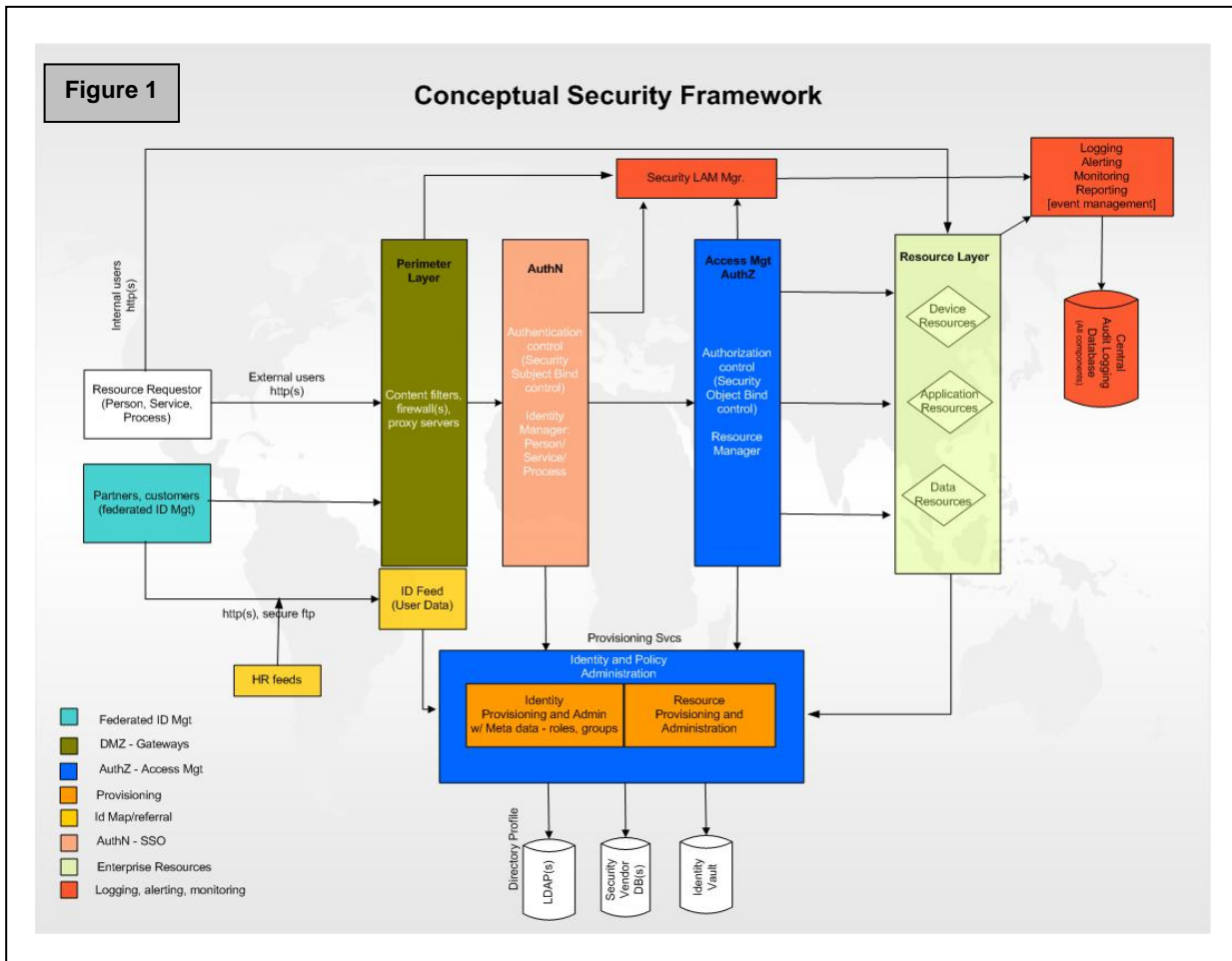
- **Reduced development and QA time** – a security framework gives the engineering groups a set of components to include in their projects. Using these standard components there is less new code to write; faster testing; and streamlined design walk-through processes.
- **Standardized approach** – Component development facilitates usage of a standard approach for key activities across the enterprise. This eases the maintenance burden and allows for a specialization. A dedicated group that is intimately familiar with the intricacies of the components can be formed to manage and maintain these enterprise assets.
- **Abstraction** – developing a framework helps the architects insure that the underlying tools and technologies are abstracted from the calling systems – making tool substitution easier and less disruptive.
- **Layered approach** – A layered approach to security provides a better defense than utilizing a single monolithic design. Using a security framework insures that you have a layered defense across your systems.
- **Enterprise Architecture Management (EAM)** – providing a security framework will make it easier for EAM to tie security into the other key components across the enterprise thereby facilitating security inroads through all aspects of the business.

Using multi-layered reference architecture can provide the organization with the framework necessary to deliver application level security.

Security Reference Architecture

By developing a security reference architecture that fits into your company's overall enterprise reference models will go a long way in explaining the overall strategy and approach the organization is taking with respect to security.

Figure 1 below is the first artifact in a series that documents the security architecture that I have recently created for a client. It provides the overall logical framework that the security components will be constructed from. *Each layer and component in the framework is successively decomposed into further levels of detail until the entire reference architecture is defined.*



Portions of this document based on work by the Burton.

As each of the components in figure 1 are decomposed, the organization can make choices about how the components will be constructed and how the security goals of the enterprise are being met. The security reference architecture provides the schema of the long-term enterprise strategy. Methods, tools, policies, and procedures then fit into this schema.

From a trace-ability perspective, the reference architectures that you produce should have linkages *up* to the business drivers for the enterprise and *down* to the implementation details for the software and infrastructure engineering teams. The construction guidance is further reinforced by the use of design patterns – see design pattern section below.

Security and the SDLC

If you intend to build security into your software process early in the development practice then the software development life cycle (SDLC) needs to address security issues. In short, whatever process your organization uses to develop applications it will need to be modified to address issues such as:

- Authentication and authorization methods.
- Requirements and method(s) for encryption.
- Security requirements documented from the enterprise and project perspectives.
- Security reviews as part of the normal design review process.
- Security as part of the QA test efforts.
- Data classification for this application.
- Building security into the UI, dB, services and other components.

CLASP

The Comprehensive Lightweight Application Security Process (CLASP) is a RUP plug-in that is available to assist developers and architects who want to create a repeatable process for designing and developing security into applications from the beginning of the lifecycle.

I am not sure you would classify CLASP as lightweight but it is fairly comprehensive in nature. It includes 30 activities that are integrated into the software development process. If you are using RUP the integration happens via a plug-in but you can also download a PDF of the process to integrate it into your SDLC methodology if you don't happen to be utilizing RUP. It would be my recommendation to download the list of process items from the company web site (www.securesoftware.com) and then determine for yourself what processes you find most necessary to adopt.

Some of the processes listed would not fall strictly into project level responsibilities within an SDLC. For example, "institute a security awareness program", or "monitoring security metrics", both of these examples, while useful to the enterprise would not need to be the responsibility of an individual project. These types of issues are typically dealt with at an enterprise level.

Overall, CLASP is a good method to get developers and architects thinking about security and how it can be integrated into their processes.

Design Patterns

A pattern is standard solution to a recurring software design problem. Patterns can be used to assist in the security design space. The term design pattern was first used by the "Gang of Four" or simply GoF (Gamma, Helm, Johnson, and Vlissides) in their book *Design Patterns: Elements of Reusable Object-Oriented Software*. A design pattern is in essence a tested solution to common problems. Design patterns

provide many advantages to the organization – faster design/implementations, consistent processes, they provide a more robust and appropriate solution, and design reviews are faster and standardized when patterns are employed.

The concept of design patterns as described by the GoF could be extended to include security topics. A security pattern for encryption might include the acceptable tools and methods that the organization finds suitable for performing encryption. The pattern would include details around implementation so the development community would immediately understand what is required and have a proven method to solve the problem.

The table below outlines the key components (sections) of a pattern.

Name	Component Description
Pattern Name	The name of the pattern which will uniquely identify it.
Description	Brief description of the pattern.
Purpose	The intended purpose of the pattern – what problem will it solve and a description of the solution.
Applicability	Describe when the pattern should be used. Under what circumstances will this pattern be utilized.
Consequences	Describe the costs and benefits of using this pattern.
Entities	Describe the participants/collaborators involved in this patterns implementation.
Implementation	Describe in a technology neutral way how this pattern will be implemented.
Technical Implementation	Specific technical implementation details, including – code samples, object models, sequence diagrams, etc.

Linking design patterns with changes to the SDLC process will provide the organization tools and prescriptive guidance for building secure systems and a standard set of artifacts to review during the governance process to insure that design guidance is being followed.

Governance Models

A governance model that matches the organization’s size and culture is necessary to insure that the security practices that have been put forward are making their way into implementation. It should be noted that security is but one component within the governance model. There are other important areas of concern as well -- business prioritization, architecture, project management for example.

Figure 2 provides an outline of the governance models that AegisSecurityWorks has implemented in large Fortune 100 organizations. This model would be too heavy for a Fortune 1000 company – as stated earlier, you will need to implement what makes sense for your organization.

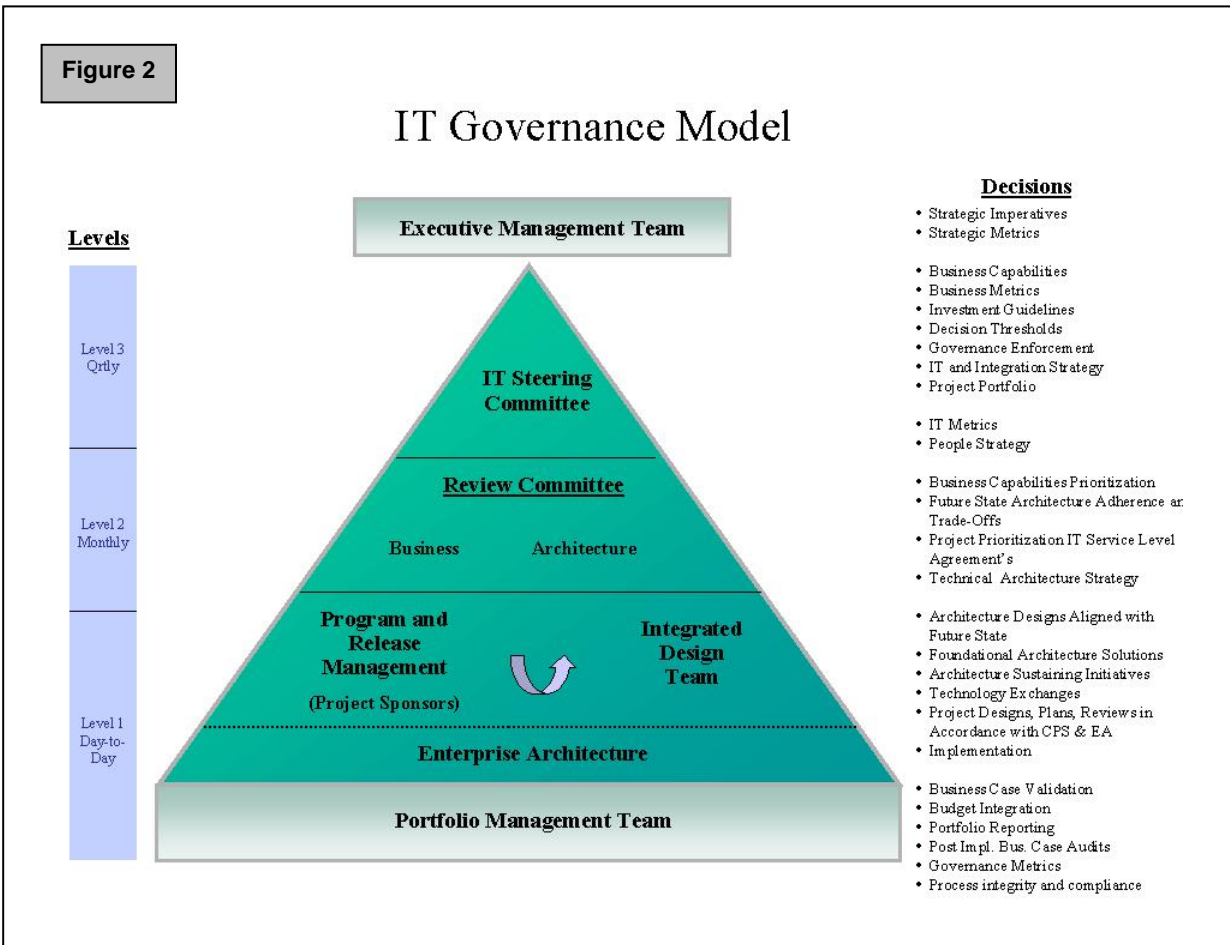


Figure 2 is a framework for an enterprise level governance model. It is the responsibility of the Integrated Design Team to insure that the implementation that is being proposed and ultimately gets constructed conforms to the standards, practices, and processes that have been defined in the reference architecture, SDLC, and in the design patterns. Typically, the integrated design team to catch errors and omissions as early in the process as possible conducts design reviews. This process covers all design aspects including security. The integrated design team would have the following responsibilities:

- Reporting on the architectural health of the project – how well does this project fit into the enterprise reference architecture.
- Technical oversight – It is the responsibility of the design team to provide overall technical supervision.
- Compliance with standards – have design patterns been followed appropriately, has appropriate level of testing been conducted.

Closing

As security requirements become a critical component to business initiatives it is necessary for the organization to institute technologies and methodologies to develop standardized efficient processes for implementation.

Outlined in this paper are several methods that I have used – developing a security framework, usage of design patterns, and changes to the SDLC and governance models – to weave security into the design process earlier in the project life cycle in an efficient fashion.

References

1. *Application Security Best Practices at Microsoft*; January 2003.
<http://iwce.calanza.com/content/AppSecurityWhitePaper.doc>
2. *Information Technology- Code of practice for information security practice*. ISO 17799; December 01, 2000.