A Brief History of IT Security & Architecture

(How Did We Get Into This Mess?)

by

Dr. Steve G. Belovich

CEO, IQware, Inc.
330-659-6300
www.IQware.us
Steve.Belovich@IQware.us

5/18/10

1.0 IT Security Overview

1.1 Recent Security Issues

The past year has witnessed an amazing number of articles, reports, seminars and news stories about successful hacking attempts and the lack of data and/or network security. The GAO recently reported that:

> "Despite indications that agencies have improved their compliance with parts of the Federal Information Security Management Act (FISMA), many major agencies still consider their information security controls a significant deficiency or material weakness" - GAO May 2009

Even "Mighty Google" is not immune and is a target, as this article shows:

> "Ever since Google disclosed in January (2010) that Internet intruders had stolen information from its computers, the exact nature and extent of the theft has been a closely guarded company secret. But a person with direct knowledge of the investigation now says that the losses included one of Google's crown jewels, a password system that controls access by millions of users worldwide to almost all of the company's Web services, including e-mail and business applications." - NY Times, April 19, 2010

A common thread that runs through these articles is the network where the focus is always on intrusion detection and how the network is configured. After all, hackers gained access to Google's code base through a web browser which, in retrospect, seems a huge oversight. **What is totally ignored in analyzing IT security issues is the fundamental engineering & architecture of the IT systems that were penetrated and what can - or should - be done about that.**

1.2 What's *Really* The "IT Security Problem"?

The "ITSP" (IT Security Problem) is a generic term for the problems that arise when trying to achieve a set of operation-related goals. There are six members of that set:

1. IT systems should do exactly what they are intended to do
2. IT systems should operate when intended to do so
3. IT systems should work on behalf of duly-authorized personnel
4. IT systems should NEVER do what is NOT intended
5. IT systems should NEVER operate when NOT intended
6. IT systems should NEVER work on behalf of NON-authorized personnel

2.0 Secure System Requirements: The Reference Monitor

2.1 Requirements For A Secure System

A lot of research was done in the 1960s to figure out how to deal with multi-user protection and preventing unauthorized system access. It was discovered that security had to be designed in to an operating system (and into an IT system) from the ground up. Note that security had two main operational components: (1) multi-user protection and (2) preventing unauthorized accesses. The results of this research revealed the necessary components of a secure, trustworthy system, summarized below:

1) Policy

Security Policy - System must enforce a well-defined security policy.
Marking - System must associate all objects with access control labels (sensitivity & access modes).

2) Accountability

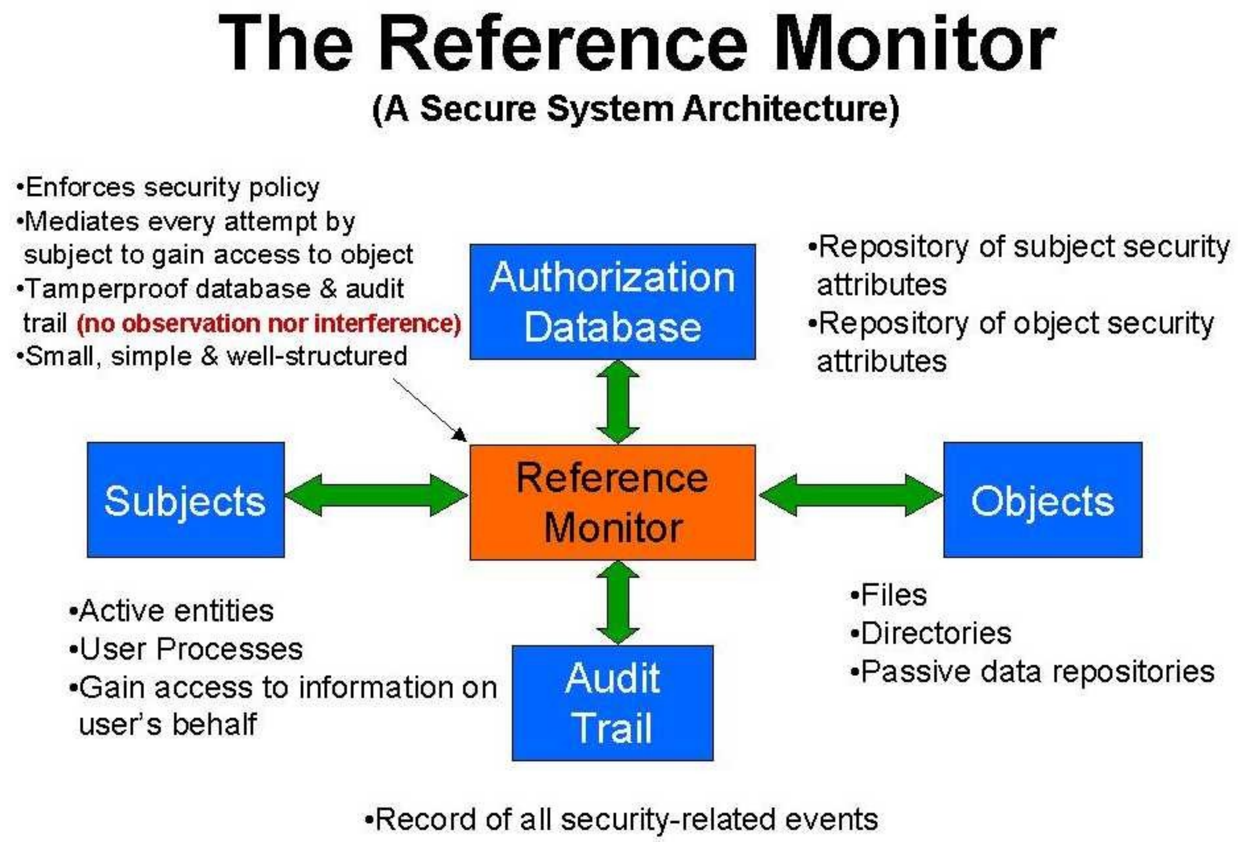
Identification - System must identify individuals and their various authorizations in a secure manner.
Audit Trail - System must keep & protect audit trail so actions may be traced to responsible party.

3) Assurance

Evaluation - System must have hardware/software mechanisms that can be independently evaluated to assure that policy & accountability are enforced.
Continuous Protection - System must continuously protect trusted mechanisms that enforce policy & accountability from tampering.

2.2 The Reference Monitor

As part of these requirements for a secure system, the “Reference Monitor” concept was introduced. This was a logical structure built into the lowest level of the Operating System (O/S) which adjudicated the access of any subject to any object. The Reference Monitor is shown in the diagram below:



The Reference Monitor

(A Secure System Architecture)

- Enforces security policy
- Mediates every attempt by subject to gain access to object
- Tamperproof database & audit trail **(no observation nor interference)**
- Small, simple & well-structured

- Repository of subject security attributes
- Repository of object security attributes

- Active entities
- User Processes
- Gain access to information on user's behalf

- Files
- Directories
- Passive data repositories

- Record of all security-related events

The reference monitor mediates all accesses of objects by subjects. With properly defined subjects and objects, the reference monitor (RM) provides a trusted – and verifiable - security policy enforcement mechanism. The reference monitor, combined with the principles of a secure system architecture, can provide trustworthy, verifiable enforcement of a security policy.

2.3 Where The Reference Monitor Is (or Is Not) Used

Some operating systems incorporated this concept at the lowest layer, right above the FLIH (first level interrupt handler). Most did not. The reason was that the installed base of existing systems prevented radical modifications to the underlying software structures. Making a radical modification to the operating system (O/S) meant rendering obsolete everything that was already sold and shipped. Anyone who tried that would be out of business quickly. The market would never forgive that.

Adding the reference monitor would have been a very radical modification, including the creation of a brand-new file system with object marking. Simply put, the size and economic inertia of the installed base of most computers prevented the fundamental re-engineering and re-deployment required for a truly secure O/S.

3.0 The Desktop Revolution (how RAM & disks got really cheap)

3.1 Consumer Market Economics Limits Design Choices

Cost dominated the retail market - as it still does. The cheapest, simplest design is the one that wins. Security, performance, etc. are all secondary to cost. The PC operating system (O/S) was made simple and dumb with bare minimum support for a file structure.

3.2 System Security Deliberately Eliminated in PCs

When the desktop operating system (O/S) was initially designed, all the security concepts learned in the mainframe/mini world were tossed out because they were not required for the intended use of a home computer. The protection mechanism was physical: lock it up. **Multi-user support, multi-user protection and system security were deliberately eliminated from the design.**

These critical and fundamental mainframe/mini-computer features were not needed for early PCs which were designed for the home market where there would be one user at a time and security was not a concern. Price and convenience drove the design and it still does today. Why spend time and money engineering features that the market did not need, did not want and would not pay for?

3.3 The Invention of the Internet

In parallel with this was the development of the Internet, which was really born out of Dr. Leonard Kleinrock's work at MIT in the early 1960s, along with DARPA (Defense Advanced Research Projects Agency).

The main networking goal in the early days was simply getting it to work! Early protocols were simple and some complexity was added later on to prevent errors such as lockups, and other early “denial of service” situations that had a variety of causes, including a lack of reassembly buffers for lengthy messages.

3.4 Early Network Protocols Ignore Security

Getting the entire network to operate correctly was the goal. Security was not an issue and was largely ignored. The technologies used were intended to be convenient and easy-to-use so that hooking up to the network would be a quick and easy thing to do.

Security was not required for early networks because access was physically controlled. Also, the built-in access control mechanisms of the mainframe or central machine were well-established, well-understood. The network simply presented the access request to the mainframe and it had the responsibility of granting or preventing access.

3.5 PC Operating Systems Had No Secure Foundation

While networking was improving, the initial O/S (Operating System) designs for PCs discarded or ignored the "mainframe/mini" concepts of shared resources, multi-user access, memory protection, multi-layer operation modes (e.g., kernel, executive, supervisor, user), user isolation, file-level access protection, ACLs (Access Control Lists), privileges, quotas, etc.

These engineering concepts were essential for a secure system because they allowed many users to share the computing resources (CPUs, RAM, disks, etc.) without interference. One user's mistakes did not percolate over into another user. The operating system handled all of the housekeeping and ensured that the entire computing system operated correctly even if an individual user did something stupid.

These critical engineering concepts were not included in the architecture of the home computer because they had no **economic** purpose. **Price always drives the consumer market** and there was simply no demand for such features.

The problem is that such features really need to be engineered in from the beginning in order to work properly. Adding them afterward is nearly impossible - and it has not occurred yet in the PC market. We are now living with the consequences of that.

3.6 Networking PCs Requires A Secure O/S

With networking, access control for personal computers became an issue. It now mattered who could access which computer and when they could do that. It now mattered who could access what specific resource of what machine and when. Privileges (what you're allowed to do) and quotas (how much of something you're allowed to use) now became important.

So, the personal computer now needed a secure foundation and it just wasn't there. Usernames, passwords and some limited permission management were the best that could be done. Later, some object access controls were added, but they were easily bypassed because the object marking was not part of the fundamental design of the file system. All such "*ex post facto*" access control mechanisms were crude and easily defeated.

There was no underlying security mechanism for the PC operating system because it was never engineered in the first place. There was no easy way to add it either. The sheer size of the installed base and the economics of the consumer market prevented the much-needed re-engineering of the desktop's operating system. Why bother to create it if you cannot sell it?

As an example, just adding proper “object marking” (a key requirement for a secure system) would require a brand new file system which would force the replacement of the entire installed base of PCs and software. **By way of a benchmark, that installed base is about a trillion dollars. Replacing all that just ain't gonna happen.**

3.7 The Fatal Flaw: Deploying Critical Stuff On A PC

Although the weaknesses of the desktop operating system were well-known early on, IT managers found the technology to be very attractive, convenient, easier to understand and cheaper. After all, if it worked at home, it should be fine for the enterprise, right? “Scale-up” just seemed easy.

So, PCs migrated from the price-driven consumer market to the performance-driven enterprise. It was cheaper, more convenient and there were no monthly computer maintenance fees to pay. All in all, it looked like a smart move for business.

The problem with that move was that the requirements of a business are far different than those for an individual user. The desktop was engineered to meet the needs of the consumer market which wanted the machine for entertainment purposes, web surfing and social networking rather than for “traditional computing”.

Sales-wise, consumer PC sales outnumber business PC sales by nearly 1000-to-1, so that part of the market controls everything else. Clearly, business requires more secure systems but the marketplace is not providing it because it's listening to the consumer side. **Truly effective security is just not possible without fundamentally changing the desktop. That can't happen due to the size of the installed base and the corresponding economics that prevent change. So here we sit!**

4.0 What to Do?

There are no quick fixes to this growing problem. One thing, however, is almost certain. The growing body of lawsuits on software security, safety and reliability issues will lead to federal and/or state regulation. While no one welcomes this prospect, it is due to the inability and unwillingness of the software industry to police itself. The identical thing occurred with the automobile industry in the early 1900s. Now, we have the NTSB (National Transportation and Safety Board) and other organizations charged with ensuring travel safety.

In the meantime here are some helpful suggestions which, if carefully followed, will reduce your risk.

1. Recognize that desktop technology is not secure and was never intended nor designed to be secure. So, do not deploy critical applications on such systems. Just don't do it. The desktop is best suited to serve as an interface to a centrally-managed, secure application using a very thin-client architecture.
2. If something is available via a web browser, it can be hacked. All web browsers on desktop operating systems are vulnerable. So, do not allow browser-based access to anything critically important. Use a thin client like XLIB for desktop to support a centrally-managed GUI rather than a browser.
3. Understand that your network will always be polluted to some extent. The TCP/IP protocol is flawed because it permits challenge/response without authentication. So, it will always be possible to do remote foot-printing, scanning and enumeration – which are the three essential steps in the hacking process. Proper firewall configuration – and the use of only "stateful" firewalls – will help a lot but cannot completely prevent unauthorized traffic.
4. Deploy critical applications only on secure O/S platforms. If the O/S itself is not secure, the application deployed on top of it cannot be secure.
5. Spend the bucks to design new systems right the first time. It is never cheaper to redo. Also, the opportunity cost of not having the system deployed properly can be huge.
6. Plan for "rolling upgrades" with system segmentation. Use multi-vendor standards for GUI, database access and network communication. That way, you can upgrade portions of your system without disturbing the rest of it. Multi-vendor standards ensure that you have alternative sources for critical pieces of software. If you cannot get access to your data unless it's through a single-vendor's proprietary interface, shy away from that.
7. Keep the architecture flexible so it can adapt as your business needs change. You want your IT systems to enable your organization - not limit your growth.
8. Choose stuff because it works and it's reliable – not because it's cheap or convenient. The money that you save will far outweigh the little extra in up-front cost.
9. Call (330-659-6300 x221) or email (Steve.Belovich@IQware.us) us at IQware because solving this issue is what we do. We can help!