SOFTWARE ASSURANCE BEST PRACTICES FOR AIR FORCE WEAPON AND INFORMATION TECHNOLOGY SYSTEMS – ARE WE BLEEDING?

THESIS

Ryan A. Maxon, Major, USAF

AFIT/GIR/ENV/08-M13

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

AFIT/GIR/ENV/08-M13

SOFTWARE ASSURANCE BEST PRACTICES FOR AIR FORCE WEAPON AND INFORMATION TECHNOLOGY SYSTEMS – ARE WE BLEEDING?

THESIS

Presented to the Faculty

Department of Systems and Engineering Management

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Information Resource Management

Ryan A. Maxon, BS

Major, USAF

March 2008

AFIT/GIR/ENV/08-M13

SOFTWARE ASSURANCE BEST PRACTICES FOR AIR FORCE WEAPON AND INFORMATION TECHNOLOGY SYSTEMS – ARE WE BLEEDING?

Ryan A. Maxon, BS

Major, USAF

Approved:

//Signed//	17 March 2008
Dennis Strouble, PhD (Chairman)	Date
//Signed//	17 March 2008
Michael R. Grimaila, PhD (Member)	Date

Abstract

In the corporate world, "bits mean money (McGraw & Viega, 2000)," and as the Department of Defense (DoD) becomes more and more reliant on net-centric warfare, *bits mean national security*. Software security threats are very real, as demonstrated by the constant barrage of Internet viruses, worms, Trojans, and hackers seeking to exploit the latest vulnerability. Most organizations focus their resources on reactive defenses such as firewalls, antivirus software, and encryption, however as demonstrated by the numerous attacks that are successful, those post facto measures are not enough to *stop the bleeding*.

The DoD defines software assurance (SwA) as the "level of confidence that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software (Polydys & Wisseman, 2007)." SwA focuses on *baking in security* versus *bolting it on* afterwards. The Department of Homeland Security and DoD each have had SwA programs for a few years; however the Air Force (AF) just recently formed the Application Software Assurance Center of Excellence at Maxwell AFB-Gunter Annex, AL.

This research seeks to identify common issues that present challenges to the development of secure software and best practices that the AF could adopt as it proactively begins to *heal* the SwA problem.

Acknowledgments

First and foremost, I wish to thank my wife and daughters for the enduring love and support throughout our time at the Air Force Institute of Technology (AFIT). I would also like to thank Colonel David Handle (retired), who was instrumental in my selection to attend AFIT, Mr. Joe Jarzombek (Department of Homeland Security), Mr. John Neumann (Government Accountability Office), Colonel Kevin Foley (HQ 754 ELSG), Captain Michael Kleffman (HQ 754 ELSG), Ms. Karen Goertzel (Information Assurance Technology Analysis Center), Colonel Patrick Condray (AFWA/CC), Mr. Walt Coley (AFWA/A6XX), Lt Col David Gindhart (754 ELSG/GC), Ms. Sandy Speake (HQ AFMC/A8), Mr. Randy Adkins (HQ AFMC/A8) and my AFIT classmates and professors.

Finally, I'd like to thank my thesis advisor, Dr. Dennis Strouble, and my reader, Dr. Michael Grimaila.

The dogmas of the quiet past are inadequate to the stormy present. The occasion is piled high with difficulty, and we must rise with the occasion. As our case is new, so we must think anew and act anew. - Abraham Lincoln, Annual Message to Congress, Concluding Remarks, 1 December 1862.

Ryan Maxon

Table of Contents

Page

Page

Page

List of Figures

Page

Page

List of Tables

Page

SOFTWARE ASSURANCE BEST PRACTICES FOR AIR FORCE WEAPON AND INFORMATION TECHNOLOGY SYSTEMS – ARE WE BLEEDING?

I. Introduction

Background

Companies are increasingly growing their e-commerce presence on the Internet, which requires them to correspondingly focus on security. Since "bits mean money," companies must protect those bits in order to be successful. If those companies fail at security, there may be disastrous consequences, including a big red mark on their fiscal health. (McGraw & Viega, 2000) As the Department of Defense (DoD) moves more and more towards net-centric warfare and the United States Air Force (AF) increasingly grows its cyber presence and cyber warfare capabilities, those bits mean national security and any software security failures on their part, could likewise have disastrous real-world repercussions. National security requires information technology (IT) security.

Software security threats and costs are very real, as the following incidents, facts and recent events illustrate:

- Business losses and damages due to Internet viruses and worms…
    - 2000 Love Bug: $15 Billion in damages; 3.9 Million systems infected
    - 2001 Code Red: $1.2 Billion in damages; $740 Million for recovery efforts
    - 2002 Slammer: $1 Billion in damages
    - 2003 Blaster: $50 Billion in damages
    - 2004 My Doom: $38 Billion in damages (Jarzombek, Security in the Software Lifecycle, 2007)
    - 2005 Zotob: $97,000 per infected business in average recovery costs; victims included CNN, ABC, and the New York Times (Gilbert, 2005)
- A Microsoft Passport service flaw discovered in 2003 allowed potential attackers access to accounts at will. Passport is used by 200 million users and Microsoft could have faced up to $2.2 trillion in fines from the Federal Trade Commission due to the flaw. (Skibell, 2003)

- Firms lose an average of 2.1 percent of their market value within two days of announcing a security breach. This is equal to an average loss of $1.65 billion in market capitalization per breach. (Cavusoglu, Mishra, & Raghunathan, 2004)
- A 2004 study showed that systems (with default settings) and no other protection were compromised within 30 seconds to 4 minutes after being connected to the Internet. Interestingly, the same study showed that systems that were up to date and/or had some protection were ignored by potential hackers. (Keizer, 2004)
- In 2005, 87 percent of survey respondents indicated their organization had at least one computer security incident in the previous year, with 20 percent of the respondents indicating there had been 20 or more incidents in the previous year. (Verduyn, 2006)
- 2006 saw the dawn of phishing, virus, and texting attacks targeted at mobile devices (cell phones, etc.). While individual users typically pay the cost of the unsolicited text messages, each incident costs mobile operators too; they lose the entire year's profit from that one customer to deal with the cost of network or device cleaning, customer service, and revenue disruption. (Shah, 2007)
- The Federal Bureau of Investigation's Internet Crime Complaint Center processed more than 200,000 complaints in 2006 and referred 86,279 to law enforcement agencies for further consideration; of the referred cases, the total dollar loss was $198.44 million (up from $183.12 million in 2005) with a median loss of $724 per complaint. (IC3, 2007)
- Analysis of all critical vulnerabilities reported in 2006 showed that three programming errors were responsible for 85% of those critical vulnerabilities. (SANS, 2007)
- Sophos, an IT company specializing in security, discovered an average of 6,000 new infected websites every day in 2007; this is equivalent to one every 14 seconds. Of those, 83 percent were legitimate websites that were compromised by an unauthorized third-party. The others were hacker sites, i.e., malicious in intent. One of the hacked legitimate websites was the U.S. Consulate General's in St Petersburg, Russia, which highlights even security-conscious organizations can fall prey. (Sophos, 2008)
- 95 percent of all e-mail was spam in 2007. Virtually all the spam originated from compromised computers ("bots" or "zombies"). (Sophos, 2008)
- Approximately 11 percent of computers world-wide are infected by bots, with botnets growing by about 500,000 new computers every 24 hours. (PandaLabs, 2008)
- 8 percent of all e-mail traffic contains links to malicious websites. (PandaLabs, 2008)
- The wireless networks of 800 retail stores in New York City were scanned in January 2008 by the wireless security vendor AirDefense; one third were found to have no security and another third were found to only have weak encryption. It is important to note that weak wireless security was the entry point for the hackers that stole 46 million credit card numbers from TJX Corporation. (Cox, 2008) The January 2008 findings by AirDefense were actually an *increase* in numbers over a similar 2007 study of 3,000 retail stores in 8 major U.S. and European cities. (McNamara, 2007)
- Also in January 2008, the Central Intelligence Agency disclosed that they had information of "cyber intrusions into utilities, followed by extortion demands ... that

> cyber attacks have been used to disrupt power equipment in several regions outside the United States… [and in] at least one case, the disruption caused a power outage affecting multiple cities." (SANS, 2008)

Current Trends

Previously, in most cases, vulnerability information was either privately turned over to the vendor responsible for the particular product, or the information was sold secretly on the black market. Now vulnerabilities are often publicly exposed before the responsible vendor gets a chance to examine the flaws. Also, a new venue exists: in 2007, a new online auction site called WabiSabiLabi was launched for the purpose of *publicly* selling vulnerability information to the highest bidder (Associated Press, 2007). The number of publicly acknowledged software vulnerabilities is rising fast:

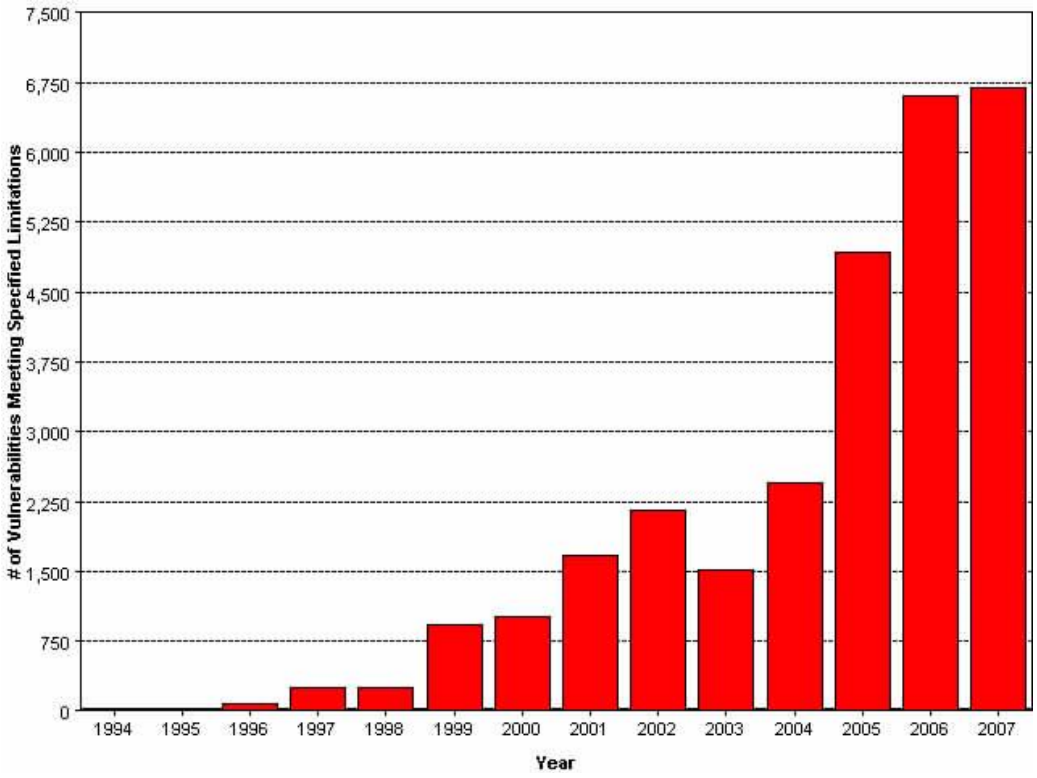

Figure 1. Software Flaws for All Vendors, Products, and Versions from January 1994 to December 2007 (NIST)

Even after a fix is released for the vulnerabilities, there's no guarantee that the fix will be implemented right away, or even implemented at all. Data from a case study by William Arbaugh, William Fithen, and John McHugh (2000) show that "intrusions increase once a vulnerability is discovered, the rate continues to increase until the vendor releases a patch, but exploits continue to occur even after the patch is issued (Viega & McGraw, 2001)." The following figure illustrates the increases in intrusions over time (based upon the case study data):

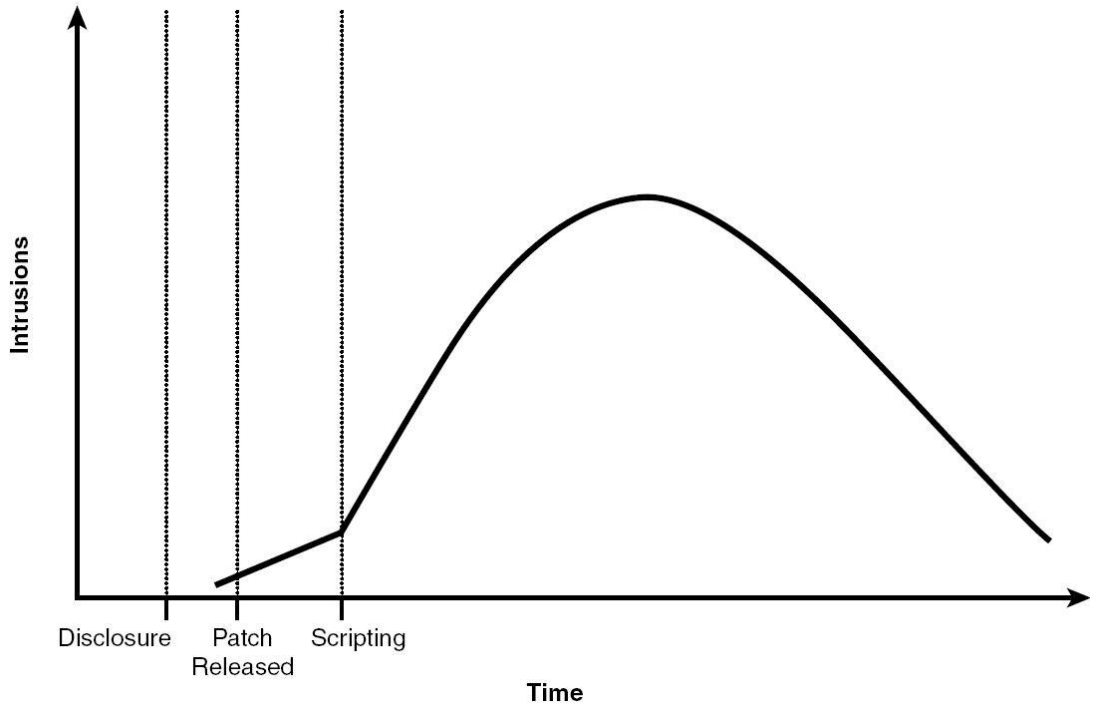

Figure 2. Average Number of Intrusions for a Security Flaw Over Time (Viega & McGraw, 2001; Arbaugh, Fithen, & McHugh, 2000)

Arbaugh, Fithen, and McHugh (2000) also described how "a system typically oscillates between hardened and vulnerable states," as shown in the following figure:

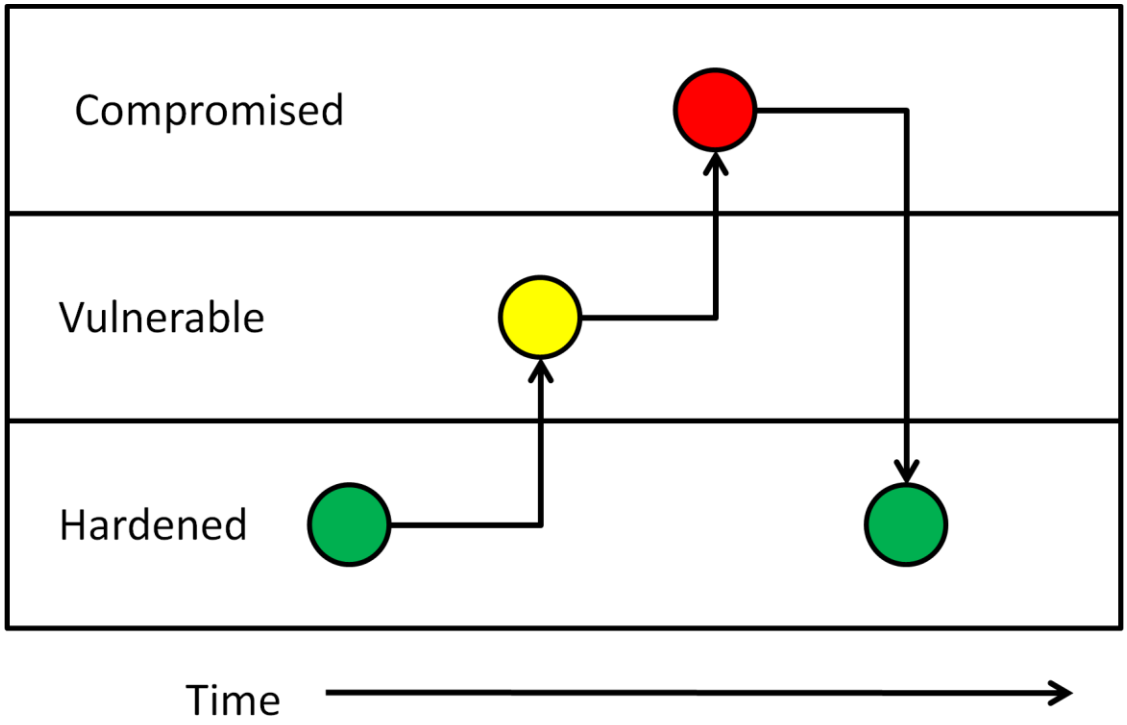

Figure 3. Oscillating Host Lifecycle (Arbaugh, Fithen, & McHugh, 2000)

The Slammer worm infiltrated the unpatched automatic teller machine networks of Bank of America and Washington Mutual six months after Microsoft had released a patch that addressed the vulnerability (Kelley, 2003). A recent survey of Oracle Database Administrators revealed that only 10 percent had installed the latest Oracle Critical Patch Update (CPU) and 67.5 percent had never applied any Oracle CPU. (Sentrigo, 2008) Updating software can be costly: for one German company that has 8,000 databases, one patch can require 32,000 hours of install labor, plus additional extensive testing on each server to ensure there were no unintended side effects (Lemon, 2007). When time or resources are not available to test and/or install the patch, administrators may just forego the patch. Some may just choose to not mess with it (or may be told by their boss to not mess with it) if the system has been running fine without it. Additionally, some

administrators are “skittish about automatically slapping the latest patches on their production servers” because some patches (not just database patches) have been known to undo previous fixes, crash the system, or introduce new flaws. (Kelley, 2003; Landwehr, Bull, McDermott, & Choi, 1994; McGraw, 1999; Vijayan, 2008)

How often updates are released is another concern. Microsoft releases their product updates on the second Tuesday of each month (“Patch Tuesday”), to which malware creators have adapted by waiting until the second Wednesday in each month to exploit unknown vulnerabilities. This allows the malware creators to achieve a higher impact by giving them a whole month until the next update is released, which may or may not address the vulnerability. (PandaLabs, 2008) The scope of Microsoft’s decision to update products once per month can have a global impact, especially when you consider that approximately 90 percent of the computers on the planet may seek updates all on the same day. In fact, in April 2004 the Microsoft Windows Update site spiked up to 4 million users per hour, which was nearly twice as high as it ever had gone. This resulted in many users not being able to download the latest patches. Microsoft resolved the problem by doubling their update server capacity. (Lemos, 2004)

Keeping track of all the vulnerabilities that affect your systems is another difficult task, which is readily apparent in statistics released in January 2008 by Secunia, an IT security company:

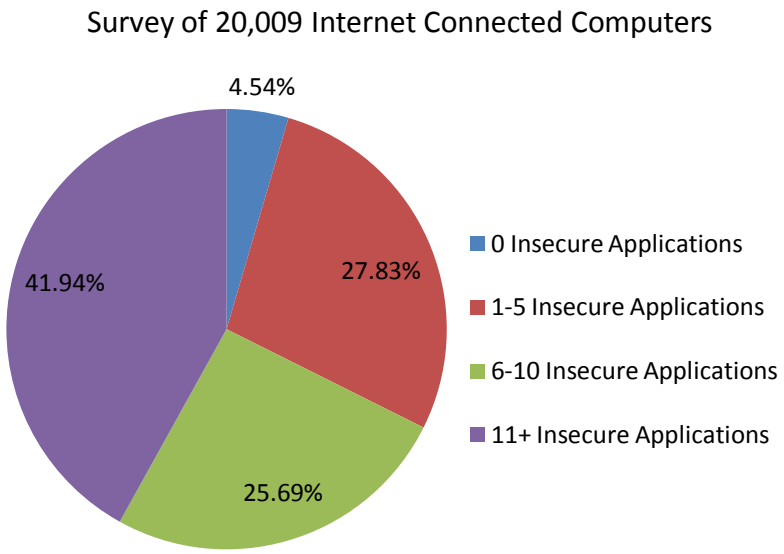

Figure 4. Secunia Personal Software Inspector (PSI) Scan Results (Balle, 2008)[1]

Within the DoD, the post facto or 'reactive' software security effort is led primarily by the Joint Task Force Global Network Operations (JTF-GNO) unit. Tracking vulnerabilities and alerting users is difficult for the DoD as well. The JTF-GNO provides antivirus software to the entire DoD community and distributes vulnerability notices to the services via Information Assurance Vulnerability Alerts (IAVAs), Information Assurance Vulnerability Bulletins (IAVBs), and Information Assurance Vulnerability Technical Advisories (IAVTAs) that cover one or more Common Vulnerability and Exposure (CVE) notices. In 2007, JTF-GNO issued 57 IAVAs, 37 IAVBs, and 53 IAVTAs. (JTFGNO, 2008) These issuances accounted for a total of 524 CVEs (dated from 2004-2007), a number far below the nearly 6,750 CVEs issued in 2007 alone. How certain is JTF-GNO that the other 6,000+ CVEs didn't apply to DoD systems?

[1] The source article indicated that 27.83% had 0-5 insecure applications, which was a typographical error. It should have read 27.83% had 1-5 insecure applications, which is as shown in this pie chart.

Furthermore, for the period from 1998 through mid-2006, IAVM notices "[did not] include 93 percent of the publicly-known root-level compromise vulnerabilities, and [did not] include 98 percent of the publicly-known vulnerabilities (Schneider W. J., 2007)."

McAfee, another IT security company, recorded over 100,000 new viruses and Trojans in 2007, which was a 50 percent jump in the total number of threats ever cataloged (McAfee Avert Labs, 2007). Parasitic viruses are just one type of viruses studied by McAfee and they are expected to grow by 20 percent in 2008, as shown in the following chart:

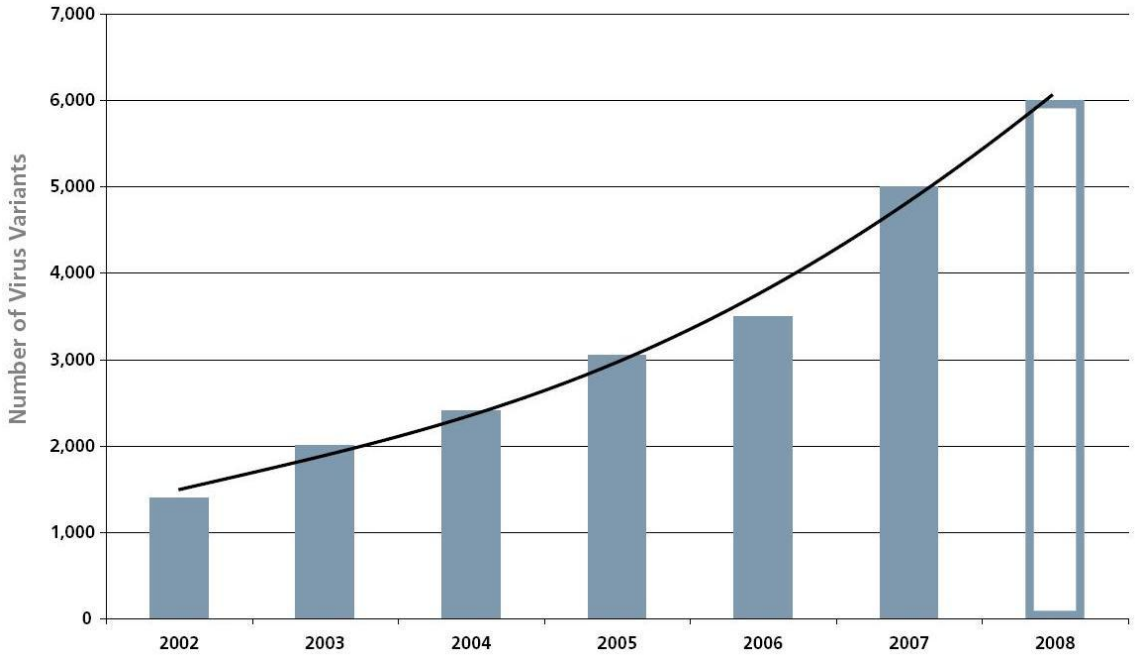

Figure 5. Parasitic Virus Growth by Year (McAfee Avert Labs, 2007)

Problem Statement

Security on the Internet has been a known problem for years, so why does this problem still exist? Why does security seem to be getting worse? Why are computer networks, including the military's network, getting hacked? The answer is simple: most

of the security effort is focused on treating the symptoms, not *curing the root of the problem*. Computer security coverage in books, journals, and popular press center around the basic technology issues: firewalls, cryptography, antivirus products, and the "hot topic of the day" such as a specific virus or denial-of-service attack (Viega & McGraw, 2001). Too many organizations believe that they're safe because they are behind firewalls; however, since web servers often being implemented inside the organization's firewall, web requests with attack code are simply passed through the firewall (Epstein, Matsumoto, & McGraw, 2006). Also, the firewalls themselves may be based on fallible software (Viega & McGraw, 2001). And while cryptography is "an important weapon in the software security arsenal," "it is insufficient for security (McGraw, 2003)." Furthermore, less than 15 percent of all Computer Emergency Response Team advisories could have been fixed or avoided using cryptography (Schneider F. B., 1999).

While the media generally misses the mark, behind every computer security problem and malicious attack lays a common enemy: bad software (Viega & McGraw, 2001). Lawrence Hale (former deputy director of the Department of Homeland Security's U.S. Computer Emergency Response Team) said that "the things that are costing us the most pain are preventable (Olsen, 2004)." Independent security consultant Shawn Moyer said recently at the Black Hat Federal Briefings that "It's the same problem over and over again. We patch, we scan, we patch, we scan, and the cycles get shorter and shorter and the problem is worse… We could fix the code… that would be refreshing (Jackson W. , 2008)." The refreshing news is that software security is a

workable problem for those organizations that want a higher level of confidence in their systems. The term used to describe such faith in applications is software assurance. There are many subtle variations to the software assurance definition (Goertzel, et al., 2007), but the DoD defines software assurance as the "level of confidence that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software (Polydys & Wisseman, 2007)."

Some organizations have realized that the reactive antivirus and firewall (perimeter security) approach "doesn't stem the tide of incidents because the software it's building and buying doesn't resist attack (Steven, 2006)." In other words, those organizations have recognized that traditional security efforts that attempt to retroactively bolt on devices to make it more difficult to exploit defects simply aren't effective (Mead & McGraw, 2005). While the Internet and software may never be completely secure, targeting known problem areas in software development can "*stop the bleeding* (Taylor & McGraw, 2005)." The bottom line is that the only way we'll "cut through the myth of security by firewall" and fix the security problem is "by building better software (Epstein, Matsumoto, & McGraw, 2006)."

Within the U.S. Government, the 'proactive' software assurance effort is being led by the Strategic Initiatives Branch of the National Cyber Security Division of the Department of Homeland Security (DHS). The DHS project is called Build Security In (BSI) and support is provided by the Software Engineering Institute at Carnegie Mellon University and other software security experts. (DHS NCSD, 2006) The DoD also has a small software assurance office that works with the DHS BSI project.

The AF, in reaction to the compromise of 33,000 officer and enlisted personnel records in an attack on the Assignment Management System in 2005, tasked the 554 Electronic System Wing (554 ELSW) to initiate a pilot software assurance project (554 ELSW, 2007). In 2007, that pilot project grew into the Application Software Assurance Center of Excellence (ASACoE) managed by the 754th Electronic Systems Group at Maxwell AFB-Gunter Annex, AL, and a $10.2 million contract was awarded to Telos Corporation to provide software assurance tools and services. Telos' team includes other application security industry leaders, such as Cigital, Inc., Fortify Software, Inc., IBM/Watchfire Corporation and Application Security, Inc. (Telos, 2007) All of these organizations are looking for the best ways to *stop the bleeding*.

Research Objective/Questions

The purpose of this research is to identify common issues that present challenges to the development of secure software and possible solutions or best practices that the Air Force could adopt as it tackles the software assurance problem. My research and investigative questions are below.

Research Question

What strategies or practices are best suited for answering the software security challenges the AF is facing as it stands up a formal software assurance program?

Investigative Questions

1. What are the common issues (findings) that present challenges to the development of secure software for military systems?
2. What practices might best resolve the findings?

Research Focus and Scope

The primary focus of this research is on publicly available, unclassified articles, reports, and other documents that depict security related problems associated with producing and acquiring software. Additionally, data is being sought from multiple AF organizations that develop and acquire software to provide an initial look at how they may or may not be working towards increasing their software assurance. This research will not assess the people involved with the AF systems in these organizations, but will instead concentrate on the software-intensive systems developed within these units. The scope of this research involves only identification of software security challenges and best practices. The scope does not include testing the impacts of best practices on those challenges within the case study units. This research will not use Unclassified For Official Use Only (FOUO) information, classified information sources, or request the case study participants reveal such restricted information.

Methodology

This research will be conducted as a combination of a content analysis and case study. The content analysis will be the primary data source and will examine government reports and various other source materials. The case study data will be used to complement the content analysis research.

Assumptions/Limitations

The amount of literature written about software assurance, quality, and reliability is vast. In the time available for this research, it is not possible to explore it all and

consider everything that is identified as a problem area or best practice. This study is limited to those sources that are identified in the following chapters. Additionally, there are many organizations across the DoD that develop software-intensive systems. These systems vary widely in complexity, exploitability, connectivity, and mission criticality. This research focuses on very small fraction of those systems, which limits the generalizability of any conclusions. Software assurance is a broad topic that simply cannot be sufficiently covered in one thesis. However, this research was meant to form a starting point for follow-on research on software assurance within the AF.

Implications

The goal of this research is to understand where the software security problem is coming from and to identify an initial list of best practices for addressing the problems. Knowing this, the AF and DoD may be able to apply the best practices to its weapon and information technology (IT) programs. Thus, the AF and DoD will be better able to defend their systems during cyber war, and to also ensure the confidentiality, integrity, and availability of systems during any conflict that involves net-centric warfare.

Preview

This chapter introduced the software assurance problem and described the research objective, questions, and focus. Chapter II will identify several best practices for software assurance through a literature review. Chapter III will describe the methodology behind the content analysis and case study. Chapter IV will provide a synopsis of the data gathered using the data found during the content analysis portion of

the research and the case study questionnaire. Chapter V will summarize the results, and provide any conclusions or recommendations formed as a result of this research.

II. Literature Review

Chapter Overview

The purpose of this chapter is to review the literature that describes software assurance best practices that are espoused by industry, academia, and government organizations.

Sources

The Software Assurance Forum for Excellence in Code (SAFECode) is a non-profit organization founded by EMC Corporation, Juniper Networks, Inc., Microsoft Corporation, SAP AG and Symantec. SAFECode's mission is to "identify and promote best practices for developing and delivering more secure and reliable software, hardware, and services." SAFECode's members have come to realize that they have a responsibility and business incentive to address software assurance and have engaged in "significant efforts to reduce vulnerabilities, improve resistance to attack and protect the integrity of the products they sell." (SAFECode, 2008)

In addition to SAFECode members, numerous other software assurance experts, often from large software companies or companies specializing in secure software, have described best practices or lessons learned. The following sections identify the SAFECode members' best practices and those from the subject matter experts. Oftentimes they overlap each other.

Software Assurance Best Practices

Training and Education

SAFECode members believe that security training is a prerequisite for the development team. In-house, external, and/or online training should be provided covering a wide range of topics including threat modeling, role-based security engineering, avoiding unsafe library function calls and preventing cross-site scripting errors. (SAFECode, 2008)

John Steven (2006) writes that development and delivery of a training curriculum is another competency that organizations should pursue and Dan Taylor and Gary McGraw (2005) state that training and mentoring developers and architects is a necessity. In an article reporting on discussions that took place during a Center for Discrete Mathematics & Theoretical Computer Science (DIMACS) Software Security Workshop, McGraw (2003) declared that "for the most part, software architects, developers, and testers remain blithely unaware of the problem" and "one form of best practice involves training software development staff on critical software security issues." Furthermore, McGraw states the "most effective form of training begins with a description of the problem" that is then followed by a demonstration of its impact and importance. Michael Howard and Steve Lipner (2003) wrote that education must be the "first part of any security push" because not everyone understands how to build secure software. Howard and Lipner (2003) also wrote that once developers understand the system threats, "they can determine if the threats are appropriately mitigated." William Schneider, Jr. (2007) wrote that it is "assumed that all engineering staff will attend ongoing security training."

General security awareness training is not enough. Kenneth Van Wyk and John Steven (2006) wrote that training "must be customized to reflect the organization's platforms, technology paradigms, languages, and packages," otherwise developers might "complain that the guidance is 'too high-level.'" Van Wyk and Steven (2006), Howard and Lipner (2003), and McGraw (2003) all agree that specialized security training tracks should be provided that are useful and informative for each job role within the organization (i.e., designers, program managers, architects, developers, testers, quality assurance, and documentation).

Build Security In

Gary McGraw and John Viega (2000) summarize this best practice well:

> Retrofitting security is always the wrong solution. Instead, security must be designed into software from the beginning… In contrast to the ad hoc security development techniques, which do not tend to work very well, security should be considered during all phases of the development cycle, instead of bolted on as an afterthought… Security is not a basic feature that you can add to a system at any time. Security is like fault tolerance; it is a system-wide emergent property that requires significant and careful planning and design.

SAFECode members accomplish this through several different development methodologies (called Software Development Frameworks), but they all share common elements: Concept, Requirements, Design and Documentation, Programming, Testing, Integration and Internal Evaluation, Release, Maintenance, Sustaining Engineering and Incident Response. During each of these lifecycle stages, SAFECode's members adhere to well established controls. (SAFECode, 2008) Rigorous development methodologies for mitigating software security threats are also recommended by Cynthia Irvine and Karl Levitt (2007), Jeremy Epstein, Scott Matsumoto, and Gary McGraw (2006), and Thorsten

Schneider (2006). Goertzel, et al. (2007), lists and compares several security-enhanced software development methodologies that "have been used successfully in either multiple real-world development projects or multiple academic pilots." One that is detailed by Goertzel, et al., is the Microsoft Trustworthy Computing Security Development Lifecycle (SDL), shown in the following figure:

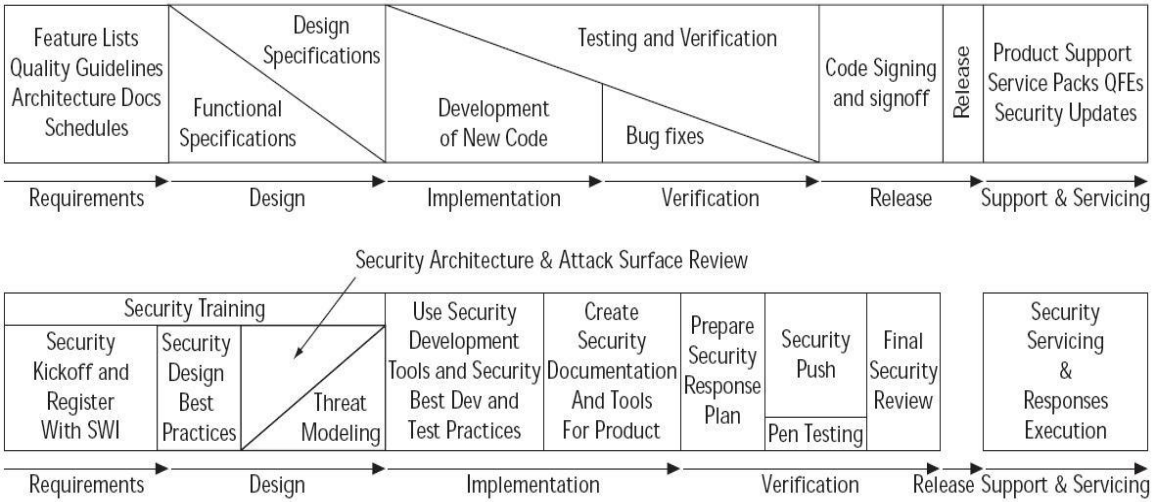

Figure 6. Microsoft Development Process with SDL Improvements (Schneider T. , 2006; Lipner & Howard, 2004; Lipner & Howard, 2005)

Security specific requirements must be defined at the beginning of the project in tandem with other requirements and checked periodically throughout the development cycle. Potential threats and risk reduction methods must be identified early, which is usually accomplished through the use of threat modeling techniques. (SAFECode, 2008) In 1998, Gary McGraw wrote that "software application code must be developed with security built in rather than patched on after development" and this philosophy "has been recognized as a sound policy within the safety-critical applications community and should likewise become a part of the development process for security-critical systems." And again in 1999, McGraw wrote that "finding and removing bugs in a software system

before its release is orders of magnitude cheaper and more effective that (sic) trying to fix systems after release" and "designing a system for security, and testing the system extensively before release, presents a much better alternative." Other authors have expressed similar thoughts and recommendations:

- Premkumar Devanbu and Stuart Stubblebine (2000) wrote that it is best to "refine requirements and design processes to bring an earlier focus on security issues."
- Viega and McGraw (2001) wrote that the fundamental technique for managing the software security risk is to "begin early, know your threats, design for security, and subject your design to thorough objective risk analyses and testing." They also wrote that it is "always better to design for security from scratch than to try to add security to an existing design."
- Paco Hope, Gary McGraw, and Annie Antón (2004) wrote that security "is not a feature, it can't be bolted on after other software features are codified" and that security must be "built in from the ground up, as a critical part of the design from the very beginning (requirements specification) and included in every subsequent development phase all the way through fielding a complete system."
- Nancy Mead and Gary McGraw (2005) wrote that "the core philosophy underlying [the build security in] approach is that security, like dependability and reliability, can't be added onto a system after the fact… Instead, security must be designed and built into a system from the ground up."
- Epstein, Matsumoto, and McGraw (2006) recommend a strong security involvement in all lifecycle stages.
- Security is often considered a quality concern, and in writing about software quality requirements (a type of nonfunctional requirement), Zayaraz and Thambidurai (2007) wrote that nonfunctional requirements "should be addressed as early as possible in the software life cycle and properly built into the software architecture before proceeding toward a detailed design."
- Schneider, W.J. (2007) wrote that design is "the critical first stage in building secure software."

Developers must use their secure coding skills to develop robust software that is "designed with attacks in mind" and "specifically designed to resist attack (Polydys, Ryan, & Ryan, 2006)." The software should also be inspected using manual analysis and/or automated analysis tools that identify potential defects (SAFECode, 2008). This type of methodology allows organizations to test and fix their software for security

"before the crackers get a chance to do so (McGraw & Viega, 2000)." Also, since the cost of fixing defects increases over time, it "makes sense to check new code promptly… by integrating the source analysis tool into the developer's desktop so that developers can run on-demand analysis, … [or] integrate scanning into the code check-in process (Chandra, Chess, & Steven, 2006)." Chandra, Chess, & Steven (2006) also wrote that source analysis tools could also be used at build time and at major milestones. Davis, et al. (2004) wrote that over time, static analysis "should become compulsory" in efforts to "find known kinds of coding defects" and that production processes must be "constantly monitored and root causes of defects determined and reduced." Schneider, W.J. (2007) wrote that regardless of the programming language used, "static analysis tools should be used to detect some classes of coding errors." He also wrote that the tools "should be run regularly and bugs triaged and fixed" and it "is also beneficial to use multiple tools."

In addition to static or source code analysis, additional robust testing methods such as vulnerability analysis, penetration testing, "fuzzing" or testing by varying external inputs should be employed (SAFECode, 2008). Epstein, Matsumoto, and McGraw (2006) also recommend penetration testing, automated vulnerability testing, manual or automated source code analysis, and third-party code reviews. Schneider, W.J. (2007) wrote that fuzzing "can be very effective at finding security bugs." James Lewis (2007) also describes corporate practices that include the use of automated tools, independent security review teams, and red-teaming and penetration attacks prior to product launch.

Security Readiness, Documentation, and Response

The product's security posture should be assessed and any risks posed by potential security gaps should be documented prior to release and checked against the security requirements set in the requirements phase. Security guides that provide explicit treatment of security issues, to include secure configuration of the product, should be a standard part of the product's documentation. (SAFECode, 2008) McGraw (1999) wrote that the risk analysis team "should not include anyone from the design and development team" in order to preserve "a completely independent view of the system, divorced from design influences." Robert Ellison (2005) wrote that "documenting the dependencies among the assembled components is an essential step in understanding the possible system failures and identifying components impacted by defects."

After the product is deployed, incidents may occur and there must be a mechanism in place where the incident information is communicated and relayed to the developers or sustainment team so that appropriate risk mitigation steps and/or software patches can be developed and released (SAFECode, 2008). Similarly, knowledge "gained by understanding attacks and exploits should be cycled back into the development organization, and security practitioners should explicitly track both threat models and attack patterns (McGraw, 2004)." Schneider, W.J. (2007) wrote that "vulnerabilities will continue to be discovered in fielded software for the indefinite future" and many COTS vendors "have recognized this fact and integrated a security response phase into their software development process." He also wrote that the security response process' role is "to accept reports of newly discovered vulnerabilities,

investigate them, and act to protect users of the vulnerable software." In commentary on incident recovery and reconstitution, Clint Kreitner (2008) lamented that he fears that "valuable learning from incidents is not occurring, or when it is, not being shared widely enough." As an example best practice, Kreitner identified the methodology by which airline industry incidents are investigated and how lessons learned are "fed back into aircraft design and airline operations to reduce the probability of recurrence of an incident from the same cause."

Code Pedigree/Provenance

Ownership and access history of objects is often been of paramount importance to the scientific, financial and artistic domains. Applying the same to software, code pedigree or provenance involves the origin of the source code, and who owned and accessed it. The provenance record involves the ownership entry and the log of the tasks applied to the code by authorized users. This is vital to ascertaining the software's trust level. (Hasan, Sion, & Winslett, 2007) The source code pedigree/provenance issue is sometimes labeled as "Software of Unknown Pedigree," "Software of Unknown Provenance," or simply, "SOUP." Gareth Rowlands (2003) defines SOUP as "software for which full and complete access to the source code, documentation, and/or development history is unavailable." Rowlands further wrote that most commercial components "are considered to be SOUP as, potentially, are software embedded in sub-systems and legacy software."

In light of SOUP, the handling of source code should be done carefully and securely and handled via strict change management controls that prevent cyber sabotage

or unauthorized persons from viewing or modifying the code. The change management systems must also be protected. (SAFECode, 2008) This sentiment is echoed by Irvine and Levitt (2007). Lewis (2007) wrote that most companies "audit any change made to code during development and track who made that change." Despite the protections implemented by some, source code repositories of commercial software development organizations have been compromised before (DoD IG, 2002). Polydys, Ryan, & Ryan (2006) wrote that change "must be managed to ensure that the software security functions are maintained at the appropriate level of acceptable risk."

Every link in the software supply chain deserves scrutiny. Beginning anew in 2002, attackers placed an emphasis on breaking into development and distribution points since it can be more efficient than locating and hacking individual systems. Traditional distribution of software via physical media offered some protection against tampering, however many people now get their software (and patches) from the Internet, in many shapes and forms. (Levy, 2003) Premkumar Devanbu, Philip Fong, and Stuart Stubblebine (1998) wrote that software "used to arrive in shrink-wrapped packages from known vendors" but increasingly, "software of unknown provenance arrives over the internet as applets or agents." If Internet distribution methods are used, the "download process itself must be secured with all appropriate methods to guarantee integrity and authenticity (Davis, et al., 2004)." Additionally, after product deployment it is important to provide a mechanism where customers or end-users can verify the integrity of the acquired software, i.e., that it is indeed from the trusted source. Code signing and

integrity checks built in to the software are two methods given by SAFECode. (SAFECode, 2008)

In 2006, the Association for Computing Machinery (ACM) Job Migration Task Force reported on outsourcing and offshoring development practices. Outsourcing software development refers to "having work for a company done by another organization" and offshoring refers to "having this work done in another country, whether or not it is done by part of the same company." The ACM task force wrote that existing and newly created risks are often poorly understood or addressed, and also magnified by offshoring. One of these risks is embedded malware, which is enhanced by offshore workers with less personal loyalty, hackers, organized crime syndicates, industrial espionage, unfriendly nations, and terrorists. Additionally, security departments rarely inspect outsourced or offshore code for trojans (malicious code disguised as legitimate software), viruses, or other types of malicious code. (ACM Job Migration Task Force, 2006)

A lack of understanding code provenance creates the possibility that "hostile nation or non-government hostile agents (terrorist/criminal) can compromise these systems." Businesses and nations should employ strategies to mitigate the risks of offshore software. These strategies may include vetting offshore providers carefully and not outsourcing work "without the explicit approval of the client." (ACM Job Migration Task Force, 2006) Outsourcing and offshoring are of particular concern, since within the DoD, "there are currently few explicit restrictions on the type of services work that can be sent offshore (Nilsen, 2005)."

Software is normally categorized as one of the following types of software:

- Internally developed software, which includes government-off-the-shelf (GOTS),
- Closed source software, which includes commercial-off-the-shelf (COTS), freeware, and shareware,
- Open source software (OSS),
- Legacy (reused) software,
- Outsourced software, which may include any of the above,
- Offshore software, which may include any of the above, and
- Mixed software (any combination of the above).

The use of third-party software (which is any of the above types except software that is strictly internal) also factors into code provenance and supply-chain issues. Just as organizations must ensure their own code is secure, organizations must "carefully consider the proper use of third-party software" and have "some means of identifying and categorizing the trust level" of the third-party components they use (Davis, et al., 2004). Davis, et al. (2004) further write that at a minimum, vendors should "disclose what third-party software, including open source software, is used in their product." Ideally, vendors should require third-parties to adopt secure processes and practices or vendors should "validate third-party software before incorporating it into their products (Davis, et al., 2004)." Otherwise, if "purchasers and administrators of a software product are not aware of the use of third-party software in the products they are using," "they may not know that they have a security issue if the producers of the third-party software issue a warning or patch (Davis, et al., 2004)."

Software modules have also been reused, reengineered, or integrated into other applications far beyond their original scope. There are numerous reasons for extending the life of legacy code, however if security was not a concern during the original

development, the integration of legacy software into today's networked environment presents many challenges. While it may be more difficult to integrate security into legacy code than to integrate it into new software development, quality and security principles can be applied to legacy projects if all project team members closely follow the guidelines of the chosen structured quality/security model. (Pruessner & Paternostro, 2006) Also, organizations must keep in mind that legacy code that is secure in one environment may be "completely insecure when placed in another" and "as the world becomes more interconnected via the Internet, the environment most machines find themselves in is at times less than friendly (McGraw, 1999)." The widespread reuse of unsafe library routines was cited by Sagar Chaki and Scott Hissame (2006) as one of the reasons why buffer overflows continue to be the cause of most software vulnerabilities despite increased awareness and other efforts. Schneider, W.J. (2007) wrote that at this point in time "almost all IT environments are composed of 'mixed code.'"

In addition to the normal software sources, organizations cannot ignore the provenance of any sample code that they provide to customers or use in their products. For example, Microsoft provides several software development kits that contain code samples. Customers can modify these samples as needed for their applications. Early in Microsoft's security push, they realized that insecure sample code can lead to insecure end-user applications. Microsoft required all sample code to undergo "intensive security review to provide customers with guidance on how to build safe and secure applications." (Howard & Lipner, 2003)

Research, Evangelism, and Management Involvement

SAFECode members conduct research and investigate the ever evolving threats, technologies, and mitigation techniques. This in turn generates knowledge that is provided to their developers. (SAFECode, 2008) Research has shown that "up-front attention to security can save the economy billions of dollars, yet security concerns are often treated as an afterthought to functional requirements (Mead, Hough, & Stehney, 2005)." Security evangelists need to continue this research and ensure their leaders become aware of it so that they may too become evangelists and put the research to use. Michael Howard and David LeBlanc (2003) wrote that organizations should have one or more people to evangelize the security cause and be the focal point for all security-related issues. Howard and LeBlanc (2003) further stated the evangelist's main goals would include (1) staying "abreast of security issues," (2) providing security education, (3) handing out awards for "the most secure code or the best fix of a security bug," and (4) providing "security bug triaging to determine the severity of security bugs" and "offer advice on how they should be fixed."

Leaders must promote software assurance through the sharing of best practices and findings in open forums, papers, articles, and books (SAFECode, 2008). Additionally, for any security initiative to be successful, management must clearly signal that security really should be taken seriously. Howard and LeBlanc (2003) wrote that once you have "succeeded in getting the attention of the boss, have him send an e-mail or memo to the appropriate team members explaining why security is a prime focus of the company." A well publicized example was Microsoft's Trustworthy Computing

Initiative that was kicked off via a 2002 memo signed by Bill Gates, and followed by a halt to Windows development so that the entire division of 8,500 engineers could participate in a security push that lasted for more than two months. (Chess & West, 2007; Howard & Lipner, 2003) Taylor and McGraw (2005) wrote that buy-in from management and tactical technical staff is required, otherwise improvement programs will fail. Davis, et al. (2004), wrote that top management "must have a sustained and focused priority of producing secure software" and "adequate resources must be available, outside expertise must be there when needed, and a quality culture must be sustained." Polydys, Ryan, & Ryan (2006) wrote that software assurance "must be a primary concern of all acquisition managers, be they program or project managers, buyers, integrators or specifiers of system requirements."

Many computer security experts agree that there is intense market pressure on software developers to write their code as fast as they can, make it "work," then release it for sale. Security is often "an afterthought at best, and is often forgotten completely (McGraw & Viega, 2000)." Since key software project goals often include functionality, usability, efficiency, time-to-market, and simplicity (Viega & McGraw, 2001), advocates and champions must engage upper management to ensure they also voice their support for quality and security goals (Yeakley & Fiebrich, 2006). Development managers often have to make trade-offs between schedule, cost, quality (including security), and functionality, and if there is not strong management support or the security assurance activities are not mature, the development manager may buckle under the cost and schedule pressures which leads to insecure software (van Wyk & Steven, 2006).

Although the level of assurance may not ever reach 100 percent, the facts, risks, and costs need to be brought forward by the evangelists to help management make objective quantitative trade-offs that involve security.

Other Software Assurance Concerns

Malicious Insiders and Employee Quality

While malicious individuals may work for foreign suppliers, organizations must also consider the risk posed by rogue or malicious employees within their own ranks. When an UBS PaineWebber insider planted a logic bomb and brought down nearly 2,000 servers in 2004, the incident showed that malware can "cause serious issues and can occur no matter where the code is written, both inside the US and outside and within one's own company and outside of it (Gengler, 2006)." To address this risk, SAFECode members focus more on "how [software] was made" versus "where [developers] were sitting" by implementing processes for "vetting employees and contractors regardless of their country of residence" and checking and verifying "software assurance irrespective of where it was produced (SAFECode, 2008)." Lewis (2007) wrote that companies "augment audit trails with authorization software that limits the ability to change or add code to those who have been granted access, so that programmers cannot freely access areas where they are not assigned work" and that this "greatly reduces the risk that a rogue coder will surreptitiously slip malicious code into a product."

As mentioned earlier, training and educating employees is a key best practice for software assurance, especially since "most security courses at universities emphasize network security" and "very little attention is paid to building secure software (Viega &

McGraw, 2001)." In order to raise the employee quality bar, several initiatives are underway to improve secure programming skills and knowledge. These initiatives include major corporations working with universities to create secure programming courses. As programmers become more experienced in secure development practices, organizations should also consider testing current employees or even testing prospective employees' training and education levels before hiring. To this end, industry, government, and academia have teamed with the SANS Institute to create the Secure Programming Skills Assessment (SPSA). Feedback on the six exams that are tailored to various programming languages has been very positive, and Steve Christey, editor of the CVE program, said that the exams "will help everyone draw the line in the sand, to say 'No more,' and to set minimum expectations for the everyday developer." (SANS SSI, 2007) Finally, Epstein, Matsumoto, and McGraw (2006) recommended that software vendors look for developers trained in software security as a measure to increase the robustness of their products against security failures.

Standardized Assessments of Supplier Security Controls

In 2006, BITS member executives (from financial institutions) formed the Financial Institution Shared Assessments Program, which now counts over 50 major US financial institutions in its membership. This initiative allows members to use standardized questionnaires to assess the security controls of their IT service providers and partners. (BITS, 2007) Peter Buxbaum (2008) wrote that the military should consider this model for assessing offshore software developers. Several sources include language that could be used to form such an assessment. The NDIA Systems Assurance

Guidebook (Bullard, et al., 2007) includes an example checklist of questions targeting system integrators supporting the government, the DoD and DHS Software Acquisition Working Group's Draft Software Assurance Acquisition Guide (Polydys & Wisseman, 2007) includes sample software assurance requirements for work statements, and the SAFECode Best Practice whitepaper (2008) includes a list of questions that organizations can pose to "determine the assurance and security of a proposed product procurement or vendor engagement." Polydys, Ryan, & Ryan (2006) wrote that it "is important that Instructions to Suppliers are clear and unambiguous regarding what they are to submit in their proposals" and the "acquisition manager should not only ask for documentation that provides evidence of potential suppliers' ability to delivery (sic) on software assurance requirements but also provides evidence concerning foreign ownership, control or influence."

INPUT (2005) wrote that the "inability to answer or perform certain practices or processes by the supplier would represent risks to the acquiring organization, which could then be factored into source selection decisions." The Defense Science Board (DSB) Task Force on Defensive Information Operations (2001) wrote that it is "imperative that the DoD becomes a smart buyer of commercial information and information assurance technology and services" and that it is also "important to assess suppliers' conformance with applicable standards." One way the DSB suggested qualifying suppliers was to "guage their commitment to fixing security-related flaws found in their systems." Schneider, W.J. (2007) wrote that "supplier trustworthiness should consider adversarial control and influence of the business or engineering

processes of the supplier, as well as the ability of the business and engineering processes to prevent outside penetration."

Chapter Summary

This chapter provided a broad overview of software assurance best practices as identified by industry, academic, and government sources. The three most discussed best practices were training and education, build security in, and code pedigree/provenance.

III. Methodology

Chapter Overview

The purpose of this chapter is to describe the research methodology that was selected and why. This chapter will also explain how I will implement the methodology in order to investigate the findings and recommendations from past examinations (audits) of software security within DoD and AF programs. Additionally, this chapter will describe the methods that I will use to examine select programs to determine if they (1) are or are not facing the same challenges as identified in the past examinations, and (2) could potentially benefit from applying the software assurance best practices.

Qualitative Approach

Qualitative research involves the collecting numerous forms of data and examining them "from various angles to construct a rich and meaningful picture of a complex, multifaceted situation." As such, qualitative research encompasses several approaches and they all have two things in common: (1) they focus on "phenomena that occur in natural settings – that is, in the 'real world,'" and (2) they involve studying "those phenomena in all their complexity." Qualitative research examines the phenomena's many dimensions and layers and tries to "portray the issue in its multifaceted form." (Leedy & Ormrod, 2005)

Additionally, qualitative studies can help define what is important or what needs to be studied when "little information exists on a topic, when variables are unknown, when a relevant theory base is inadequate or missing." Qualitative research can serve

many purposes, two of which are descriptive and interpretative. Descriptive research "can reveal the nature of certain situations, settings, processes, relationships, systems, or people." Interpretative research can "enable a researcher to (a) gain new insights about a particular phenomenon, (b) develop new concepts or theoretical perspectives about the phenomenon, and/or (c) discover the problems that exist within the phenomenon." Qualitative studies may also evolve over time as more is learned during the investigation and there are "no magic formulas, no cookbook recipes for conducting a qualitative study." (Leedy & Ormrod, 2005)

As shown in Chapter II of this thesis, the development of secure software can be quite complex, and involves many varied types of people, processes, and technologies. Little information exists on military software assurance practices and this research is an attempt to examine the real-world and complex nature of secure software development's many dimensions and layers. This research serves the descriptive purpose in that it seeks to reveal the nature of best practices and software development processes, systems, and people. This research also serves the interpretative purpose in that it seeks to gain new insights and discover problems that exist within military software development practices. It is for these reasons that a qualitative approach was deemed most appropriate. Since this thesis research has evolved and is actually a deviation from the researcher's original plans, select attributes of multiple qualitative methods will be drawn upon to conduct this research. Finally, the mixed qualitative methods approach was also selected since no "magic research formula or cookbook research recipe" was found for mapping past audit findings and recommendations to best practices.

Qualitative studies, as a general rule, "do not allow the researcher to identify cause-effect relationships… quantitative research, especially experimental studies, [is needed] to answer questions of [that] kind (Leedy & Ormrod, 2005)." It is very important to note that this research is not intended to identify a cause-effect relationship between software assurance best practices and the actual security performance (software assurance metrics) of AF systems. This research is only intended to identify past software security issues and which best practices *might* be applied to mitigate those risks and improve system software assurance. This research therefore may be used as a basis for a future quantitative study of whether or not the software assurance best practices actually have an effect on software security within specific AF systems.

Best Practices Research Method

Volumes too numerous to count have been written about the best practices for conducting research, however very little apparently has been written about researching best practices. Neither of the two research methods textbooks used in recent courses at this institution even mentions best practice research. An Internet search on the topic identified only a few sources that described best practice research.

James Mold and Mark Gregory (2003), in researching best practices within the medical community, wrote that "the tasks involved in primary care are complex and varied", "thousands of extremely bright people struggle on a daily basis with the same kinds of practice challenges, come up with a variety of solutions, and rarely share them with anyone," and this "collective wisdom" is an "immense untapped reservoir of practical information that could, if properly evaluated, described, and disseminated,

improve the quality and efficiency of primary care services throughout the country." Mold and Gregory define best practice research as "a systematic process used to identify, describe, combine, and disseminate effective and efficient clinical and/or management strategies developed and refined by practicing clinicians." They furthermore describe five steps in conducting best practice research: "development of a conceptual model or series of steps, definition of "best" based on values and standards, identification and evaluation of potentially effective methods for each component or step, combination of most-effective methods, and testing of combined methods." Mold and Gregory also identify some limitations and disadvantages:

- Some processes cannot be so easily broken down into steps or components,
- No one in a particular network or group may have figured out how to effectively accomplish a particular task,
- Best methods for individual steps may be practice specific or may not fit nicely together into a combined best practice method,
- Combined best practice solutions may not be applicable to all practices, and
- Since the research is not theory driven, solutions tend to be issue specific.

Ophelia Eglene (2000) wrote that "people and organizations all over the world are looking for more effective, less expensive, innovative ways to get work done," and that research into current practice is "an organized attempt to learn from the experience of others," and best practices involve taking current practices another step by separating mistakes from successes you'd like to emulate. In other words, best practice research involves looking "deeper into the characteristics that led to success." Eglene also wrote that there "is no one best way to conduct the research; it is more of a question of finding the method that works best for you and your research area." However, Eglene described

three basic steps: "formulation of the question, gathering preliminary information, and conducting in-depth interviews."

Similarly to primary care, the tasks involved in secure software development are complex and varied. Both of the best practice research descriptions above can be applied to this qualitative study of software assurance best practices, but like Eglene said, there is no one best way. Therefore, select portions of each method were chosen to form a best practice research method that works best for this particular researcher and research area. Chapter II of this thesis maps to Mold and Gregory's first and third steps (development of a conceptual model and identification of potential methods, respectively). Mold and Gregory's second step was not used, but their fourth step (combining best components) is similar to content analysis methods and will be discussed in the content analysis section below. Mold and Gregory's fifth and final step (test combined method) is outside the scope of this research study, and would be best suited for a future quantitative study as suggested earlier in this chapter. Chapter I and II of this thesis essentially constitute the first two steps (respectively) in Eglene's best practice research methodology. Eglene's third step, in-depth interviews, is similar to case study methods and is discussed in the research design section later in this chapter.

Content Analysis Method

A content analysis is "a detailed and systematic examination of the contents of a particular body of material for the purpose of identifying patterns, themes, or biases." Content analyses "are quite systematic, and measures are taken to make the process as objective as possible," with the crucial step being to "tabulate the frequency of each

characteristic found in the material being studied." The tabulations are used by the researcher to interpret the data "as they reflect on the problem under investigation." (Leedy & Ormrod, 2005)

Case Study Method

A case study is when a "particular individual, program, or event is studied in depth for a defined period of time." Sometimes a single case is focused on, but in other instances two or more cases, referred to as a multiple or collective case study, are the focus. When a single case is used, generalizations made are tentative, and must "await further support from other studies – perhaps support from additional case studies, other kinds of qualitative studies, or experimental research." Case studies are "especially suitable for learning more about a little known or poorly understood situation." (Leedy & Ormrod, 2005) Robert Yin (2003) wrote that case studies are "used in many situations to contribute to our knowledge of individual, group, organizational, social, political, and related phenomena" and the preferred strategy when "the focus is on a contemporary phenomenon within some real-life context." Yin further wrote that exploratory case studies are appropriate for research questions that focus on "what" questions, as this thesis does.

Qualitative Approach Summary

Multiple forms of data and research methodologies are used in qualitative studies, and many are also characterized by emerging designs whereby early data collection often influences the kinds of data subsequently gathered (Leedy & Ormrod, 2005). In my research, I will employ multiple methodologies and gather multiple forms of data. This

approach also benefits the concept of triangulation within qualitative research, which is the "combining of methods, data sources, and other factors in examining what is under study" in an effort to strengthen data collection and analysis (Caudle, 1994).

Research Design

A research design is the "logical sequence that connects the empirical data to a study's initial research questions and, ultimately to its conclusions." (Yin, 2003) For the content analysis research design, Leedy and Ormrod (2005) identify three essential items: (1) a description of the body of material studied; (2) precise definitions and descriptions of the characteristics sought; and (3) the coding or rating procedure. All three of these items are discussed in this section. Leedy and Ormrod also identify two other items (tabulations for each characteristic and a description of patterns that the data reflect), but those correlate to Chapter IV and V of this thesis, respectively.

Yin (2003) identifies five important components of a cast study research design: (1) a study's questions; (2) its propositions, if any; (3) its unit(s) of analysis; (4) the logic linking the data to the propositions; and (5) the criteria for interpreting the findings. Chapter I and II of this thesis match Yin's research design components 1 and 2, respectively. Yin's unit of analysis component is analogous to Leedy and Ormrod's description of the body of material to be studied. Yin's remaining components (3, 4, and 5) are discussed in this section.

Unit of Analysis and Body of Material

The unit of analysis involves defining what the "case" is, and relates to the way the research question is defined (Yin, 2003). This thesis' research and investigative

questions seek to examine software security challenges facing the military, in particular the common findings from past audits of systems. Prior to the inclusion of net-centric warfare as part of the DoD's force transformation effort from late 2001 to 2003 (DoD, 2004), several government reports documented various issues related to software assurance in the DoD and made recommendations on how to address these issues. One would think that the DoD would include software assurance as part of its focus on transformation or at least work hard at adopting some of the recommendations made in the reports, especially since high quality shared awareness, a key component of net-centric warfare, depends heavily on "secure and assured networks and information that can be defended (DoD, 2005)." However, in the years since the transformation effort and focus on net-centric warfare began in earnest, several more government reports have shown that many of these problems still exist and are not being addressed.

Therefore, for this thesis, the primary body of materials will encompass past audits of software-intensive military systems, complimented by secondary units of analysis (case studies) that include participants from selected organizations that were accessible to the researcher and willing to cooperate with this study. The audits were found by searching online document repositories, including the Defense Technical Information Center (DTIC) Public Scientific and Technical Information Network (STINET),[2] the Defense Science Board webpage,[3] the DoD Inspector General website,[4] the AF Audit Agency's military restricted website,[5] and the Government Accountability

[2] located at http://stinet.dtic.mil/str/guided-tr.html
[3] located at http://www.acq.osd.mil/dsb/reports.htm
[4] located at http://www.dodig.osd.mil/PUBS/index.html
[5] Located at https://www.afaa.hq.af.mil/afck/plansreports/reports.shtml

Office website.[6] Various search terms, including "software security," "software assurance," "provenance," "pedigree," and "code analysis" were used. In some cases, the documents themselves provided clues in their text or bibliography/references section as to other documents that may apply to this research. The content analysis body of material was also restricted to documents that were: (1) about a specific military system or systems, and (2) published from 1999 to February 2008. The date restriction was placed on the body of materials since that is just before the DoD started focusing on net centric warfare and transformation, and also because the early 2000s is when the offshoring of software phenomenon "took off (ACM Job Migration Task Force, 2006)." A list of the documents included in the content analysis is provided in Appendix A.

Definitions and Descriptions of Content Analysis Characteristics

Leedy and Ormrod (2005) wrote that researchers must define "each characteristic precisely enough that another researcher could replicate [the] study." Sharon Caudle (1994) wrote that this step involves unitizing ("identifying the smallest piece of information about something that can stand by itself") and categorizing (coding the "units or chunks together in some type of relationship"). For this content analysis effort, one unit is defined as a phrase (sentence fragment), sentence, or paragraph. These units will be categorized as either software security related findings or not. For example, if one of the documents contains a sentence like "It was determined that software security policies do not fully address the risk of using foreign suppliers to develop weapon system software", the unit would be "software security policies do not fully address the risk of

[6] located at http://www.gao.gov/

using foreign suppliers" and it would be categorized as a software security related finding. Units that are not software security related findings will not be tallied, however they may be examined to gain a better understanding of what the finding relates to. Each unit/finding will only be counted once; they will not be weighted by the number of systems fielded, number of affected clients or servers, or number of affected users since this information may not consistently provided in each document.

Examples of units that would not be considered software security findings are: (1) insufficient software licenses, (2) lack of an alternate site within the Continuity of Operations Plan (COOP), (3) insufficient or deficient authorized access forms for users, and (4) human failures to use software security features or controls (such as failure to update the antivirus software or failure to delete old accounts). Although closely related to the COOP issue above, a failure to have or test a contingency plan would be a software security related finding (and therefore associated with the "ensure post-deployment incident response measures are in-place" best practice). Also, existence of unauthorized software on the system would be a software security related finding since the system apparently did not have a feature or control designed to prevent unauthorized software from being installed ("employ mechanisms to mitigate threat/impact of malicious insiders" best practice).

Content Analysis Coding Procedure

The following steps will be followed to identify the findings and recommendations within the documents:

1. As each document is read, the rater will highlight, underline, or circle any units that he/she feels is a finding or recommendation.

2. Next the rater will reexamine the units and ensure that units identified in the abstract or executive summary are not duplicated (counted more than once) with the same finding in the main body of the document.
3. Next, the rater will compare the findings to the items listed in the 'best practice' table as shown in Appendix D and decide which best practice would best resolve the finding. If the rater feels none are applicable, spaces labeled as "other:" are provided for the rater to fill in. The best practices provided in the table are those that were identified in Chapter II of this thesis. The rater's decision is to be noted in the margin next to the finding.
4. Finally, after completing the document review, the rater will go back through the document and tally the results and fill out the 'best practice' table as shown in Appendix D.

For example, the previously mentioned example finding "software security policies do not fully address the risk of using foreign suppliers," would probably best be resolved by a new Evangelism or Management Involvement best practice such as "Ensure organizational software assurance policies are detailed, current, and followed."

I will code most, if not all of the documents included in the unit of analysis. Additional raters may also be used to review and validate my coding decisions. If additional raters are used, prior to them reviewing any documents, I will provide them with an overview of software assurance, the best practices identified in Chapter II, and training in these coding procedures.

Case Study Survey Design

One of the documents included in the content analysis unit of analysis was the *United States General Accounting Office (GAO) Report to Congressional Requesters on Defense Acquisitions – Knowledge of Software Suppliers Needed to Manage Risks*, published in May 2004. This report was based upon in-depth interviews conducted by the GAO. The GAO was contacted and provided a copy of all the questions that were

asked during the interviews. Two lists of questions were obtained: (1) questions for the DoD program managers, and (2) questions for the contractors working on the DoD program. Appendix B contains a copy of the GAO's original questions.

In 2006, the CIO Executive Council published results of a September 2006 poll that showed Chief Information Officers (CIOs) "lacked confidence in software's ability to function as intended and be free of flaws, security vulnerabilities and malicious code (Fogerty, 2006)." Along with the poll results, the questions used in the poll were also published. This poll clearly highlighted software assurance concerns and thus was deemed potentially relevant to this study.

The GAO report's list of questions for the program managers and the CIO Executive Council's list of poll questions were combined to create a list of interview/survey questions for military weapon or IT program managers. In an effort to potentially expand the software assurance knowledge base, I created some new questions for the survey based on common software assurance themes or problems noted by authors when I was conducting the literature review. For example, Carol Woody (2005) noted that there is "never sufficient time and resources to address all project requirements, and organizational importance factors heavily in the selection process." Therefore, I added a question to the survey that inquired if the respondent felt their organization provided adequate resources to address the system's software assurance problem. Other questions were similarly added. It is hoped that the responses to these questions might provide a more thorough understanding of the software assurance practices within the case study units. The table in Appendix E contains all of the questions, and the left-hand column

identifies if the question came from the GAO report, CIO Executive Council survey, or is one that I added.

Pilot Case Study

A point of contact (POC) within a unit local to Wright-Patterson AFB was contacted and expressed interest in participating in the case study research. During a meeting with the POC, the survey questions were reviewed and modified based upon various factors. For example, some of the original GAO questions were removed because any response given would most likely be classified, which was outside the scope of this project. Also, during the meeting, in an effort to improve the response rate, the survey was split into three new lists of questions, each targeted at a specific role typically found within military weapon or IT system program staffs.

The roles were program manager, system or software engineer, and program security manager. With the questions split by role, for example, the program manager would not need to waste time tracking down information that only the security manager might know or be able to provide. Some of the questions are duplicated on more than one list. This was done as a check to see if the responses within each program office would be the same or similar between the different roles. The final list of questions for each role is in Appendix C. The method was also changed from in-person interviews to surveys distributed and collected via e-mail due to access restrictions (including a desire to keep the respondents anonymous) and potential scheduling conflicts with the respondents.

Unfortunately, a short while after the surveys were distributed within the case unit, initial feedback from respondents indicated that several of the questions still might involve classified or highly sensitive subject areas. As mentioned earlier, classified information was outside the research scope and if the case unit's participation was pursued further, the responses would have had to go through a lengthy approval process that would most likely exceed the time allotted for this research. Therefore that unit's participation was halted, and four other organizations with less sensitive systems were contacted. Three of the four new case study units are not located at Wright-Patterson, so due to the geographic separation of the respondents and the time restrictions for conducting this research, the decision to not conduct in-person interviews was maintained.

Conducting the Case Study

In line with AF and institutional guidance and policy, an exemption to Human Subjects Research (HSR) was gained prior to distribution of the surveys, including distribution of the survey to the pilot case unit. Additionally, unit commander approval to conduct the surveys was sought. POCs within each organization were also requested. The same surveys as the original case were distributed to the POCs (via e-mail), who then distributed the surveys within the organization. The POCs collected the completed surveys and ensured all personally identifiable information was removed from the responses prior to them being returned to me. This included removing any information contained within the e-mail headers. This is referred to as the "honest broker" technique.

Logical Linking of Survey Questions to Software Assurance Best Practices

Many of the questions on the survey are open ended or have the potential for eliciting a wide range of responses. However, in most cases the questions are written such that potential responses will likely logically link to a particular best practice. Therefore, I "mapped" each question (consolidated from the three surveys) to primary and/or secondary best practices that I anticipated responses would most likely relate to. For example, the responses to the questions that refer to foreign suppliers may provide a clue as to whether or not the respondent's organization needs to improve their mitigation strategy as it applies to foreign suppliers and offshore code. So the foreign supplier questions were mapped to the best practice of protecting against threats posed by offshore code. The mapping for each question is provided in a table in Appendix E.

Criteria for Interpreting Survey Responses

Although it is an important component, Yin (2003) wrote that there may be no precise way of setting the criteria for interpreting these types of findings [the survey responses]. Some of the responses, especially the ones based upon the CIO Executive Council survey, will be easy to compare and contrast, however some – most notably the open ended questions – may not. This is the point where I will need to go "beyond the facts themselves to [my] interpretation of the facts (Leedy & Ormrod, 2005)."

Research Design Quality

Some believe that a qualitative researcher's ability "to interpret and make sense of what he or she sees is critical for understanding any social phenomenon" and the researcher "is an instrument." Additionally, if the researcher has a "firm grasp of

previous research related to the problem," then he or she will be better suited to know "what to look for and can separate important information from unimportant details in what he or she observes" and "the researcher must be adept at wading through huge amounts of data and finding a meaningful order in what, to someone else, may seem like chaos." (Leedy & Ormrod, 2005) With respect to my abilities, I am in a unique position for this research. Over the course of my many assignments as a Communications and Information Officer in the AF, I have served in various positions which included being a software developer, project manager, and information assurance (security) systems certifier. Those positions have provided me invaluable experiences that will assist in interpreting and making sense of the data collected.

However, one item of concern in qualitative research is the researcher's ability to be objective and be "influenced as little as possible by any perceptions, impressions, and biases they may have." (Leedy & Ormrod, 2005) This concern is a valid one since it is possible that I may know or have served with some of the respondents and any personal biases (positive or negative) could potentially exist as I examined the data. This concern is mitigated through the use of the "honest broker" technique – the respondent identities will be masked, which will prevent any personal biases from impacting the data analysis.

Yin (2003) wrote that any given design can be judged according to certain logical tests, which commonly are construct validity, internal validity, external validity, and reliability.

Construct Validity

Construct validity is "establishing correct operational measures for the concepts being studied. (Yin, 2003)" As noted by Yin, common tactics include using multiple sources of evidence, establishing a chain of evidence, and having key informants review the draft case study report. All three of these techniques are in use for this research.

Internal Validity

Internal validity is "establishing a causal relationship, whereby certain conditions are shown to lead to other conditions (Yin, 2003)." Since this thesis research is descriptive and exploratory (not causal in nature), the internal validity test is not applicable.

External Validity

External validity is "establishing the domain to which a study's findings can be generalized (Yin, 2003)." Yin's recommended tactic for external validity is to use replication logic in multiple case studies. This thesis research is attempting to satisfy external validity concerns by incorporating case studies of multiple programs within four distinct AF units.

Reliability

Reliability is "demonstrating that the operations of a study – such as the data collection procedures – can be repeated, with the same results (Yin, 2003)." This chapter has been written with this test in mind, and the details of the research design have been thoroughly documented.

Research Limitations

The primary limitation on this research is time. There's always more that can be done, always more documents to find and analyze. Additionally, given more time, additional case units could have been approached to see if they would participate in the survey. There are some additional limitations to the approach taken on these case studies:

1. The only participants are units with less sensitive IT systems, which may show a lower software assurance posture than expected since the less sensitive programs likely have less money in their budget to spend on software security than weapon programs of higher criticality. The research would be more robust if it had been feasible to include weapon programs that managed more mission critical or more sensitive systems
2. The preferred implementation of the case study was to conduct in-person interviews; however that would have been costly since the case study units were located at four different bases across the United States. In-person interviews would most likely provide richer data.

Chapter Summary

This chapter described the qualitative research methods, why they were chosen, and the specifics of how each will be carried out.

IV. Analysis and Results

Chapter Overview

This chapter provides the results from the data collection and analysis.

Qualitative Data

The following sections discuss the results of the content analysis and case study.

Content Analysis Results and Discussion

Detailed data are provided in Appendix F. The following charts and discussions summarize how the number of findings related to the software assurance best practices.

Figure 7. Results for Training and Education Best Practices

The findings related to the Training and Education Best Practices primarily were targeted at users and system administrators (other category). The audits simply did not attribute the security related findings to members of the system's development team.

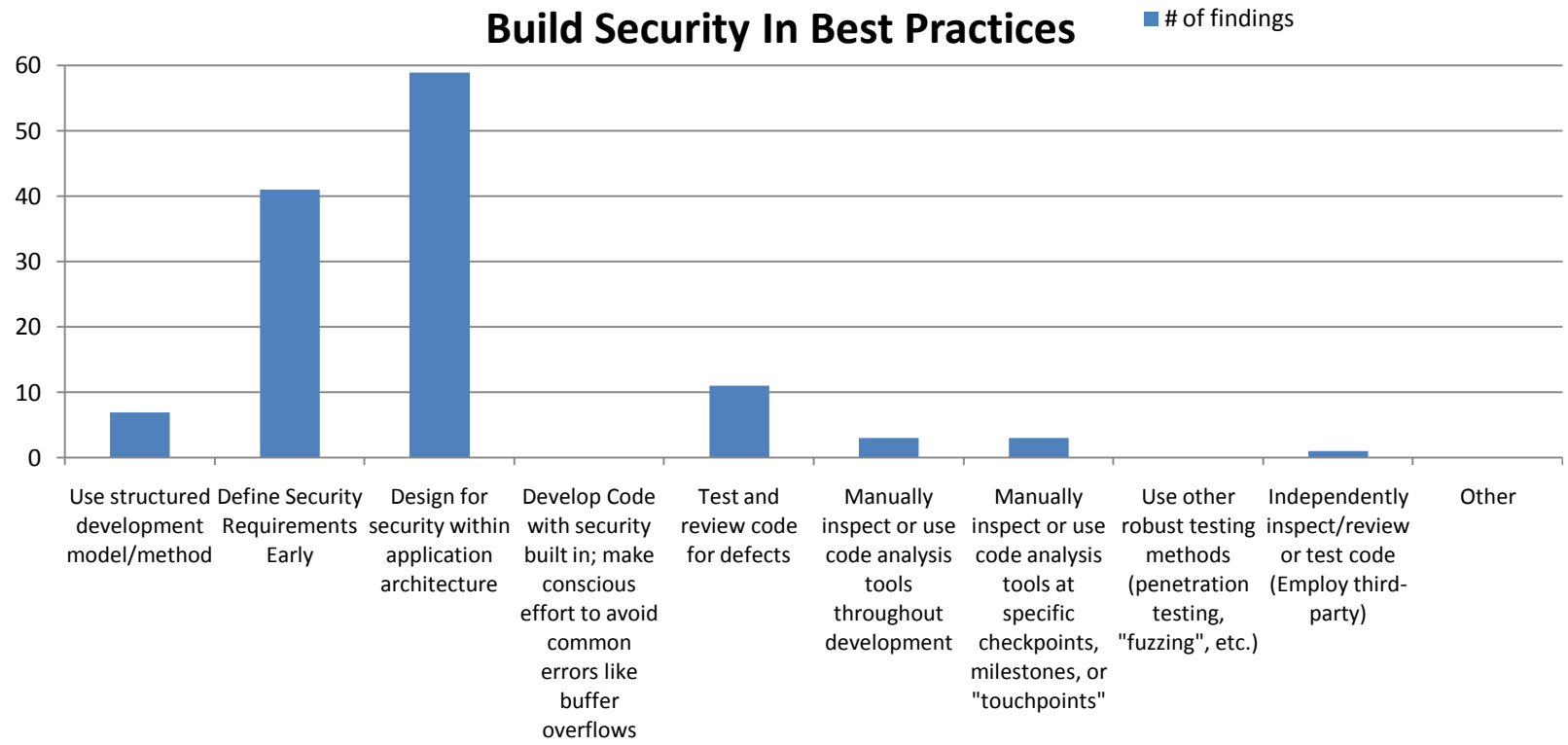

Figure 8. Results for Build Security In (BSI) Best Practices

The findings related to the Build Security In Best Practices were many, and primarily focused on failures to consider security in the requirements and design phases of system development. Specifically, most of the findings with system design applied to the use of generic accounts, insecure default settings, and userid/password logon routines that were non-compliant with policies. Code testing and structured development methods would have also resolved some of the findings.

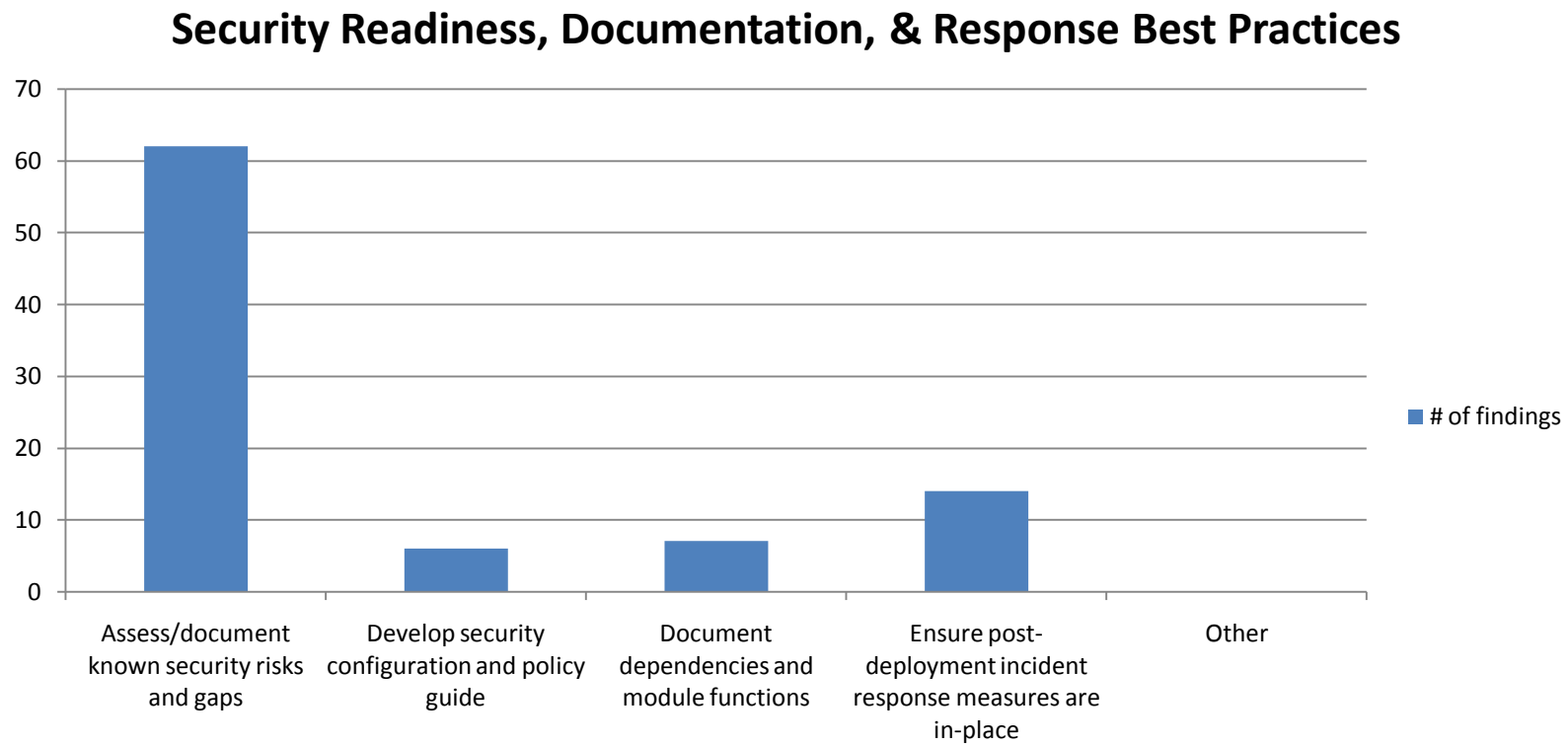

Figure 9. Results for Security Readiness, Documentation, and Response Best Practices

The findings related to Security Readiness, Documentation, and Response Best Practices primarily were focused on failures to conduct a risk analysis of the system. This usually was mentioned in the context of completing the certification and accreditation process. As shown in the results, failures to plan for incident response often went along with the certification and accreditation audits.

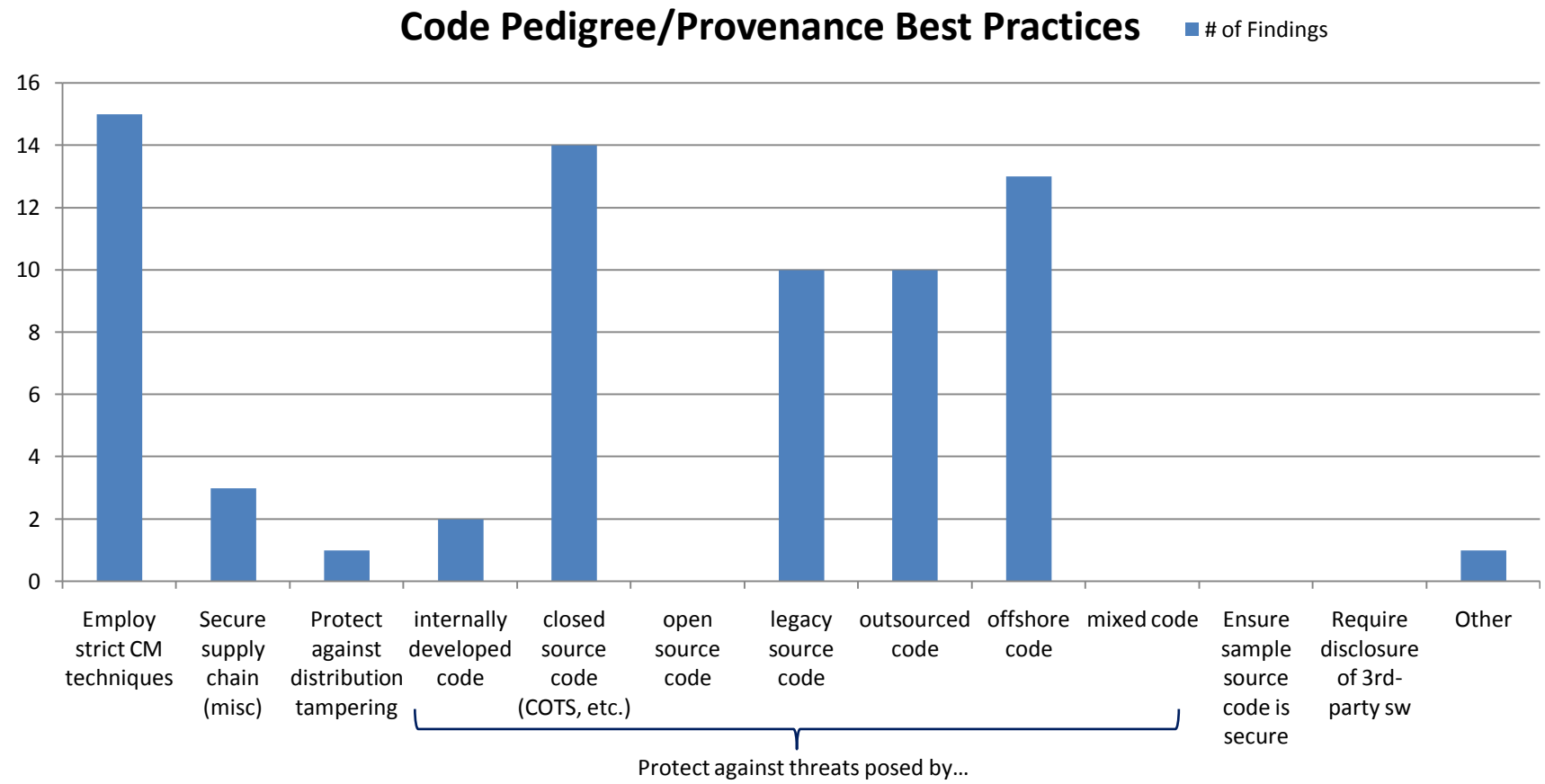

Figure 10. Results for Code Pedigree/Provenance Best Practices

The findings related to Code Pedigree/Provenance Best Practices focused on several areas, most notably change management and threats posed by commercial (COTS), legacy, outsourced (contracted), and offshore (foreign) software.

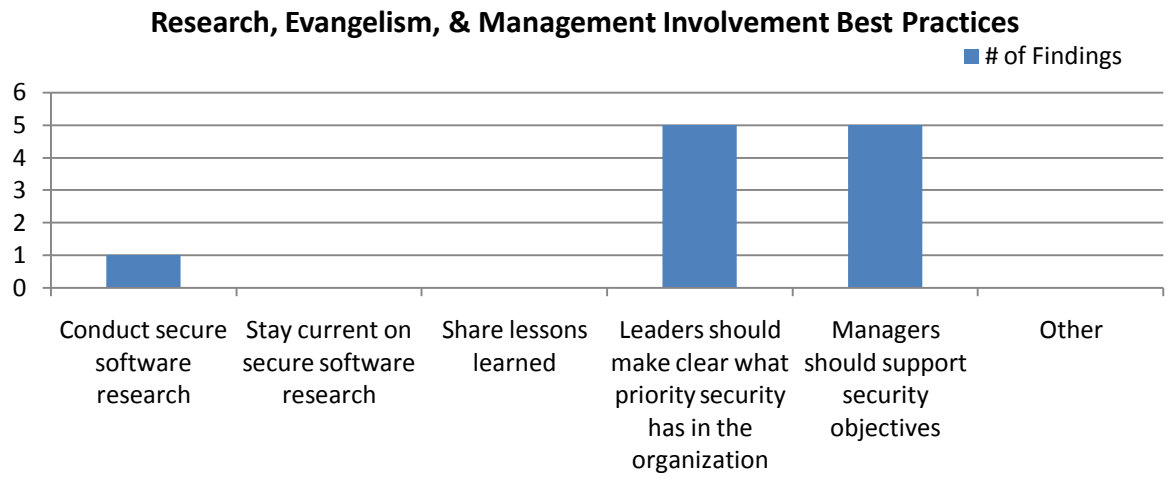

Figure 11. Results for Research, Evangelism, and Management Best Practices

The findings related to Research, Evangelism, and Management Involvement Best Practices were few, but those that were noted mainly targeted the organization's leadership and managers.

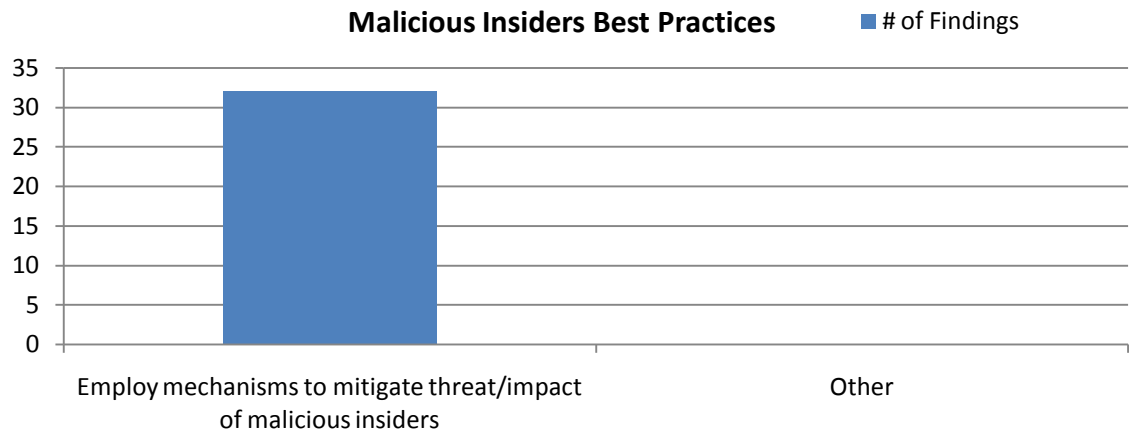

Figure 12. Results for Malicious Insiders Best Practices

The findings related to Malicious Insiders Best Practices appear numerous, but most of them involved failures to prevent users from installing unauthorized software and failures to segregate duties associated with system administration and management. The

large number of unauthorized software findings is most likely being mitigated by the AF's ongoing efforts to standardize the entire AF enterprise network and implement the Standard Desktop Configuration.

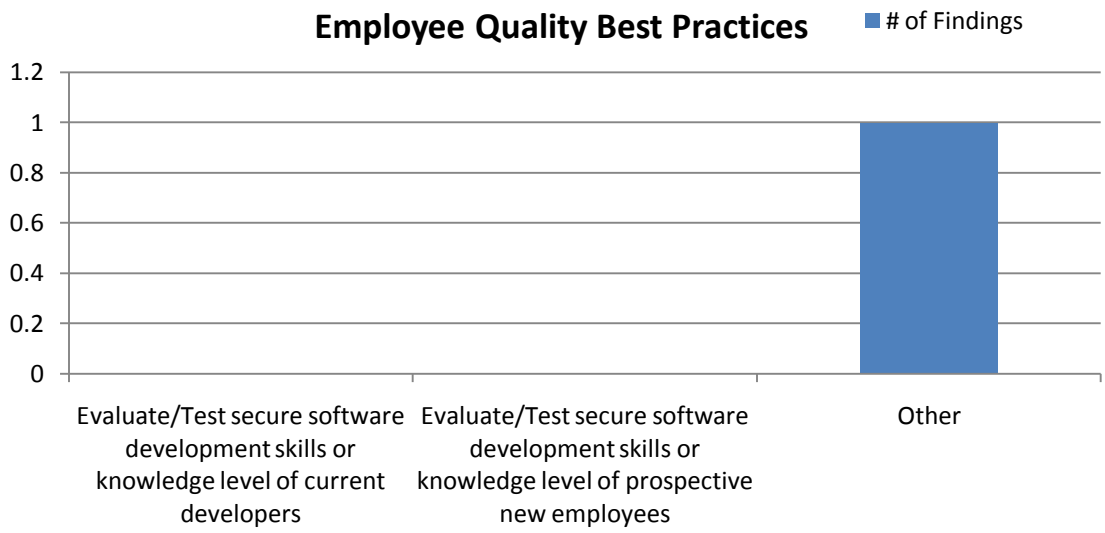

Figure 13. Results for Employee Quality Best Practices

Only one finding was related to Employee Quality Best Practices.

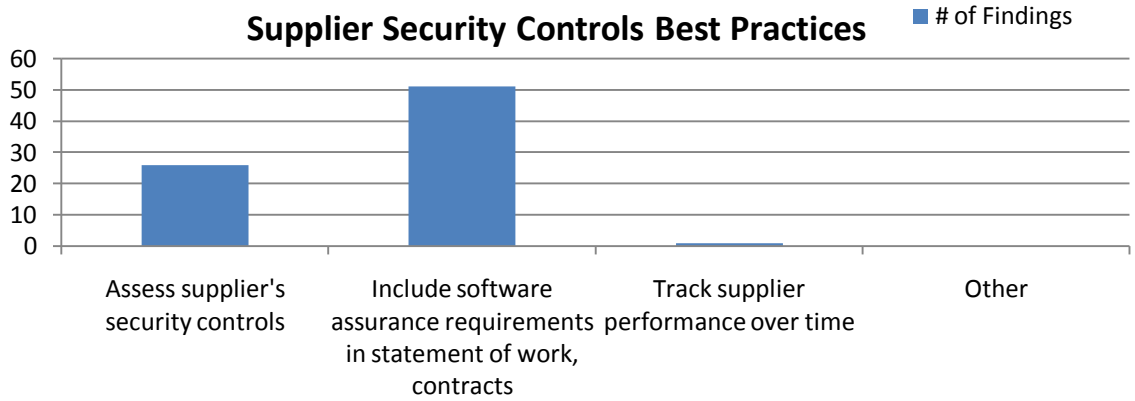

Figure 14. Results for Supplier Security Controls Best Practices

The findings related to Supplier Security Controls Best Practices largely focused on including software assurance requirements in contracts and assessing the security measures or controls employed by the supplier.

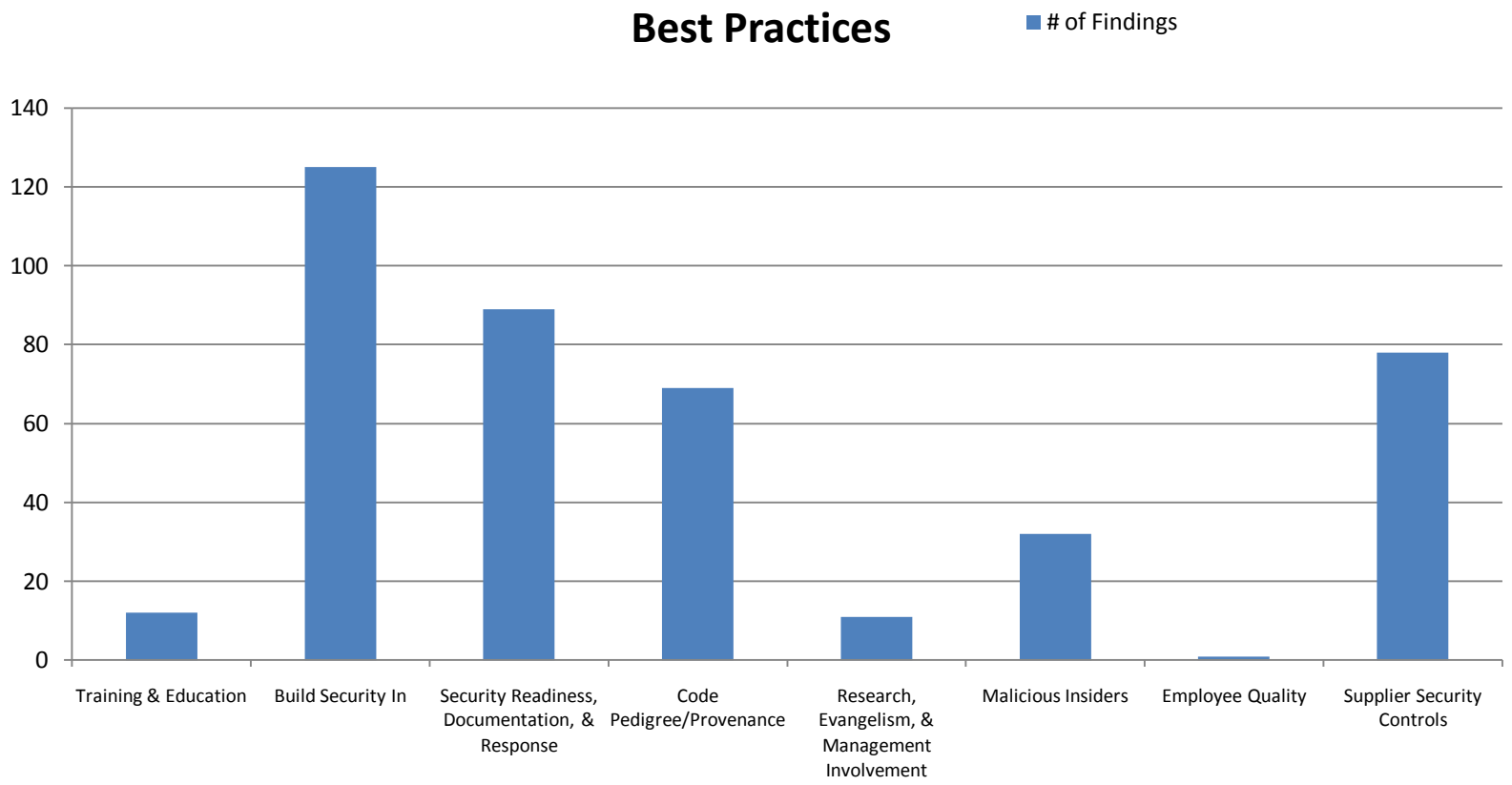

Figure 15. Result Totals for Best Practice Main Categories

This figure shows the totals for each best practice category. Most findings were attributed to Build Security In, Security Readiness, Documentation and Response, Code Pedigree/Provenance, and Supplier Security Controls.

Case Study Survey Response Rate

Only one survey was returned by a program manager during the course of the case study period.

Case Study Survey Data Discussion

Due to the extremely low response rate, the lone response was not analyzed.

Case Study Results

Since no results are provided for this case study, the following are select results from the original sources from which the case study survey questions were based.

From GAO's *Knowledge of Software Suppliers Needed to Manage Risks*, May 2004:

- None of the 16 weapon system programs that were reviewed "could fully identify all foreign-developed software for their systems," but six program offices did have "significant knowledge of foreign software developers"
- 4 (of 16) program offices were able to identify all the software produced by foreign suppliers, and 11 (of 16) program offices were able to identify at least some of the software produced by foreign suppliers
- Prime contractors did not always share information on who actually performed software development for "onboard" systems with government program managers; therefore, that information was not available to the government program managers to use to make risk management decisions to address software security
- 10 (of 16) programs indicated they had very little knowledge of the developers for their "offboard" software, including those portions that may have been developed by foreign suppliers
- 13 (of 16) program officials had almost no insight into the use of foreign developers for any COTS software placed on their systems
- 10 (of 16) of the programs accepted legacy software without fully identifying the sources of development
- 5 of the prime contractors were required to notify the program office concerning their decisions on software subcontracting (only 15 contractors were interviewed)
- Contracts for 12 (of 16) programs contained a requirement that the contractors provide a software development plan that included information on some of their planned suppliers, development risks, and action plans
- 11 (of 16) program managers have not identified foreign supplier involvement in software development as a significant risk to the security of their weapon systems

- The following diagram was developed by Ellen Walker (2005) based upon the GAO report; it depicts the complex relationships between the prime contractor, various types of software, and where the software may originate. The shaded region depicts the limited scope of control program offices typically have, that is, they only see as far as the prime contractor:

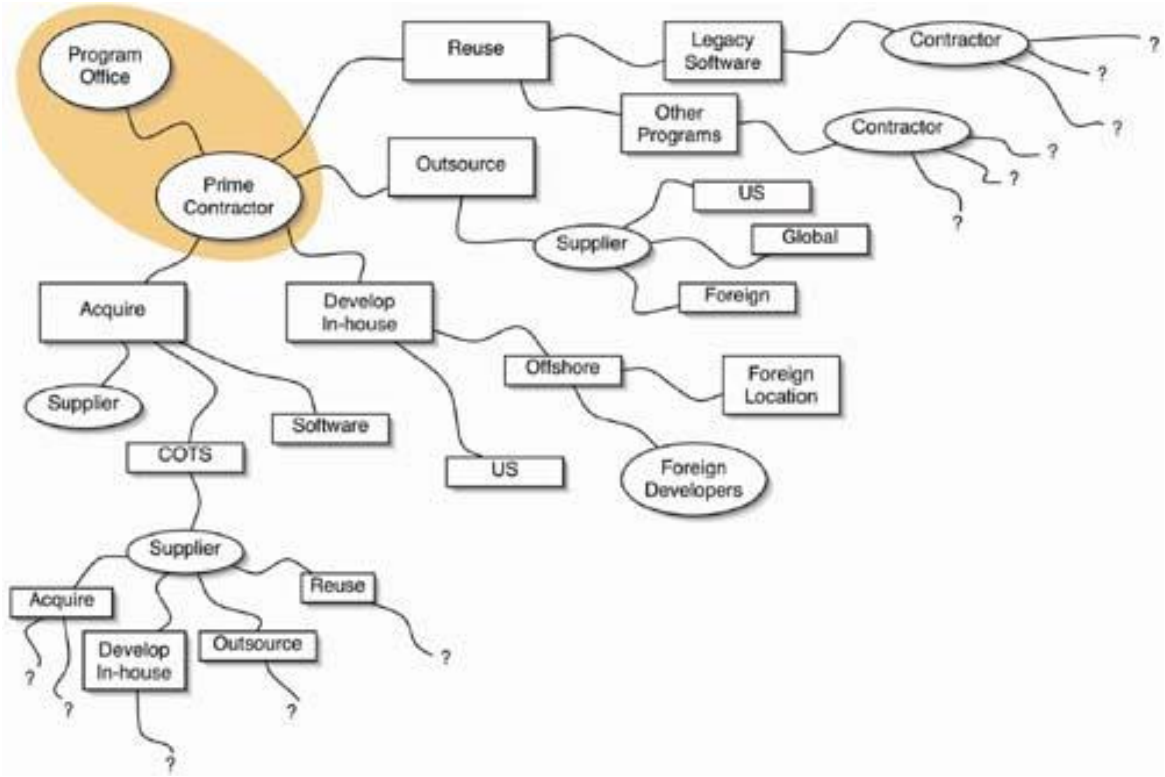

Figure 16. Scope of Supplier Expansion and Foreign Involvement (Walker, 2005)

From the CIO Executive Council's poll, October 2006:

- When asked "*Of the following potential actions that a software vendor could take, which would be most effective in reassuring you that the vendor's products will operate as intended and will be free from flaws, security vulnerabilities and malicious code?*" (pick three), the top results were:

73%	Certify that software meets a designated security target
62%	Provide evidence that software has been scanned for flaws and security vulnerabilities using qualified tools
48%	Provide a list of known flaws and security vulnerabilities
36%	Provide test cases for their software
29%	Provide more transparency or visibility into their development process

- When asked "*Of the following sources of information and/or methods, which do you routinely use to determine if software is free from flaws, security vulnerabilities and malicious code?*" (check all that apply), the top results were:

85%	Internal testing
65%	Contract requirements and/or service level agreements (SLAs)
54%	Reputation of the vendor and product among senior IT executives in other organizations
36%	Third-party testing
35%	Vendor statements or assurance claims

- When asked "*Of the following, which software attributes are most important to your organization?*" (check all that apply), the results were:

95%	Reliable software that functions as promised
70%	Software that is free from security vulnerabilities and malicious code
55%	Ease of integration and configuration
32%	Software that conforms to requirements and industry standards
27%	Convenience and ease of use
11%	Rich feature set

- The following diagrams are also based upon the poll data:

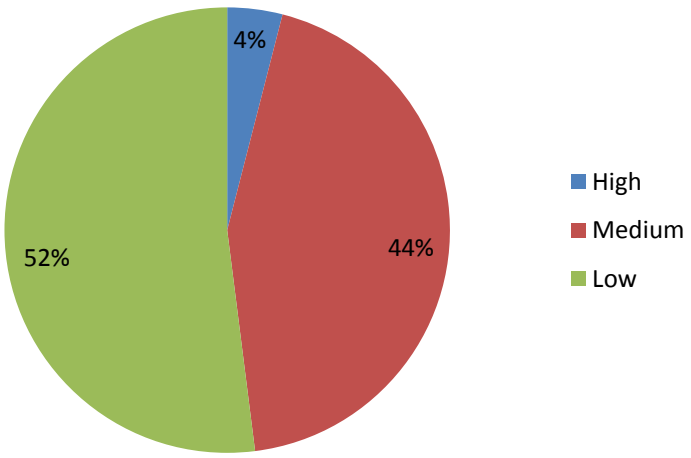

Figure 17. Overall level of confidence that software will function as intended and be free of flaws, security vulnerabilities, and malicious code (Fogerty, 2006)

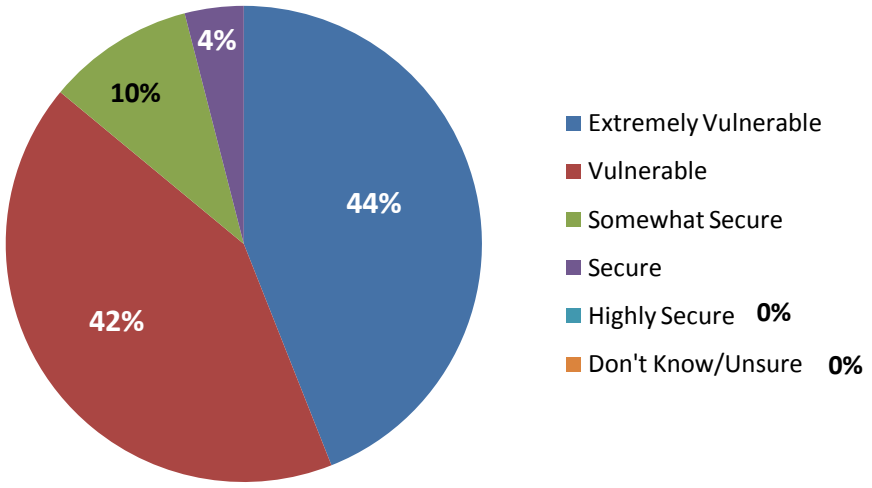

Figure 18. Overall rating of the fundamental security of software without the use of firewalls, intrusion detection systems, anti-virus scanners, etc. (Fogerty, 2006)

Investigative Questions Answered

Question 1: What are the common issues (findings) that present challenges to the development of secure software for military systems?

Result summary: The primary challenges that military systems faced were: (1) including security requirements in the beginning of the development effort, (2) designing for security, (3) assessing system risk through either code testing or certification and accreditation actions, (4) planning for incident response, (5) change management, (6) protecting the supply chain from threats, (7) and concerns regarding supplier security controls.

Question 2: What practices might best resolve the findings?

Result summary: The practices that might best resolve most of the findings were: (1) Build Security In, (2) Security Readiness, (3) Security Response, (4) Code Pedigree/Provenance, and (5) Supplier Security Controls.

Main Research Question

What strategies or practices are best suited for answering the software security challenges the AF is facing as it stands up a formal software assurance program?

Result summary: Build Security In and Security Readiness would be two best practices that the AF could implement immediately to have a large impact, while code pedigree/provenance and supplier security controls would also be important mid-term best practices to implement.

Chapter Summary

This chapter provides the results of the content analysis and identified which best practices the AF should consider implementing as it stands up its software assurance program.

V. Conclusions and Recommendations

Chapter Overview

This chapter discusses my conclusions and recommendations based upon all that I have discovered during the course of my research. It also provides some recommendations for additional research.

Conclusions of Research

The basic conclusion of this research is that yes, *we are bleeding*. The reactive, post facto approach is just like putting band-aids on a wound, and there "are not enough band-aids to stop the bleeding with a laceration of this size (McGraw, 2002)." This "medical epidemic" is entirely man-made and self-inflicted too. Lyle Long (2008) wrote that software is the "Achilles Heel of aerospace systems." I believe it goes beyond just aerospace systems and is the Achilles Heel of our national infrastructure and entire defense systems and networks. We need to stop focusing so much on buying defensive security products; we must be proactive and attack the heart of the problem, which is bad software (Potter & McGraw, 2004). We must target "known problem areas in software development programs (Taylor & McGraw, 2005)" by implementing software assurance best practices AF-wide as soon as possible – the longer we wait, the worse the wound gets. And just because the results of this thesis research recommend certain best practices over others does not mean the others should be ignored.

The software assurance best practices relate to each other, so an improvement in one may also improve another. For example, if you spend resources on training and

education best practices (training your managers, architects, and developers), they most likely will get better at building security in (defining requirements, designing security into application architectures, and writing secure code). Also, to further illustrate this point, one audit had multiple findings related to the malicious insider best practice, but the root cause was traced back to leadership/management failures. Specifically, AF leadership continually funded below the system's maintenance workload requirements, forcing program office personnel to "choose to apply the limited funding available to functional requirements, and not maintenance of system controls." In fact, in FY2006, maintenance funding was only at 66% of requirements, and in one instance, when a critical software update was required, "the program office recalled a laid-off contract employee to implement the update, and then laid him off a second time." (Prentkiewicz & Massey, 2006)

Clearly, despite repeated investigations into software assurance related issues, and recommendations on how to address those issues, the DoD is not taking software security seriously despite the potential for severe, negative impacts on national security. For example, in the AF Audit Agency's report *Implementation of Selected Aspects of Security in Air Force Systems (17 April 2006)*, SAF/XC concurred with the recommendations and agreed to complete several actions to produce software development guidance by certain dates (with the final guidance due 31 August 2006). However, such guidance has not been published, making it look like AF leadership did not take the audit findings and recommended fix actions seriously. Vulnerabilities are "not just a developer problem," they are "a management and business problem as well (Schneider T. , 2006)." We need

to learn that if we cannot afford to secure the system (or keep it secure), then we cannot afford the system, period.

As my research came to a close, I collected several key points that best illustrate the theme of this endeavor:

Transformation/Dependence

- In the DoD, "the transformational effects of information technology, joined with a culture of information sharing, called Net-Centricity, constitute a powerful force multiplier" and the DoD "has become increasingly dependent for mission-critical functionality upon highly interconnected, globally sourced, information technology of dramatically varying quality, reliability and trustworthiness (Schneider W. J., 2007)."
- "During testing, the software operating the [Expeditionary Fighting Vehicle's] guns didn't always fire on command (Merle, 2007)."

Globalization of the Supply Chain & Foreign Influence

- "Some intelligence analysts believe that software offers one of the best mechanisms for technical intelligence collection by a range of adversaries. A foreign intelligence service or a hostile group could infiltrate the global supply chain and surreptitiously introduce "malicious code" that would make it easier to gain remote access to networks and information or remotely trigger disruptive events during a crisis after the product is sold and installed (Lewis J. A., 2007)."
- "Consider that some of [the commercial or contract] products that are employed in highly sensitive applications are being crafted, tested, packaged and supported by individuals who would never be allowed into the locations where those applications are used because of national origin, criminal history, and/or personal behavior (Spafford, 2005)."
- The DoD "should work with its industry partners and commercial industry to improve the assuredness of COTS software (Schneider W. J., 2007)."
- "… some of the hardware and software components in use in critical applications are designed and produced in countries that may be adversaries in future military or political conflict (Spafford, 2005)."
- *E-jihad* is already occurring (ACM Job Migration Task Force, 2006).
- "One foreign-owned contractor appeared to have had access to U.S. classified information for at least 6 months before a protective measure was implemented (GAO, 2007)."
- "How can we trust software written abroad? The answer is that we can't. However, like anything else this is a risk management issue. There is risk even in software

produced in the most secure U.S. environments. The only question is how much risk and at what cost (Fields, 1999)."

- DoD computing systems are "a constant target of foreign exploitation (Schneider W. J., 2007)."
- "While code developed in the United States is not immune from risk, the opportunity for an adversary is greatly enhanced by globalization (Schneider W. J., 2007)."
- "The Nation's defense is dependent upon software that is growing exponentially in size and complexity, and an increasing percentage of this software is being written offshore in easy reach of potential adversaries (Schneider W. J., 2007)."
- "Global software development presents an opportunity for threat agents to attack the confidentiality, integrity, and availability of operating systems, middleware, and applications that are essential to the operation of the U.S. Government and the DoD (Schneider W. J., 2007)."
- Many commercial software companies "develop products in multiple geographic locales. Even a U.S.-based company of any size can and likely does have a worldwide presence if not worldwide development (Schneider W. J., 2007)."
- Ignoring "direct imports of foreign origin COTS, COTS purchased from U.S. companies can be expected to become increasingly foreign sourced (Schneider W. J., 2007)."
- "Both U.S. and foreign-developed software are vulnerable, although the opportunity may be somewhat greater in foreign processes, where an opponent may face fewer obstacles and less scrutiny (Lewis J. A., 2007)."
- "The same technology that has enabled globalization also allows terrorists, criminals and violent ideologues to join forces against larger adversaries with relative ease and to carry out small, inexpensive actions… that will generate a huge return… Our new enemies are looking for gaps in vital systems where a small, cheap action will generate a huge return… The use of 'systems disruption' as a method of strategic warfare gives rise to a nightmare scenario in which any nation – including the United States – can be driven to bankruptcy by an enemy it can't compete with economically (Osinga, 2007)."

Malicious Individuals

- Software security experts, including those in the DoD and Central Intelligence Agency, have "expressed concern that organizations and individuals with hostile intentions, such as terrorist organizations and foreign government economic and information warfare units, could gain direct access to software code by infiltrating or otherwise influencing contractor and subcontractor staff, and then use this code to perpetuate attacks on U.S. infrastructure systems or conduct industrial or other forms of espionage." (Nilsen, 2005)
- The DoD "does not fully know when or where intruders may have already gained access to existing computing and communications systems (Schneider W. J., 2007)."

- Commercial software products have "many weaknesses that are exploitable by even moderately capable hackers (Schneider W. J., 2007)."
- "The offense gets to pick the time, the place and the means to attack a target. The defense must be strong enough to withstand the strength of the attacker at the defender's weakest point (Schneider W. J., 2007)."
- "It is not uncommon for foreign nationals to work on software products within the United States. It is also conceivable that some US citizens working on software projects may be subverted (Goertzel, et al., 2007)."

SOUP

- "Many companies may also embed code from other vendors (open source or code licensed from a third-party) which itself may be of unknown or unproven provenance (Schneider W. J., 2007)."

Leadership

- Decision-makers are "inadequately informed regarding the potential consequences of system subversion, and the value of mitigating that risk (Schneider W. J., 2007)."
- "The DoD should advance the issue of software assurance and globalization on the national agenda as part of a coordinated effort to reduce national cyber risk (Schneider W. J., 2007)."

Policies

- Policies "intended to mitigate information system vulnerabilities focus mostly on operational software security threats, such as external hacking and unauthorized access to information systems, but not on insider threats, such as the insertion of malicious code by software developers (Schinasi, 2004)."
- "It is not currently DoD policy to require any program, even those deemed critical by dint of a Mission Assurance Category I status, to conduct a counterintelligence review of its major suppliers, unless classified information is involved (Schneider W. J., 2007)."
- Existing "software certification mechanisms are inadequate to assure software security (Goertzel, et al., 2007)."

Education

- "One of the root causes of poor software assurance is the current culture of software development. Developers are, in general, not indoctrinated in either the techniques or the values of secure coding practice in their degree programs… Vendors will always need and want to indoctrinate their employees in vendor-specific coding practices, but

they should not need to be teaching the basics of good, defensible programming (Schneider W. J., 2007)."

- Gerald Weinberg's Second Law: "If builders built houses the way programmers built programs, the first woodpecker to come along would destroy civilization (Lewis J. J., 2004)."

In a paragraph discussing the complex nature of detecting vulnerabilities, Schneider W. J. (2007) wrote that current code analysis tools "find about one-third of the bugs prior to deployment that are ever found subsequently, and the rate of false positives is about equal to that of true positives." I contend that one-third of the bugs is better than none at all, and firmly believe we should employ those tools on as many systems we can, as early as we can in system development, and as often as we can throughout the entire system lifecycle.

Significance of Research

This research did not seek a quantifiable impact of software assurance best practices; however, the ASACoE has started using source code analysis tools. This following chart summarizes their metrics from 14 applications as of mid-January 2008:

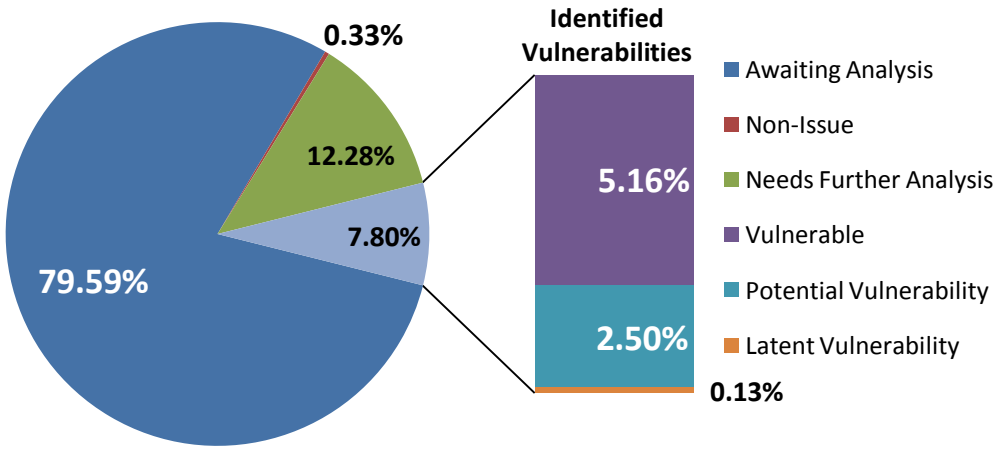

Figure 19. Breakdown of Findings from Source Code Analysis of 14 AF Applications

The ASACoE metrics are significant, because it shows that there are indeed vulnerabilities in our system that need to be addressed right away. And implementing a software assurance program that utilizes the best practices identified in this research most likely will help improve their systems.

Furthermore, this research could be used by other organizations to improve their quality programs in general, including software assurance. A study published in 2002 estimated that software bugs or errors cost the U.S. economy an estimated $60 billion annually (Chaki & Hissam, 2006; RTI, 2002) and a study in 2000 estimated that hacker attacks "will cost the world economy a whopping $1.6 trillion" that year (McDonald, 2000). Attention to quality early in the life cycle, as encouraged by the Build Security In best practice, "leads to defect detection and avoidance," and it is "well-known that such defects, if undetected, can propagate downstream, where the costs of detection and removal are greatly amplified (Devanbu & Stubblebine, 2000)." Studies have shown that fewer schedule deviations, reductions in cost per software line of code, decreases in defect density, and reductions in average costs to fix defects can be achieved (Galin & Avrahami, 2007). Static analysis tools have allowed one organization's security team to become more efficient; they have tripled the number of lines of code they can review in one year without increasing the size of the team. Also, another organization's security team has decreased the application code review time from 3-4 weeks to 1-2 weeks by using static analysis tools. (Chess & West, 2007)

A review of 11 weapon systems showed that quality and reliability problems have resulted in "billions of dollars in cost overruns, years of schedule delays, and reduced

weapon system availability (Sullivan, 2008)." Another review of 62 DoD weapon systems found "that only 27 percent of the programs demonstrated that they had attained a stable design at the completion of the design phase (Sullivan, 2008)." One AF organization that has implemented software process improvements has shown a "return of more than 4.48 dollars on every dollar invested" and if work that is in progress is excluded, the business value ratio is 6.27 to 1 (Herbsleb, Carleton, Rozum, Siegel, & Zubrow, 1994).

If only a fraction of the over "3 million or more threats each day (USAF, 2008)" on the AF network are stopped by implementing the recommended software assurance best practices, the military will have improved its warfighting and information assurance posture greatly. A wider implementation of software assurance best practices certainly would help improve system security and potentially reduce any corresponding economic impact, such as IT maintenance expenditures. This is critical as the DoD moves "to complex net-centric, software-based services and applications" that increasingly rely on "widely targeted" commercial software, which is often developed offshore in "uncontrolled development environments (Goertzel, McKinley, & Winograd, 2005)."

Limitations

Audits typically are based upon organizational policies that exist at the time of the audit; therefore any findings and recommendations will only reflect violations of those policies. Many of the documents that I examined for this research identified deficiencies in current DoD and AF policies as they relate to software assurance. In light of that, it is not expected that past audits would reveal many findings or recommendations directly

related to software assurance, or not identify the security challenges that are most critical to the AF. Additionally, based upon my personal experience with audits, organizations do not offer up information that may make them look bad in an audit, so if the auditor does not look for or does not ask for the right things, the audit may not reveal the true problems. Auditor quality becomes an issue.

Findings in the audits may also be faulty. For example, a software issue was cited as one cause for the loss of one F-22 test aircraft (Kresge, 2005; Jackson, Thomas, & Millett, 2007) and the F-22A program had to spend an unplanned $400 million to "address numerous quality problems and help the system achieve its baseline reliability requirements" after being in production for 7 years (Sullivan, 2008). Additionally, in February 2007 while twelve F-22A aircraft were enroute from Hickam Air Force Base in Hawaii to Kadena Air Base in Japan, the entire group had to turn back when six of the aircraft experienced a navigational system malfunction that was later determined to be caused by a software glitch (Songini, 2007). Since quality and reliability relate very closely to security, these findings cast doubt on the conclusion in the 2007 *Report of the Defense Science Board Task Force on Mission Impact of Foreign Influence on DoD Software* that says the "F-22 appears to be at the high end of weapons programs when measured against current efforts for secure software development of its most critical systems."

The software assurance challenge may be much better, or much worse than thought. We may never know the true reality, especially since, as "the amount of software on weapon systems increases, it becomes more difficult and costly to test every

line of code (Schinasi, 2004)." In my opinion, this does not mean however that we should give up on implementing software assurance best practices.

Recommendations for Action

As stated in the conclusion section above, the DoD needs to *proactively* "build software that can withstand attack" through a "process of designing, building, and testing software for security" and getting "to the heart of the matter by identifying and expunging problems in the software itself (McGraw, 2002)." Other than implementing the software assurance best practices described in Chapter II throughout the DoD, I took note of numerous other recommendations throughout my research.

I believe that the AF's current effort to minimize the amount of time it takes to complete the annual information assurance (IA) training is a bad trend. *Every* leader, manager, system developer, and user needs to be aware of and understand how severe the risk is and what their role is in proactively preventing cyber war from severely impacting our national security. Ancillary training that is *click-click-click-done*, like the IA training has become, simply does not provide the necessary attention the problem deserves, and may do more harm than good in the long run.

Several of the audits cited issues with unauthorized software, which have the potential for being very insecure… especially if malicious insiders circumvented security controls to install the software. While this problem may be drastically reduced through the AF's Standard Desktop Configuration project, if future audits show no impact, then there is sufficient concern to warrant a partnership between the DoD and industry to address this problem and develop potential solutions.

As mentioned in the limitations section above, audit and auditor quality may need improvement. I recommend that the AF Audit Agency engage the AF Communications Agency experts and develop more thorough and more relevant audit procedures, to include training for the auditors prior to conducting information assurance related audits. The AF audits may need to go beyond the narrow focus of what is specified in the formal AF policy documents.

There is a clear need to improve the system acquisition process so that software assurance concerns are properly addressed. Many sources included great recommendations that should be implemented, including:

- Focusing software-related practices on "Four P's": (1) "Practices for creating and updating software in a software assurance environment," (2) "Processes supporting software assurance practices," (3) "Protection from threats to code during and after development," and (4) "Pedigree of those involved in the software development/maintenance process" (INPUT, 2005)
- Providing Request for Proposal (RFP) and Statement of Work (SOW) templates that include software assurance language; numerous suggestions have already been published for these documents, but final templates need to be published, advertised, distributed, and put into mandatory use
    - As previously discussed in Chapter II, the NDIA Systems Assurance Guidebook, DoD/DHS Software Acquisition Working Group's Draft Software Assurance Acquisition Guide, and SAFECode Best Practice whitepaper each provide content recommendations
    - Additionally, besides just identifying questions for the vendor, INPUT (2005) discusses providing assistance "to buyers on how to interpret vendor answers"
- Giving preference to suppliers with a track record of quickly fixing reported flaws (DSB, 2001).
- Implementing a "scalable supplier assurance process to ensure that critical suppliers are trustworthy" and defining an "evaluation regime that is capable of reviewing vendors' actual development processes and rendering a judgment about their ability to produce assured software (Schneider W. J., 2007)."
- Developing, without exception, software by in-house organizations or by vendors who have demonstrated Capability Maturity Model (CMM) Level 3 processes within the previous 24 months. (Hansen & Nesbit, 2000)

- Requiring updated, mandatory training of program managers and key program staff before initiation of any software-intensive system development program and at selected key milestones. (Hansen & Nesbit, 2000)
- Increasing "knowledge and awareness among its cyber-defense and acquisition communities of the capabilities and intent of nation-state adversaries (Schneider W. J., 2007)."
- Weighting "past performance and development process maturity in the source selection process" and keeping software-specific performance data "collected during the Contractor Performance Assessment Reporting (CPAR) process" in a central database (Hansen & Nesbit, 2000).
- Positioning security "as an investment and not as an expense (Woody, 2007)."
- Considering what can go wrong as thoroughly as functional needs (Woody, 2007; van Wyk K. , 2007).
- Basing all decisions in the acquisition process "on not only costs and benefits, but also risks (Polydys, Ryan, & Ryan, 2006)."
- Holding back a portion of the contract value until the product demonstrates key metrics over the lifetime of the product. For example, Intelsat "makes progress payments to its manufacturers throughout development and production" but "holds about 10 to 20 percent of the contract value to award to the manufacturer after a satellite is successfully launched." The "10 to 20 percent is paid to the manufacturer over the expected life of the satellite, which is typically 15 years, when the satellite performs as expected (Sullivan, 2008)." For software, we could hold back a portion of the contract value to see if the software's security vulnerabilities are kept under a certain level, or similarly, we could hold back a portion of the contract value on the forthcoming KC-X to see if it meets its maintenance goals during the first 5 years of operational flight.
- Requiring "program managers to collect and maintain information on software suppliers, including software from foreign suppliers" and evaluate the information "periodically to assess changes in the status of suppliers and adjustments to program security requirements (Schinasi, 2004)."

I feel that it is imperative that the DoD finds a way to get around the red tape and incorporate these recommendations into policy in the fastest way possible.

If the U.S. can afford to spend $*n* billion to clean up after the outbreak of a malicious Internet worm, then the U.S. can afford to invest one-half of that (corporate, private, and government funds) to prevent the next ten global malicious incidents from

occurring. If I had the money to spend, the following are a few of the first actions I would take:

- Organize a group of industry, academia, and government experts to develop an undergraduate and a graduate level computer science course on secure software development that could be freely used by any university; there are already some collegiate courses out there, which would serve as good benchmarks
- Organize a group of industry, academia, and government experts to develop an undergraduate and graduate level business management course on managing IT organizations that could be freely used by any university; the syllabus would include a lecture on secure software development practices
- Pay 100% of the course tuition for any U.S. citizen who wanted to attend either course, in exchange for a research paper to be written by each student as part of the coursework; the topic would need to be about some aspect of software assurance and approved by the professor
- Pay 25-50% of the course tuition for any non-U.S. citizen attending the course at a U.S. university, with the same research paper stipulation
- Buy every DoD software developer a copy of "Writing Secure Code, 2nd Edition" by Michael Howard and David LeBlanc, "19 Deadly Sins of Software Security" by Michael Howard, David LeBlanc, and John Viega, and/or other great software assurance books
- Implement the K-12 education and "human capital tax credits" proposals as described by Nilsen (2005)

I firmly believe that the AF desperately needs to expand the ASACoE and scan all AF web applications that touch the public Internet (and AF contracted .com websites) for vulnerabilities using ASACoE's code analysis tools (Fortify SCA, Watchfire AppScan, and AppSec Inc. AppDetective). Finally, an AF-level directive, *and funding*, to have this done by the end of CY2008 should be sent ASAP. Many security experts agree that firewalls are not enough to prevent intrusions – bad software is the problem; malicious individuals can bypass firewalls and get in through web applications (Epstein, Matsumoto, & McGraw, 2006; Viega & McGraw, 2001). The bottom line is this: we must secure our web border! And the first step is to scan the web-code.

Recommendations for Future Research

Software assurance is a deep, complex topic that offers numerous opportunities for additional research. One topic was previously discussed: a quantitative study of whether or not software assurance best practices can have an effect on software security within specific AF systems. The case studies described in Chapter III and IV most certainly afford an opportunity for a researcher to complete.

Goertzel, et al. (2007), list several of the many definitions for software assurance. Both the Committee on National Security Systems (CNSS) and DoD include the phrase "level of confidence," which suggests that security assurance in software can be quantified. A potential topic for further exploration might be the development of a model that does precisely that. Based upon the CNSS and DoD definitions, I surmise that the following draft model might serve as a baseline that could be developed in more detail:

$$\hat{y} = 100 - \hat{\beta}_1(I) - \hat{\beta}_2(U) - \hat{\beta}_3(K) + \varepsilon$$

Where:

$\hat{y}$ = Percent confidence that the software will function as intended,
I = the number of undiscovered intentional (malicious) coding errors,
U = the number of undiscovered unintentional coding errors,
K = the number of known, but unfixed coding errors, and
ε = the random error.

In a presentation, Goertzel (2008) asks "What is secure software?" and the answer she provides is Dependability, Trustworthiness, and Resiliency. So, alternatively, the following draft model might also be worth exploring:

$$\hat{y} = 100 + \hat{\beta}_1(D) + \hat{\beta}_2(T) + \hat{\beta}_3(R) + \varepsilon$$

Where:

$\hat{y}$ = Percent confidence that the software will function as intended,
D = A measure of Dependability,
T = A measure of Trustworthiness,
R = A measure of Resiliency, and
ε = the random error.

Also, I believe AFIT and the AF Academy should partner with Microsoft, Secunia, Symantec, and the Software Engineering Institute at Carnegie Mellon University to conduct software assurance research. Those organizations most likely have more application security data than they can handle, so why not have AFIT and AF Academy students collaborate on research projects? For example, as briefly mentioned in Chapter II, Secunia's PSI application that collects all sorts of data on home users' applications is probably a data gold mine just waiting to be exploited (in a good way).

Additionally, I had many other thoughts on topics that I believe would be worthy of immediate, additional research. They are listed below, along with either the direct source of the topic or the source that gave me the idea for the topic.

Education

- Since the education investments described in Chapter V are not likely to happen soon, a great research topic would be an examination of the few software assurance courses offered at various universities and determine how to improve the curriculum and/or best implement the curriculum in the schools that do not yet offer software assurance courses. Also, the study might want to examine who is taking the courses (nationalities, sex, etc.), and what type of companies they go to work for after college. This might reveal interesting information on which vendors are specifically recruiting employees with software assurance knowledge. In light of the recommendations of this research to assess supplier security controls and consider employee quality, this information might be beneficial to the DoD acquisition community. How many Fortune 500 companies are taking security knowledge into account when hiring IT personnel? Does the AF do the same when hiring government civilians?

AF System Software Assurance

- Of the Non-secure Internet Protocol Router Network (NIPRNET) systems in the AF's Enterprise Information Technology Database Repository (EITDR), how many have had source code analysis performed and how many of these can be reached from the Internet? Statistical methods such as an analysis of variances could be part of the research design to examine the differences between Mission Assurance Category I, II, and III systems.
- Replicate the CIO Executive Council survey discussed in Chapters III and IV, but instead poll AF Communications Squadron Commanders or Communications and Information Directors (A6s) from the Major Commands (MAJCOMs) to determine their software assurance confidence levels for comparison against corporate Chief Information Officers.
- Conduct a study to see how consistent (or inconsistent) organizations are at implementing or following through with the agreed to actions as a result of an audit. Reference the example given in Chapter V regarding the AF Audit Agency's report *Implementation of Selected Aspects of Security in Air Force Systems (17 April 2006)*, and the apparent failure to complete the actions by the agreed upon suspense dates.
- ASACoE is using source code analysis tools to collect data from local programs; as they branch out and collect data from units at other bases, case studies could be conducted to determine why units or different developer types (military, government civilians, contractors) are "better or worse" than others.
- Compare/contrast the defect density figures in Davis, et al. (2004), and other studies, to that of GOTS applications.
- What processes are used by AF units to assess commercial software products and are these processes adequate? (Jarzombek, Considerations for Quality & Assurance with Scale and Uncertainty, 2007)

Supplier Security

- What is the median length of time from when a product is released and a vulnerability is made public? For example, for Microsoft Windows XP, what is the median number of days between the product's public release and when each vulnerability was publicly disclosed? Does comparing the differences between the means of different products or companies yield any significant patterns? Are vulnerabilities in one company's products likely to be discovered earlier than another company's products? The results of this study would be important if the DSB (2001) recommendation to give preference to suppliers with a track record of timely fixes is ever implemented. Also, since the cost of fixing defects increases over time (Chess & West, 2007; Chess & McGraw, 2004), there is potential for huge financial savings.
- "Consumers generally reward vendors for adding features and for being first to market. These two motivations are in direct tension with the goal of writing more secure software, which requires time consuming testing and a focus on simplicity.

Nonetheless, the problems of software insecurity, viruses, and worms are frequently in the headlines; why does the potential damage to vendors' reputations not motivate them to invest in more secure software?" (Ozment, 2007) Can the damage to vendor reputations be quantified? Do CIOs feel that the exposure of security vulnerabilities in their products puts the company's reputation at risk?

- What percentage of published vulnerabilities are repeats of previous mistakes? (Chess & McGraw, 2004)
- What processes are currently in use by the acquisition community to evaluate the capabilities of suppliers to deliver secure software? (Jarzombek, Considerations for Quality & Assurance with Scale and Uncertainty, 2007)

Software Development Process

- Since many organizations struggle with the task of ensuring security requirements are considered early in system development, perhaps research could be conducted to identify what makes a "good" security requirement.
- Many software development life cycle processes and methods exist that could be compared and contrasted to determine which ones are best for the Air Force. Goertzel, et al. (2007) identify 6 development methodologies, 8 modeling and assessment techniques, 7 requirements specification and modeling techniques, and 5 security analysis and test techniques.
- "Developers are told to "deploy anyway" in the face of known security gaps based on cost and schedule demands. When does security become a strong motivator for decision making (Woody, 2007)?"
- What measurement or metrics should be collected to better support decision-making associated with IT and software assurance? (Jarzombek, Considerations for Quality & Assurance with Scale and Uncertainty, 2007)
- How do we quantify security trade-offs? (McGraw, 2003)
- How do we reveal and minimize assumptions in security system designs? (McGraw, 2003)
- "What needs to be done to get to the point where tools integrated in development environments would incrementally and continuously check, test, and analyze the various artifacts—code, design, requirements—in regards to security, pretty much the way today tools like CruiseControl support continuous configuration management, regression testing, and integration (Beznosov & Kruchten, 2004)?"

National-level Issues

- What criteria would enable the DoD to "decide when a cyber attack is an act of war?" and "Would it be possible for some kind of action inside a network to lead to a shooting war without some kind of overt physical threat occurring first?" (florescent_beige, 2008)

- How should certification and accreditation processes and policies be changed to better address software assurance? (Jarzombek, Considerations for Quality & Assurance with Scale and Uncertainty, 2007)
- "How are different sectors of the U.S. economy being affected by offshoring, and what sectors are emerging as new sources of comparative advantage?" (Nilsen, 2005) (Recommended data sources, methodological approaches, and potential limitations are provided in the GAO report.)
- "To what extent is offshoring affecting employment in the U.S." (Nilsen, 2005) (Recommended data sources, methodological approaches, and potential limitations are provided in the GAO report.)
- "To what extent is offshoring affecting the distribution of income in the U.S.?" (Nilsen, 2005) (Recommended data sources, methodological approaches, and potential limitations are provided in the GAO report.)
- "What are the reemployment experiences of workers dislocated due to services offshoring?" (Nilsen, 2005) (Recommended data sources, methodological approaches, and potential limitations are provided in the GAO report.)
- Similarly, what are the reemployment experiences of AF officers impacted by force shaping and reduction in force measures?
- What is the economic impact of cybercrime on the AF? (Reference Powner & Rhodes, 2007)
- Test the hypothesis that "globalization will only accelerate the offshoring trend (Goertzel, et al., 2007)."
- Examine the hypothesis, in a software assurance context, that "offshoring could have positive effects on national security" since "increased international trade may reduce the threat of international tensions because countries with integrated economies have a stake in one another's well-being (Nilsen, 2005)." This is also known as the "Golden Arches Theory of Conflict Prevention (Friedman, 2000)."

Understanding Flaws

- After 20 years, buffer overflows continue to be the most common cause of vulnerabilities; why? What can be done and what would be the best way to end this perennial thorn? (Chaki & Hissam, 2006)
- Study the taxonomy categorization frequency of known vulnerabilities to determine which areas should receive greater focus and attention; perhaps compare COTS vs. GOTS to see if they have similar categorization frequencies. (Landwehr, Bull, McDermott, & Choi, 1994)
- How do we prevent or withstand denial-of-service attacks? (McGraw, 2003)

Understanding Malicious Individuals

- Besides the "combination of ego and desire for intellectual challenge (Goertzel, et al., 2007)," what other motivations exist for blackhats?

- Examine the ethical, legal, and economic implications of paying blackhats or hackers for reporting previously unknown vulnerabilities. As noted by Goertzel, et al., this is already being done by iDefense. *What if the AF did this?*
- Examine hacker motivation in terms of Abraham Maslow's Hierarchy of Needs.

Miscellaneous

- Examine software piracy risk/cost models, such as the one identified by Devanbu and Stubblebine (2000)
- The President's Information Technology Advisory Committee (PITAC) Report titled *Cyber Security: A Crisis of Prioritization (February 2005)*, identified 'secure software engineering and software assurance' and 'metrics, benchmarks, and best practices' in its top 10 areas in need of increased support; specific ideas for future research may be obtained by reviewing the report in detail or contacting the President's Council of Advisors on Science and Technology (PCAST) of which PITAC is now part of. (Jarzombek, Considerations for Quality & Assurance with Scale and Uncertainty, 2007)
- Additional research ideas may be found by reviewing the: (1) Report of the 2nd National Software Summit titled Software 2015: A National Software Strategy to Ensure U.S. Security and Competitiveness (29 April 2005), (2) Report by the Interagency Working Group on Cyber Security and Information Assurance titled Federal Plan for Cyber Security and Information Assurance Research and Development (April 2006), and (3) Report of the ACM Job Migration Task Force titled Globalization and Offshoring of Software (2006).

Summary

This chapter provided my research conclusions, research significance, research limitations, recommendations for action, and recommendations for future research.

Appendix A: Content Analysis Document List

The following documents were examined during the content analysis:

Defense Science Board (DSB)

- *Report of the Defense Science Board Task Force on Defense Software*, November 2000
- *Report of the Defense Science Board Task Force on Mission Impact of Foreign Influence on DoD Software*, September 2007

DoD Inspector General (IG)

- *Computer Security for the Defense Civilian Pay System*, DoD Inspector General, 16 March 1999, Report #99-107
- *Computer Security for the Defense Civilian Pay System*, DoD Inspector General, 8 April 1999, Report #99-128
- *General Controls over the Electronic Document Access System*, DoD Inspector General, 27 December 2000, Report #D-2001-029
- *Information Assurance at Central Design Activities*, DoD Inspector General, 7 February 2001, Report #D-2001-046
- *Controls for the Electronic Data Interchange at The Defense Finance and Accounting Service Columbus*, DoD Inspector General, 6 April 2001, Report #D-2001-095
- *User Authentication Protection at Central Design Activities*, DoD Inspector General, 29 July 2002, Report #D-2002-135
- *Security Controls for the Defense Procurement Payment System*, DoD Inspector General, 11 October 2002, Report #D-2003-009
- *Development Testing of Prophet Mission-Critical Software*, DoD Inspector General, 22 January 2003, Report #D-2003-051
- *Development Testing of Space Based Infrared System Mission-Critical Software*, DoD Inspector General, 24 November 2003, Report #D-2004-022
- *Defense Finance and Accounting Service Corporate Database User Access Controls*, DoD Inspector General, 7 December 2005, Report #D-2006-033
- *DoD Organization Information Assurance Management of Information Technology Goods and Services Acquired Through Interagency Agreements*, DoD Inspector General, 23 February 2006, Report #D-2006-052
- *Select Controls for the Information Security of the Ground-Based Midcourse Defense Communications Network*, DoD Inspector General, 24 February 2006, Report #D-2006-053
- *Review of the Information Security Operational Controls of the Defense Logistics Agency's Business Systems Modernization-Energy*, DoD Inspector General, 24 April 2006, Report #D-2006-079

- *The General and Application Controls over the Financial Management System at the Military Sealift Command,* DoD Inspector General, 2 January 2007, Report #D-2007-040
- *Defense Information Systems Agency Controls over the Center for Computing Services,* DoD Inspector General, 9 April 2007, Report #D-2007-082

Government Accountability Office (GAO)

- *Report to the Secretary of Defense on DoD Information Security – Serious Weaknesses Continue to Place Defense Operations at Risk,* United States General Accounting Office (GAO), August 1999
- *Report to Congressional Requesters on Defense Acquisitions – Stronger Management Practices Are Needed to Improve DoD's Software-Intensive Weapon Acquisitions,* United States General Accounting Office (GAO), March 2004
- *Report to Congressional Requesters on Defense Acquisitions – Knowledge of Software Suppliers Needed to Manage Risks,* United States General Accounting Office (GAO), May 2004
- *Report to Congressional Committees on Defense Acquisitions – Restructured JTRS Program Reduces Risk, but Significant Challenges Remain,* United States Government Accountability Office (GAO), September 2006
- *Report to Congressional Committees on Best Practices – Increased Focus on Requirements and Oversight Needed to Improve DoD's Acquisition Environment and Weapon System Quality,* United States Government Accountability Office (GAO), February 2008

AF Audit Agency (AFAA)

- *Financial Management System Controls, 45th Space Wing, Patrick AFB, FL,* Installation Report of Audit, Air Force Audit Agency, 12 May 2003, Report #F2003-0036-FDW000
- *Computer Hardware and Software, Air Force Weather Agency, Offutt AFB, NE,* Installation Report of Audit, Air Force Audit Agency, 6 June 2003, Report #F2003-0056-FBG000
- *Financial Information Resource System Controls,* Audit Report, Air Force Audit Agency, 15 March 2004, Report #F2004-0003-FB2000
- *System Controls for Financial Inventory Accounting and Billing System,* Audit Report, Air Force Audit Agency, 20 September 2005, Report #F2005-0010-FB2000
- *Computer Software Licenses (Revised), Air Force Materiel Command, Wright-Patterson AFB, OH,* Installation Report of Audit, Air Force Audit Agency, 22 September 2005, Report #F2005-0031-FCW000
- *Reliability of Data Supporting Air Force Information and Logistics System,* Audit Report, Air Force Audit Agency, 15 November 2005, Report #F2006-0001-FB2000

- *Automated Data Processing Equipment Controls, 225th Combat Communications Squadron, Alabama Air National Guard, Gadsden, AL,* Installation Report of Audit, Air Force Audit Agency, 16 February 2006, Report #F2006-0039-FDD000
- *Implementation of Selected Aspects of Security in Air Force Systems*, Audit Report, Air Force Audit Agency, 17 April 2006, Report #F2006-0004-FB2000
- *Automated Civil Engineer System-Real Property Controls,* Audit Report, Air Force Audit Agency, 12 April 2006, Report #F2006-0003-FB2000
- *Controls for the Wholesale and Retail Receiving And Shipping System*, Audit Report, Air Force Audit Agency, 19 May 2006, Report #F2006-0006-FB2000
- *Missile Readiness Integrated Support Facility/Integrated Missile Database System Controls,* Audit Report, Air Force Audit Agency, 30 May 2006, Report #F2006-0007-FB2000
- *System Controls for Item Manager Wholesale Requisition Process System*, Audit Report, Air Force Audit Agency, 21 June 2006, Report #F2006-0008-FB2000
- *Controls for the Security Assistance Management Information System*, Audit Report, Air Force Audit Agency, 13 July 2006, Report #F2007-0006-FB2000
- *Selected Aspects of Automated Information Systems Access and Security, 42d Air Base Wing, Maxwell AFB, AL*, Installation Report of Audit, Air Force Audit Agency, 23 August 2006, Report #F2006-0081-FDD000
- *Air Force Equipment Management System Controls*, Audit Report, Air Force Audit Agency, 25 September 2006, Report #F2006-0011-FB2000
- *Military Personnel Data System Controls,* Audit Report, Air Force Audit Agency, 20 November 2006, Report #F2007-0001-FB2000
- *Local National Payroll System, 31st Fighter Wing, Aviano AB, Italy*, Installation Report of Audit, Air Force Audit Agency, 16 March 2007, Report #F2007-0037-FDE000
- *Reliability, Availability, Maintainability Support System for Electronic Combat Pods System Controls*, Audit Report, Air Force Audit Agency, 25 May 2007, Report #F2007-0004-FB2000
- *Computer Software Management, 151st Air Refueling Wing, Salt Lake City, UT*, Installation Report of Audit, Air Force Audit Agency, 9 July 2007, Report #F2007-0050-FCI000
- *Standard Base Supply System Controls*, Audit Report, Air Force Audit Agency, 13 July 2007, Report #F2007-0005-FB2000
- *Software Management, 146th Airlift Wing, Channel Islands ANGB, CA*, Installation Report of Audit, Air Force Audit Agency, 1 August 2007, Report #F2007-0058-FCI000
- *Standard Materiel Accounting System Controls,* Audit Report, Air Force Audit Agency, 22 August 2007, Report #F2007-0007-FB2000
- *Nonappropriated Fund Transformation system Implementation (Phase 1), Air Force Services Financial Management System Controls,* Audit Report, Air Force Audit Agency, 31 October 2007, Report #F2008-0001-FB2000

Appendix B: GAO's Original Interview Questions

This appendix contains the original questions that the *United States General Accounting Office (GAO) Report to Congressional Requesters on Defense Acquisitions – Knowledge of Software Suppliers Needed to Manage Risks*, published in May 2004, was based upon.

Questions for the DoD Software Program Office:

1. What security risks (malicious coding, unauthorized access, use of COTS) have been identified on your program? To what degree, if any, is foreign involvement considered an additional risk factor in these areas?

Follow-up:

1. Were studies or reviews conducted to determine program security risks?
2. In the process used to determine that unauthorized access to classified or program technical information was a security risk, what potential risks were determined not to be a risk?
3. Please describe the security practices identified above? Are they different from standard DOD security practices? Does your program have a Program Protection Plan?

2. Are you able to determine what portions of your software have been developed by foreign entities, either foreign suppliers or foreign nationals working for U.S. suppliers? If so, are foreign suppliers or nationals responsible for significant levels of software content or for software development in critical system areas?

Follow-up:

1. Why was the decision made to use only U.S. software suppliers? Was security a factor in this decision?
2. Is there a specific requirement for U.S. only software development on your program? If so, why, and how is this requirement documented and enforced?
3. Has ABC company, or its U.S. suppliers, hired any foreign developers to work on XYZ program software at U.S. facilities?
4. Is there any non "flight software" on the XYZ program that was developed overseas, or in the U.S. by foreign nationals?

3. What processes are in place to manage security risks for software development, where necessary (access controls, software tools, etc.)? Would any of these techniques be considered above and beyond those normally required for developing software? Are any of these specifically targeted toward mitigating risks associated with foreign developed software?

Follow-up:

1. Who implements these security processes? Prime contractor? Sub contractors? Program office?
2. Does your program have any processes in place that would mitigate the risks associated with malicious code inserted by contractors?

4. Which laws/policy/guidance govern software security or the use of foreign software/foreign developers on your program?

_____ITAR _____ DOD 5000 _____DODD 8500 ____NISPOM

Follow-up:

1. Do you limit the use of foreign developed software or COTS, either contractually or in any other manner?

5. Briefly explain the processes your program uses to ensure compliance with these laws/policies both within the program office and with contractors (i.e. contractual requirements, site visits, etc)?

Follow-up:

1. Do you have any processes in place to verify that contractors are following relevant rules/regulations? What are these processes?

6. Does your program use specific provisions to limit or constrain the use of foreign nationals or foreign firms for developing software? If so, please describe briefly.

Follow-up:

1. Please share with us the information you alluded to, which requires a security clearance, at our in-person meeting with you.

7. Are your contractors required to report or notify DOD or the program office regarding the use of foreign firms or employees for software development?

Follow-up:

1. Is COTS software allowed on XYZ program?

2. Is any legacy code used on XYZ program? If so, do you know the origin of this code?
3. Does XYZ program use software algorithms from a software library? Or any other software of unknown pedigree?

Requested Documents:

1. Copy of XYZ program contract (electronic, if available)
2. Copy of Program Protection Plan (if available), or other documents detailing program software security practices.
3. Document/Report indicating that unauthorized access is a risk for your program (as identified in question 1)
4. Document (if any) that indicates foreign developed software is not allowed on XYZ program.
5. Contract, or other document, requiring contractor and subcontractors to not use foreign developed software.
6. Any relevant studies/reports on software security of XYZ software.

Questions for the Program's Contractor:

1. Guidance from DOD/Program Office
   a. Describe the overall guidance you receive from the XYZ program office concerning the use of foreign software developers (either foreign companies or foreign nationals employed by U.S. suppliers). Are you specifically limited as to those sources you may or may not use for software development?
   b. If limited, what is the mechanism for doing so (FAR, ITAR, NISPOM, contract, business case, other)? Do these requirements flow down to your sub-contractors?
   c. Are you required to notify the XYZ program office or any other DOD organization prior to or after the selection of a foreign source for software development? If so, is this process unique to software development or part of an overall requirement to report all foreign contracting (FAR/DFAR/Buy American)?
   d. Has any guidance been issued with regard to how software and related technologies must be developed, managed, or controlled? Is this guidance specific to XYZ software, or part of general export control or other mechanisms for protecting sensitive or technical data and/or hardware?
   e. What is the impact, if any, of the guidance issued relative to software sourcing or development (i.e. additional resource burdens)?
   f. What role, if any, do other government organizations (DCMA, DSS, etc.) play in guiding, monitoring, and/or inspecting the XYZ program or ABC company facilities?

2. Company Software Development/Security Practices
    a. Does ABC company have any corporate policies, guidelines, etc. relating to software, manufacturing, or security? Are these policies specific to government/defense contracts?
    b. Have the ABC facilities for the XYZ program been rated based on the SEI/CMM software development models? If so, is this a corporate policy for government contracts? Is there a minimum SEI Level required for ABC company software development facilities?
    c. What, if any, company specific mechanisms are utilized to minimize software and/or security risks (be as general as necessary).
    d. Does ABC company employ any software or security efforts in developing software for the XYZ program that would be considered above and beyond general corporate practices? If so, please describe.

3. Risk mitigation
    a. Describe how software security requirements are defined and/or communicated from the XYZ program office (contract, SOW, operational requirements document, etc.). Is ABC company or the program office generally responsible for determining software security requirements? Is foreign software development risk a specific factor for your program, or is it considered as an element of larger software risks?
    b. Describe the nature of software management reviews related to the XYZ program. Are elements of these reviews related to software security? Are the independent reviews or are they held as part of normal risk management reviews?
    c. Please describe any current risk identification and/or mitigation practices that you employ to protect software? Are these tactics specifically focused on software security, or as part of higher-level program risks (cost, schedule)? If directly aimed at software protection, are any of these specific to foreign software development risks (as opposed to functionality or quality assurance)?
    d. What role, if any, does ABC company and/or its suppliers play in the development of the basic security documents for the XYZ program (Program Protection Plan, Technology Assessment/Control Plan, Security Classification Guide, etc)?
    e. Provide an overview of the XYZ program architecture. Identify the types of interfaces between the critical components and other mission related systems that could be affected by corrupt software. What security measures, such as data bus controller, internal/network firewalls, etc. exist to mitigate corruption between interfaces (both from other XYZ functions and from off-board sources)?

f. In the absence of additional risk associated from foreign suppliers, what advantages are there to contracting with these companies for software (cost, technical skills, etc.)?

4. Software Testing
   a. Describe the nature of any software specific testing performed by ABC company for the XYZ program? Is this testing specific to software security (i.e. testing for malicious code) or is it more aimed at quality assurance and/or functionality?
   b. How are software components received from other suppliers tested before integration into the XYZ program? Are there additional tests performed for software code or hardware embedded code provided by outside sources (foreign or domestic)? Does this process differ from hardware components integrated from outside sources? Is this true for reused code (defense provided (DOTS) or ADA library)?
   c. Is the XYZ program required to comply with the DOD Information Technology Security Certification and Accreditation Process (DITSCAP – DOD 5200.40)? If so, what is the current phase of completion for the program? Is there a completed System Security Authorization Agreement (SSAA)? Please provide an overview of the security requirements identified and current plans to verify, validate, and monitor security processes.

5. Determining Foreign Content
   a. Do you track foreign software content for your program? If so, explain the justification for doing so (contracting, export control, program security, configuration control). Is this a government requirement?
   b. Do you require your sub-contractors to track foreign software content on your program? Is this self-initiated or a flowdown from a government requirement?
   c. Do you track foreign content on COTS software for your program? Is it possible and/or feasible to track source and pedigree for all foreign content in COTS software?

6. Provide the following information for each software provider on your program, including sub-contractors:
   a. supplier name;
   b. supplier location (by state/country);
   c. Number of lines of code supplied;
      - on-board
      - off-board
      - COTS
   d. Percent of total lines of code on program;
   e. brief description of application.

Appendix C: Final Role-based Case Study Survey Questions

This appendix contains the questions for three role-based surveys used in the case studies.

Privacy Act Statement, Informed Consent Statement, Instructions, and other Information provided to the Respondent Prior to the Questions:

About this Questionnaire:
This questionnaire is part of research being conducted by Major Ryan Maxon for his Master's thesis from the Air Force Institute of Technology (AFIT). Dr. Dennis Strouble (AFIT/ENV) is Maj Maxon's advisor. Contact information for Dr. Strouble and Maj Maxon is below.

This questionnaire was approved by the <*insert approval info*>.

Participation in this research (or any follow-up research) is voluntary.

Privacy Act Statement:
Authority: We are requesting disclosure of personal information, to include your name and rank. Researchers are authorized to collect personal information on research subjects under The Privacy Act-5 USC 552a, 10 USC 55, 10 USC 8013, 32 CFR 219, 45 CFR Part 46, and EO 9397, November 1943.
Purpose: It is possible that latent risks will not be discovered until some time in the future. The purpose of collecting this information is to aid researchers in locating you at a future date if further disclosures are appropriate.
Routine Uses: Information (including name) may be furnished to Federal, State and local agencies for any uses published by the Air Force in the Federal Register, 52 FR 16431, to include, furtherance of the research involved with this study.
Disclosure: Disclosure of the requested information is voluntary. No adverse action whatsoever will be taken against you, and no privilege will be denied you based on the fact you do not disclose this information. However, your participation in this study may be impacted by a refusal to provide this information.

Informed Consent:
The decision to participate in this research is completely voluntary on your part. No one may coerce or intimidate you into participating in this program. You are participating because you want to. The individuals named above have adequately answered any and all questions you have about this study, your participation, and the procedures involved. The individuals named above will also be available to answer any questions concerning procedures throughout this study. If significant new findings develop during the course of this research, which may relate to your decision to continue participation, you will be informed. You may withdraw this consent at any time and discontinue further

participation in this study without prejudice to your entitlements. Maj Maxon or Dr. Strouble may terminate your participation in this study if they feel this to be in your best interest. Any hardcopies of your completed questionnaire will be stored in a locked safe when not in use by Maj Maxon. Any electronic files containing your responses will be carefully protected and stored on non-publicly accessible computer(s). It is intended that the only person having the ability to know which response came from a specific respondent is <*insert unit POC name*>. When no longer needed for research purposes, your responses will be destroyed in a secure manner (shredding) or deleted if electronic. If you have any questions or concerns about your participation in this study or your rights as a research subject, please contact Dr. Dennis Strouble.

Contact information:
Maj Ryan Maxon: <*contact info was inserted here*>
Dr. Dennis Strouble: DSN 785-3636 x3323 or dstroubl@afit.edu.

Questionnaire Instructions:

- Please be open, honest, and candid with your responses. Even if the question is a yes/no type of question, you may answer with more than just a yes/no. This is a qualitative (not quantitative) study, so descriptive or lengthy answers are acceptable.
- Three questionnaires have been developed: one each tailored to a program's (1) management, (2) engineering, and (3) security management staff. Please do not request assistance in answering these questions from staff members outside your area. Answer each question from your area's point-of-view.
- If you do not know the answer or are unable to answer without doing a lot of research (or if the response would be classified), please state so.
- If any portions of your responses are FOUO or otherwise restricted, please mark it as such so that Maj Maxon may take appropriate measures to protect the information.
- Please do not include any classified information in your responses.
- Please answer the questions in the order they are presented – please do not read or skip ahead.

Please e-mail the completed questionnaire back to Maj Maxon at ryan.maxon@afit.edu. (This e-mail account is on an official, AF-controlled network.) **You may also e-mail the completed questionnaire back to <*insert unit POC name*> (<*insert unit POC e-mail address*>)**; He/she will remove your e-mail header info before the responses are forwarded to Maj Maxon.)

Maj Maxon **will not** be using any personally identifiable information from your e-mail header in this research. In the published thesis, your responses ***will not*** be attributable directly to you. The responses, combined with documentation reviews and other research methods, will be described in a manner similar to the following: "The research showed that four of five programs…". Your responses may also be used verbatim, but no name or other identifier will be associated with the response.

Please complete the questionnaire as soon as possible. *<desired response suspense was included here>*.

Respondent Number: (To be filled in by Maj Maxon)

Program, IT, or Weapon System Name: _____(Respondent fill in)_____
Mission Assurance Category (MAC): _____(Respondent fill in)_____

Connectivity (Circle or mark all that apply):
Hosted on a non-.mil (ex: outsourced to a .com) _____
NIPRNET (Internal .mil only) _____
NIPRNET & Internet accessible (CAC, userid/password, public, etc) _____
SIPRNET _____
JWICS _____
Other _____

Prime Contractor (Company Name): _____(Respondent fill in)_____
Sub-Contractors (Company Names): _____(Respondent fill in)_____

If you're involved with more than one program or system, please list them, and provide the information above for each system. You may copy/paste the above as many times as needed. If your responses to the questions below vary depending upon the system, in the space after the question, please identify each system and the response for each.

In the following questions, "ABC company" refers to the Prime or sub-contractors that work within your weapon system program. Also, "XYZ program" refers to your program.

Questions for the Program Manager:

1. Without accessing any outside references (like google.com), in your own words what does the term "Software Assurance" mean to you?

2. Overall, what is your level of confidence that software (software in general, not just your program's software) will function as intended and be free of flaws, security vulnerabilities and malicious code? (Bold your response or delete all but your response)
   - High (based on objective evidence, convinced that there are no flaws, security vulnerabilities or malicious code
   - Medium (based on objective evidence, reasonably sure there are no flaws, security vulnerabilities or malicious code)

- Low (based on objective evidence, flaws, security vulnerabilities and/or malicious code are likely.)
- Don't know/unsure

3. Overall, how would you rate the fundamental security of software (software in general, not just your program's software) WITHOUT the use of firewalls, Intrusion Detection Systems (IDS), anti-virus scanners, etc.? (Bold your response or delete all but your response)
   - Extremely vulnerable
   - Vulnerable
   - Somewhat secure
   - Secure
   - Highly secure
   - Don't know/unsure

4. Of the following, which software attributes are most important to your organization? (choose up to 3) (Delete those that are not your response)
   - Reliable software that functions as promised
   - Software that is free from security vulnerabilities and malicious code
   - Ease of integration and configuration
   - Software that conforms to requirements and industry standards
   - Convenience and ease of use
   - Rich feature set
   - Other (please write-in)
   - Don't know/unsure

5. To what degree, if any, is foreign involvement considered an additional risk factor in your program's software development or software acquisition?

6. What process was used to determine program security risks?

7. How often are your program's security risks re-evaluated?

8. Please describe the security practices identified (in questions 5-7) above. Are they different from standard DOD security practices? If so, why?

9. Does your program have a Program Protection Plan (DoD 5200.1-M)?

10. As defined by DoD, Software Assurance is the *level of confidence* that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software.

    a) For the XYZ program, on a scale of one (low confidence) to seven (high confidence), what is your level of confidence in the system's software?

b) Do you agree with the DoD definition for Software Assurance? If no, why?

11. For the XYZ program, were the software assurance or software security practices of the bidding contractors a factor in the source selection process? For the bidding contractors, were any other supplier trustworthiness considerations used during source selection?

12. Are you able to determine what portions of your software have been developed by foreign entities, either foreign suppliers or foreign nationals working for U.S. suppliers? If so, are foreign suppliers or nationals responsible for significant levels of software content or for software development in critical system areas?

13. Has ABC company, or its U.S. suppliers, hired any foreign developers to work on XYZ program software at U.S. facilities?

14. Is there any non-"flight software" (support software that interacts with onboard systems to provide information in support of operational activity) on the XYZ program that was developed overseas, or in the U.S. by foreign nationals?

15. If XYZ program software is supplied 100% by U.S. software suppliers, …
    a) Why was the decision made to use only U.S. software suppliers? Was security a factor in this decision?

    b) Is there a specific requirement for U.S. only software development on your program? If so, why, and how is this requirement documented and enforced? Who is levying this requirement?

16. What processes are in place to manage security risks for software development? Would any of these techniques be considered above and beyond those normally required for developing software? Are any of these specifically targeted toward mitigating risks associated with foreign developed software?

17. Who implements the security processes (reference previous question)? Prime contractor? Sub contractors? Program office?

18. Does your program have any processes in place that would mitigate the risks associated with malicious code inserted by contractors?

19. Does the program office have access to the system's source code?

20. Which of the following, if any, negative outcomes has your organization experienced as a result of flaws, security vulnerabilities or malicious code in software? (Delete those that do not apply):

- Had to redeploy IT staff to deal with the problem(s)
- Increased IT costs
- Reduced IT productivity
- Delayed projects
- Unanticipated patch management costs and related business disruption
- Reduced end-user productivity
- Didn't implement desired features or functions
- Adverse impact on our customers (negative perception of us by customers or impact on our customer's ability to do business with us)
- Limited the benefits IT could deliver to internal clients
- None, we have had no negative experiences
- Hindered ability to comply with regulations
- Other (please write-in)
- Don't know/unsure

21. How is the software security posture documented? (System Security Authorization Agreement (SSAA), DIACAP package or Other (please specify))

22. Is the potential for foreign influence in the system's software life cycle addressed in the system's certification and accreditation documentation?

23. Which laws/policy/guidance govern software security or the use of foreign software/foreign developers on your program?

_____ITAR _____ DOD 5000 _____DODD 8500 ____NISPOM

Other (please list): _____

24. Do you limit the use of foreign developed software or COTS, either contractually or in any other manner?

25. Briefly explain the processes your program uses to ensure compliance with these laws/policies both within the program office and with contractors?

26. With respect to the potential for an adversary to insert malicious logic, during what phase of the software life cycle for your weapon system do you feel is the weakest?

27. Do you have any processes in place to verify that contractors are following relevant rules/regulations? What are these processes?

28. Does your program use specific provisions to limit or constrain the use of foreign nationals or foreign firms for developing software? If so, please describe briefly.

29. Are your contractors required to report or notify DOD or the program office regarding the use of foreign firms or employees for software development?

30. Is COTS software allowed on XYZ program? If yes, …
    a. What secure software development education, training, and/or certification do the companies that provide your COTS require of their software developers and/or software project managers?

       If none is required or if you do not know, do you know if they plan on requiring any education, training, or certification (such as the new SANS Secure Software Programmers Certification) in the near future?

    b. Do you take a supplier's software security education, training, or certification requirements or programs into account when selecting COTS?

31. Is Open Source software allowed on XYZ program?

32. Is any internally developed (custom) code used on XYZ program? If yes, …
    a. What secure software development education, training, and/or certification do you provide or require of your internal software developers and/or software project managers?

       If none is required or if you do not know, do you plan on requiring or providing any education, training, or certification (such as the new SANS Secure Software Programmers Certification) in the near future?

    b. Do you (or your contractors) take a prospective employee's software security education, training, or certification into account when making hiring decisions?

33. Is any legacy code used on XYZ program?

34. Does XYZ program use software algorithms from a software library?

35. Does XYZ program use software of unknown pedigree or provenance?

36. Are there any restrictions on developers using code from open sources on the Internet? If yes, how is this enforced or monitored?

37. With respect to XYZ program, how do you rate the risk of foreign influence on weapon system software?

38. With respect to AF weapon system programs in general, how do you rate the risk of foreign influence on weapon system software?

39. Of the following potential actions that a software vendor could take, which would be most effective in reassuring you that the vendor's products will operate as intended and will be free from flaws, security vulnerabilities and malicious code? (check up to three) (Delete those that are not your response)
    - o Certify that software meets a designated security target
    - o Provide evidence that software has been scanned for flaws and security vulnerabilities using qualified tools
    - o Provide a list of known flaws and security vulnerabilities
    - o Provide test cases for their software
    - o Provide more transparency or visibility into their development process
    - o Provide information concerning the provenance of the code (e.g. where it was built, who it was acquired from, documentation of individuals who contributed to the code, etc.)
    - o Other (please write-in)
    - o Don't know/unsure
    - o No additional information or actions by software vendors are needed

40. As a program manager, do you feel you are adequately informed by the Prime contractor of the risks involved with your system's software?

41. Do you feel that your program has been provided with adequate resources to address the system's software assurance problem?

42. Have you or your staff received a threat briefing regarding the capabilities and intent of adversaries to attack and subvert DoD systems and/or networks through supply chain exploitations or through other sophisticated techniques?

    If yes, did you find the briefing useful?

    From who or where did you receive this briefing?

    How long ago did you receive the briefing?

    If you have not received this type of briefing, do you think this type of briefing would be useful to you or your staff?

43. Reflecting upon the issues and ideas that these questions may have brought up regarding your system's software assurance, is your answer to question 7 still the same?

    If no, please elaborate on which issue(s) changed your mind.

44. Are there any additional software assurance related incidents, anecdotes, comments, concerns, etc. not addressed above that you would like to relate, address, or add to?

Please provide a copy of your system's software and hardware list, Ports/Protocols/Services (PPS) list, and connectivity diagram from the SSAA or DIACAP package along with your response. You may also provide a copy of any other documents that you feel would be relevant to software security within your program, such as a configuration management plan.

Questions for the System/Software Engineer:

1. Without accessing any outside references (like google.com), in your own words what does the term "Software Assurance" mean to you?

2. Overall, what is your level of confidence that software (software in general, not just your program's software) will function as intended and be free of flaws, security vulnerabilities and malicious code? (Bold your response or delete all but your response)
   - o High (based on objective evidence, convinced that there are no flaws, security vulnerabilities or malicious code
   - o Medium (based on objective evidence, reasonably sure there are no flaws, security vulnerabilities or malicious code)
   - o Low (based on objective evidence, flaws, security vulnerabilities and/or malicious code are likely.)
   - o Don't know/unsure

3. Overall, how would you rate the fundamental security of software (software in general, not just your program's software) WITHOUT the use of firewalls, Intrusion Detection Systems (IDS), anti-virus scanners, etc.? (Bold your response or delete all but your response)
   - o Extremely vulnerable
   - o Vulnerable
   - o Somewhat secure
   - o Secure
   - o Highly secure
   - o Don't know/unsure

4. Of the following, which software attributes are most important to your organization? (choose up to 3) (Delete those that are not your response)
   - o Reliable software that functions as promised
   - o Software that is free from security vulnerabilities and malicious code
   - o Ease of integration and configuration
   - o Software that conforms to requirements and industry standards
   - o Convenience and ease of use

- Rich feature set
- Other (please write-in)
- Don't know/unsure

5. To what degree, if any, is foreign involvement considered an additional risk factor in your program's software development or software acquisition?

6. What processes are in place to manage security risks for software development?

   Would any of these techniques be considered above and beyond those normally required for developing software?

   Are any of these specifically targeted toward mitigating risks associated with foreign developed software?

7. Who implements the security processes (reference previous question)? Prime contractor? Sub contractors? Program office?

8. Does your program have any processes in place that would mitigate the risks associated with malicious code inserted by contractors?

9. As defined by DoD, Software Assurance is the *level of confidence* that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software.

   a) For the XYZ program, on a scale of one (low confidence) to seven (high confidence), what is your level of confidence in the system's software?

   b) Do you agree with the DoD definition for Software Assurance? If no, why?

10. Does your program utilize source code analysis tools to detect programming defects, including vulnerabilities, in the software?

    If yes, which applications are used?

    Who runs the analysis software (program office, prime contractor, sub-contractors)?

    Who are the results of the scans provided to?

11. Of the following sources of information and/or methods, which do you routinely use to determine if software is free from flaws, security vulnerabilities and malicious code? (check all that apply) (Delete those that are not your response)
    - Internal testing
    - Contract requirements and/or service level agreements (SLAs)

- Reputation of the vendor and product among senior IT executives in other organizations
- Third-party testing
- Vendor statements or assurance claims
- Comprehensive review of source code
- Common Criteria evaluation (international evaluation standard)
- Not applicable
- Other (please write-in)

12. Does the program office have access to the system's source code?

13. Which of the following, if any, negative outcomes has your organization experienced as a result of flaws, security vulnerabilities or malicious code in software? (Delete those that do not apply):
    - Had to redeploy IT staff to deal with the problem(s)
    - Increased IT costs
    - Reduced IT productivity
    - Delayed projects
    - Unanticipated patch management costs and related business disruption
    - Reduced end-user productivity
    - Didn't implement desired features or functions
    - Adverse impact on our customers (negative perception of us by customers or impact on our customer's ability to do business with us)
    - Limited the benefits IT could deliver to internal clients
    - None, we have had no negative experiences
    - Hindered ability to comply with regulations
    - Other (please write-in)
    - Don't know/unsure

14. Has malicious code or security vulnerabilities been identified in your system's code? If yes, please answer the following:

    a) Were the vulnerabilities found by specific mechanisms your program has in place?

       If yes, what mechanisms?

       If not through specific mechanisms, how were they detected?

    b) Was the source of the vulnerability traceable?

    c) In what phase of the software life cycle did the vulnerability enter?

    d) Was the vulnerability in code supplied by foreign suppliers or U.S. suppliers?

e) Was the malicious code or vulnerability in COTS, Open Source, GOTS, or contractor-developed code?

15. With respect to the potential for an adversary to insert malicious logic, during what phase of the software life cycle for your weapon system do you feel is the weakest?

16. Is COTS software allowed on XYZ program? If yes, …

    a. How is the software vetted to ensure it is free of vulnerabilities?

    b. What process is used to ensure that updates to the original sources are incorporated into your system?

17. Is Open Source software allowed on XYZ program? If yes, …

    a. What sources are used?

    b. How is the software vetted to ensure it is free of vulnerabilities?

    c. What process is used to ensure that updates to the original sources are incorporated into your system?

18. Is any internally developed (custom) code used on XYZ program? If yes, …

    a. Who is it written by? Mark all that apply:

        __ Government civilians ___ Active duty military

        __ Contractors (e.g., CMEs, prime contractor, sub-contractors)

        __ Other (please explain)

    b. How is the software vetted to ensure it is free of vulnerabilities?

    c. What secure software development education, training, and/or certification do you provide or require of your internal software developers and/or software project managers?

        If none is required or if you do not know, do you plan on requiring or providing any education, training, or certification (such as the new SANS Secure Software Programmers Certification) in the near future?

d. Do you (or your contractors) take a prospective employee's software security education, training, or certification into account when making hiring decisions?

19. Is any legacy code used on XYZ program? If yes, …

    a. Do you know the origin of this code?

    b. How is the software vetted to ensure it is free of vulnerabilities?

    c. What process is used to ensure that updates to the original sources are incorporated into your system?

20. Does XYZ program use software algorithms from a software library?

    a. Do you know the origin of these libraries?

    b. How are the libraries vetted to ensure they are free of vulnerabilities?

    c. What process is used to ensure that updates to the original sources are incorporated into your system?

21. Does XYZ program use software of unknown pedigree or provenance?

    a. How is the software vetted to ensure it is free of vulnerabilities?

    b. What process is used to ensure that updates to the original sources are incorporated into your system?

22. Are there any restrictions on developers using code from open sources on the Internet?

    If yes, how is this enforced or monitored?

23. With respect to XYZ program, how do you rate the risk of foreign influence on weapon system software?

24. With respect to AF weapon system programs in general, how do you rate the risk of foreign influence on weapon system software?

25. Of the following potential actions that a software vendor could take, which would be most effective in reassuring you that the vendor's products will operate as intended and will be free from flaws, security vulnerabilities and malicious code? (check up to three) (Delete those that are not your response)

- Certify that software meets a designated security target
- Provide evidence that software has been scanned for flaws and security vulnerabilities using qualified tools
- Provide a list of known flaws and security vulnerabilities
- Provide test cases for their software
- Provide more transparency or visibility into their development process
- Provide information concerning the provenance of the code (e.g. where it was built, who it was acquired from, documentation of individuals who contributed to the code, etc.)
- Other (please write-in)
- Don't know/unsure
- No additional information or actions by software vendors are needed

26. Do you feel that your program has been provided with adequate resources to address the system's software assurance problem?

27. Have you or your staff received a threat briefing regarding the capabilities and intent of adversaries to attack and subvert DoD systems and/or networks through supply chain exploitations or through other sophisticated techniques?

    If yes, did you find the briefing useful?

    From who or where did you receive this briefing?

    How long ago did you receive the briefing?

    If you have not received this type of briefing, do you think this type of briefing would be useful to you or your staff?

28. Do you have any recommendations on how the AF might improve the software assurance of its weapon systems?

29. Reflecting upon the issues and ideas that these questions may have brought up regarding your system's software assurance, is your answer to question 6 still the same?

    If no, please elaborate on which issue(s) changed your mind.

30. If the AF could provide your office with code analysis tools (software and licenses), installation assistance, and the training on how to use them, would your program office want them?

31. Are there any additional software assurance related incidents, anecdotes, comments, concerns, etc. not addressed above that you would like to relate, address, or add to?

Questions for the Security Manager:

1. Without accessing any outside references (like google.com), in your own words what does the term "Software Assurance" mean to you?

2. Overall, what is your level of confidence that software (software in general, not just your program's software) will function as intended and be free of flaws, security vulnerabilities and malicious code? (Bold your response or delete all but your response)
   - o High (based on objective evidence, convinced that there are no flaws, security vulnerabilities or malicious code
   - o Medium (based on objective evidence, reasonably sure there are no flaws, security vulnerabilities or malicious code)
   - o Low (based on objective evidence, flaws, security vulnerabilities and/or malicious code are likely.)
   - o Don't know/unsure

3. Overall, how would you rate the fundamental security of software (software in general, not just your program's software) WITHOUT the use of firewalls, Intrusion Detection Systems (IDS), anti-virus scanners, etc.? (Bold your response or delete all but your response)
   - o Extremely vulnerable
   - o Vulnerable
   - o Somewhat secure
   - o Secure
   - o Highly secure
   - o Don't know/unsure

4. Of the following, which software attributes are most important to your organization? (choose up to 3) (Delete those that are not your response)
   - o Reliable software that functions as promised
   - o Software that is free from security vulnerabilities and malicious code
   - o Ease of integration and configuration
   - o Software that conforms to requirements and industry standards
   - o Convenience and ease of use
   - o Rich feature set
   - o Other (please write-in)
   - o Don't know/unsure

5. To what degree, if any, is foreign involvement considered an additional risk factor in your program's software development or software acquisition?

6. What process was used to determine program security risks?

7. How often are your program's security risks re-evaluated?

8. Please describe the security practices identified (in questions 5-7) above. Are they different from standard DOD security practices? If so, why?

9. As defined by DoD, Software Assurance is the level of confidence that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software.

    a) For the XYZ program, on a scale of one (low confidence) to seven (high confidence), what is your level of confidence in the system's software?

    b) Do you agree with the DoD definition for Software Assurance? If no, why?

10. Of the following sources of information and/or methods, which do you routinely use to determine if software is free from flaws, security vulnerabilities and malicious code? (check all that apply) (Delete those that are not your response)
    - Internal testing
    - Contract requirements and/or service level agreements (SLAs)
    - Reputation of the vendor and product among senior IT executives in other organizations
    - Third-party testing
    - Vendor statements or assurance claims
    - Comprehensive review of source code
    - Common Criteria evaluation (international evaluation standard)
    - Not applicable
    - Other (please write-in)

11. Are you able to determine what portions of your software have been developed by foreign entities, either foreign suppliers or foreign nationals working for U.S. suppliers?

    If so, are foreign suppliers or nationals responsible for significant levels of software content or for software development in critical system areas?

12. Has ABC company, or its U.S. suppliers, hired any foreign developers to work on XYZ program software at U.S. facilities?

13. Is there any non-"flight software" (support software that interacts with onboard systems to provide information in support of operational activity) on the XYZ program that was developed overseas, or in the U.S. by foreign nationals?

14. How is the software security posture documented? (System Security Authorization Agreement (SSAA), DIACAP package or Other (please specify))

15. Is the potential for foreign influence in the system's software life cycle addressed in the system's certification and accreditation documentation?

16. Do you have any processes in place to verify that contractors are following relevant rules/regulations? What are these processes?

17. What secure software development education, training, and/or certification do you provide or require of your internal software developers and/or software project managers?

    If none is required or if you do not know, do you plan on requiring or providing any education, training, or certification (such as the new SANS Secure Software Programmers Certification) in the near future?

18. Do you (or your contractors) take a prospective employee's software security education, training, or certification into account when making hiring decisions?

19. With respect to XYZ program, how do you rate the risk of foreign influence on weapon system software?

20. With respect to AF weapon system programs in general, how do you rate the risk of foreign influence on weapon system software?

21. Of the following potential actions that a software vendor could take, which would be most effective in reassuring you that the vendor's products will operate as intended and will be free from flaws, security vulnerabilities and malicious code? (check up to three) (Delete those that are not your response)
    - Certify that software meets a designated security target
    - Provide evidence that software has been scanned for flaws and security vulnerabilities using qualified tools
    - Provide a list of known flaws and security vulnerabilities
    - Provide test cases for their software
    - Provide more transparency or visibility into their development process
    - Provide information concerning the provenance of the code (e.g. where it was built, who it was acquired from, documentation of individuals who contributed to the code, etc.)
    - Other (please write-in)
    - Don't know/unsure
    - No additional information or actions by software vendors are needed

22. Do you feel that your program has been provided with adequate resources to address the system's software assurance problem?

23. Have you or your staff received a threat briefing regarding the capabilities and intent of adversaries to attack and subvert DoD systems and/or networks through supply chain exploitations or through other sophisticated techniques?

    If yes, did you find the briefing useful?

    From who or where did you receive this briefing?

    How long ago did you receive the briefing?

    If you have not received this type of briefing, do you think this type of briefing would be useful to you or your staff?

24. Do you have any recommendations on how the AF might improve the software assurance of its weapon systems?

25. Reflecting upon the issues and ideas that these questions may have brought up regarding your system's software assurance, is your answer to question 6 still the same?

    If no, please elaborate on which issue(s) changed your mind.

26. Are there any additional software assurance related incidents, anecdotes, comments, concerns, etc. not addressed above that you would like to relate, address, or add to?

Appendix D: Content Analysis Coding Table

Table 1. Content analysis coding procedure table

Document Type/Title:	
Total # of Findings	**Best Practices**
	Training & Education
	Improve College-level courses
	Train Developers
	Train Managers
	Train software architects/designers
	Train software testers
	Train quality assurance personnel
	Train documentation personnel
	Other:
	Build Security In
	Use structured development model/method
	Define Security Requirements Early
	Design for security within application architecture
	Develop Code with security built in; make conscious effort to avoid common errors like buffer overflows
	Test and review code for defects
	- Manually inspect or use code analysis tools throughout development
	- Manually inspect or use code analysis tools at specific checkpoints, milestones, or "touchpoints"
	- Use other robust testing methods (penetration testing, "fuzzing", etc.)
	- Independently inspect/review or test code (Employ third-party)
	Other:
	Security Readiness, Documentation, & Response
	Assess/document known security risks and gaps
	Develop security configuration and policy guide
	Document dependencies and module functions
	Ensure post-deployment incident response measures are in-place
	Other:
	Code Pedigree/Provenance
	Employ strict source code change management techniques
	Secure supply chain… protect against:
	- tampering of physical media or electronic distribution methods
	- threats posed by internally developed code
	- threats posed by closed source code (COTS, Shareware, Freeware)
	- threats posed by open source code

Document Type/Title:	
Total # of Findings	**Best Practices**
	- threats posed by legacy source code
	- threats posed by outsourced code
	- threats posed by offshore code
	- threats posed by mixed code
	Ensure sample source code is secure
	Require vendors to disclose usage of third-party software
	Other:
	Research, Evangelism, & Management Involvement
	Conduct secure software research
	Stay current on secure software research
	Share lessons learned
	Leaders should make clear what priority security has in the organization
	Managers should support security objectives
	Other:
	Malicious Insiders
	Employ mechanisms to mitigate threat/impact of malicious insiders
	Other:
	Employee Quality
	Evaluate/Test secure software development skills or knowledge level of current developers
	Evaluate/Test secure software development skills or knowledge level of prospective new employees
	Other:
	Supplier Security Controls
	Assess supplier's security controls
	- Include software assurance requirements in statement of work, contracts
	Track supplier performance over time
	Other:
	Other Best Practices (list)

Appendix E: Survey Questions Mapped to Best Practices

The following is a list of the software assurance best practices identified in Chapter II of this thesis. Letters have been assigned to the main best practices and numbers have been assigned to the subordinate best practices. The letters/numbers are used in right-hand columns of the table below to represent the corresponding best practice.

Letter/Number = Best Practices

A = **Training & Education**
A1 = Improve College-level courses
A2 = Train Developers
A3 = Train Managers
A4 = Train software architects/designers
A5 = Train software testers
A6 = Train quality assurance personnel
A7 = Train documentation personnel
B = **Build Security In**
B1 = Use structured development model/method
B2 = Define Security Requirements Early
B3 = Design for security within application architecture
B4 = Develop Code with security built in; make conscious effort to avoid common errors like buffer overflows
B5 = Use code analysis (or other secure coding test) tools throughout development
B6 = Use code analysis (or other secure coding test) tools at specific checkpoints, milestones, or "touchpoints"
B7 = Manually inspect code throughout development
B8 = Manually inspect code at specific checkpoints, milestones, or "touchpoints"
B9 = Use other robust testing methods (penetration testing, "fuzzing", etc.)
B10 = Employ third-party to inspect or test code
C = **Security Readiness, Documentation, & Response**
C1 = Assess/document known security risks and gaps
C2 = Develop security configuration guide
C3 = Document dependencies
C4 = Ensure post-deployment incident response measures are in-place
D = **Code Pedigree/Provenance**
D1 = Employ strict source code change management techniques
D2 = Secure supply chain… protect against:
D3 = - tampering of physical media or electronic distribution methods
D4 = - threats posed by internally developed code
D5 = - threats posed by closed source code (COTS, Shareware, Freeware)
D6 = - threats posed by open source code
D7 = - threats posed by legacy source code
D8 = - threats posed by outsourced code
D9 = - threats posed by offshore code
D10 = - threats posed by mixed code
D11 = Ensure sample source code is secure
D12 = Require vendors to disclose usage of third-party software

E = **Research, Evangelism, & Management Involvement**
E1 = Conduct secure software research
E2 = Stay current on secure software research
E3 = Share lessons learned
E4 = Leaders should make clear what priority security has in the organization
E5 = Managers should support security objectives
F = **Malicious Insiders**
F1 = Employ mechanisms to mitigate threat/impact of malicious insiders
G = **Employee Quality**
G1 = Test current developers
G2 = Evaluate secure software development skills or knowledge level of prosepective new employees
H = **Supplier Security Controls**
H1 = Assess supplier's security controls
H2 = - Include software assurance requirements in statement of work, contracts
H3 = Track supplier performance over time

Table 2. Survey Questions Logically Linked to Best Practices

Source	Question	Maps to Best Practice #	
		Primary	Secondary
Thesis Author	1. Without accessing any outside references (like google.com), in your own words what does the term "Software Assurance" mean to you?	A3	
CIO Exec	2. Overall, what is your level of confidence that software (software in general, not just your program's software) will function as intended and be free of flaws, security vulnerabilities and malicious code? (Bold your response or delete all but your response) o High (based on objective evidence, convinced that there are no flaws, security vulnerabilities or malicious code o Medium (based on objective evidence, reasonably sure there are no flaws, security vulnerabilities or malicious code) o Low (based on objective evidence, flaws, security vulnerabilities and/or malicious code are likely.) o Don't know/unsure	B, A, E	
CIO Exec	3. Overall, how would you rate the fundamental security of software (software in general, not just your program's software) WITHOUT the use of firewalls, Intrusion Detection Systems (IDS), anti-virus scanners, etc.? (Bold your response or delete all but your response) o Extremely vulnerable o Vulnerable o Somewhat secure o Secure o Highly secure o Don't know/unsure	B, A, E	

Source	Question	Maps to Best Practice #	
		Primary	Secondary
CIO Exec	4. Of the following, which software attributes are most important to your organization? (choose up to 3) (Delete those that are not your response) o Reliable software that functions as promised o Software that is free from security vulnerabilities and malicious code o Ease of integration and configuration o Software that conforms to requirements and industry standards o Convenience and ease of use o Rich feature set o Other (please write-in) o Don't know/unsure	E4 B4, B B1, B	
GAO	~~What security risks have been identified on your program?~~ Removed. Response may be classified or too sensitive.		
GAO	5. To what degree, if any, is foreign involvement considered an additional risk factor in these areas? Changed to: To what degree, if any, is foreign involvement considered an additional risk factor in your program's software development or software acquisition?	 D9	
GAO	6. Were studies or reviews conducted to determine program security risks? Changed to: What process was used to determine program security risks?	 C1	 B5-10, H1
GAO	~~In the process used to determine your program's security risks, what potential risks were determined not to be a risk?~~ Removed. Response may be classified or too sensitive.		
Thesis Author	7. How often are your program's security risks re-evaluated?	C1	B5-10, H1
GAO	8. Please describe the security practices identified (in questions 5-7) above. Are they different from standard DOD security practices? If so, why?		
GAO	9. Does your program have a Program Protection Plan (DoD 5200.1-M)?	C1	

Source	Question	Maps to Best Practice #	
		Primary	Secondary
Thesis Author	10. As defined by DoD, Software Assurance is the *level of confidence* that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software. a) For the XYZ program, on a scale of one (low confidence) to seven (high confidence), what is your *level of confidence* in the system's software? b) Do you agree with the DoD definition for Software Assurance? If no, why?		
Thesis Author	11. For the XYZ program, were the software assurance or software security practices of the bidding contractors a factor in the source selection process? For the bidding contractors, were any other supplier trustworthiness considerations used during source selection?	H1	D
GAO	12. Are you able to determine what portions of your software have been developed by foreign entities, either foreign suppliers or foreign nationals working for U.S. suppliers? If so, are foreign suppliers or nationals responsible for significant levels of software content or for software development in critical system areas?	D9	H1
GAO	13. Has ABC company, or its U.S. suppliers, hired any foreign developers to work on XYZ program software at U.S. facilities?	D9	F1
GAO	14. Is there any non-"flight software" (support software that interacts with onboard systems to provide information in support of operational activity) on the XYZ program that was developed overseas, or in the U.S. by foreign nationals?	D9	F1
GAO	15. If XYZ program software is supplied 100% by U.S. software suppliers, … a) Why was the decision made to use only U.S. software suppliers? Was security a factor in this decision? b) Is there a specific requirement for U.S. only software development on your program? If so, why, and how is this requirement documented and enforced? Who is levying this requirement?	D D	H H
GAO	16. What processes are in place to manage security risks for software development? Would any of these techniques be considered above and beyond those normally required for developing software? Are any of these specifically targeted toward mitigating risks associated with foreign developed software?	B1	D9

Source	Question	Maps to Best Practice #	
		Primary	Secondary
GAO	17. Who implements the security processes (reference previous question)? Prime contractor? Sub contractors? Program office?	F	A, D, B
GAO	18. Does your program have any processes in place that would mitigate the risks associated with malicious code inserted by contractors?	D8	
Thesis Author	19. Does your program utilize source code analysis tools to detect programming defects, including vulnerabilities, in the software? If yes, which applications are used? Who runs the analysis software (program office, prime contractor, sub-contractors)? Who are the results of the scans provided to?	B5, B6, B9	
CIO Exec	20. Of the following sources of information and/or methods, which do you routinely use to determine if software is free from flaws, security vulnerabilities and malicious code? (check all that apply) (Delete those that are not your response) o Internal testing o Contract requirements and/or service level agreements (SLAs) o Reputation of the vendor and product among senior IT executives in other organizations o Third-party testing o Vendor statements or assurance claims o Comprehensive review of source code o Common Criteria evaluation (international evaluation standard) o Not applicable o Other (please write-in)	B5-9 H2 H3 B10 H B5-10 H	 D12 B, D
Thesis Author	21. Does the program office have access to the system's source code?	H1	B5-10
Thesis Author	22. Has malicious code or security vulnerabilities been identified in your system's code? If yes, please answer the following: a) Were the vulnerabilities found by specific mechanisms your program has in place? If yes, what mechanisms? If not through specific mechanisms, how were they detected? b) Was the source of the vulnerability traceable? c) In what phase of the software life cycle did the vulnerability enter? d) Was the vulnerability in code supplied by foreign suppliers or U.S. suppliers? e) Was the malicious code or vulnerability in COTS, Open Source, GOTS, or contractor-developed code?	C4 B5-10 D1 B1 D8,9 D	

Source	Question	Maps to Best Practice #	
		Primary	Secondary
CIO Exec	23. Which of the following, if any, negative outcomes has your organization experienced as a result of flaws, security vulnerabilities or malicious code in software? (Delete those that do not apply): o Had to redeploy IT staff to deal with the problem(s) o Increased IT costs o Reduced IT productivity o Delayed projects o Unanticipated patch management costs and related business disruption o Reduced end-user productivity o Didn't implement desired features or functions o Adverse impact on our customers (negative perception of us by customers or impact on our customer's ability to do business with us) o Limited the benefits IT could deliver to internal clients o None, we have had no negative experiences o Hindered ability to comply with regulations o Other (please write-in) o Don't know/unsure		
Thesis Author	24. How is the software security posture documented? (System Security Authorization Agreement (SSAA), DIACAP package or Other (please specify))	C1-3	
Thesis Author	25. Is the potential for foreign influence in the system's software life cycle addressed in the system's certification and accreditation documentation?	C1	D9
GAO	26. Which laws/policy/guidance govern software security or the use of foreign software/foreign developers on your program? ___ITAR ___ DOD 5000 ___DODD 8500 __NISPOM Other (please list): _____		
GAO	27. Do you limit the use of foreign developed software or COTS, either contractually or in any other manner?	D9	
GAO	28. Briefly explain the processes your program uses to ensure compliance with these laws/policies both within the program office and with contractors?	D8	F1
Thesis Author	29. With respect to the potential for an adversary to insert malicious logic, during what phase of the software life cycle for your weapon system do you feel is the weakest?	B1	
GAO	30. Do you have any processes in place to verify that contractors are following relevant rules/regulations? What are these processes?	H1	

Source	Question	Maps to Best Practice #	
		Primary	Secondary
GAO	31. Does your program use specific provisions to limit or constrain the use of foreign nationals or foreign firms for developing software? If so, please describe briefly.	D9	
GAO	32. Are your contractors required to report or notify DOD or the program office regarding the use of foreign firms or employees for software development?	D9	H1
GAO (main question only) & Thesis Author	33. Is COTS software allowed on XYZ program? If yes, … a. How is the software vetted to ensure it is free of vulnerabilities? b. What process is used to ensure that updates to the original sources are incorporated into your system? c. What secure software development education, training, and/or certification do the companies that provide your COTS require of their software developers and/or software project managers? If none is required or if you do not know, do you know if they plan on requiring any education, training, or certification (such as the new SANS Secure Software Programmers Certification) in the near future? d. Do you take a supplier's software security education, training, or certification requirements or programs into account when selecting COTS?	B5-10 D1 A G G	B
Thesis Author	34. Is Open Source software allowed on XYZ program? If yes, a. What sources are used? b. How is the software vetted to ensure it is free of vulnerabilities? c. What process is used to ensure that updates to the original sources are incorporated into your system?	D6 D1	B

Source	Question	Maps to Best Practice #	
		Primary	Secondary
Thesis Author	35. Is any internally developed (custom) code used on XYZ program? If yes, … a. Who is it written by? Mark all that apply: __ Government civilians __ Active duty military __ Contractors (e.g., CMEs, prime contractor, sub-contractors) __ Other (please explain) b. How is the software vetted to ensure it is free of vulnerabilities? c. What secure software development education, training, and/or certification do you provide or require of your internal software developers and/or software project managers? If none is required or if you do not know, do you plan on requiring or providing any education, training, or certification (such as the new SANS Secure Software Programmers Certification) in the near future? d. Do you (or your contractors) take a prospective employee's software security education, training, or certification into account when making hiring decisions?	D4 A A G2	B G1
GAO (main question only) & Thesis Author	36. Is any legacy code used on XYZ program? If yes, … a. Do you know the origin of this code? b. How is the software vetted to ensure it is free of vulnerabilities? c. What process is used to ensure that updates to the original sources are incorporated into your system?	D7 D1	B
GAO (main question only) & Thesis Author	37. Does XYZ program use software algorithms from a software library? a. Do you know the origin of these libraries? b. How are the libraries vetted to ensure they are free of vulnerabilities? c. What process is used to ensure that updates to the original sources are incorporated into your system?	D D1	B

Source	Question	Maps to Best Practice #	
		Primary	Secondary
GAO (main question only) & Thesis Author	38. Does XYZ program use software of unknown pedigree or provenance? a. How is the software vetted to ensure it is free of vulnerabilities? b. What process is used to ensure that updates to the original sources are incorporated into your system?	D D1	B
Thesis Author	39. Are there any restrictions on developers using code from open sources on the Internet? If yes, how is this enforced or monitored?	D6	B
Thesis Author	40. With respect to XYZ program, how do you rate the risk of foreign influence on weapon system software?	D9	
Thesis Author	41. With respect to AF weapon system programs in general, how do you rate the risk of foreign influence on weapon system software?	D9	
CIO Exec	42. Of the following potential actions that a software vendor could take, which would be most effective in reassuring you that the vendor's products will operate as intended and will be free from flaws, security vulnerabilities and malicious code? (check up to three) (Delete those that are not your response) o Certify that software meets a designated security target o Provide evidence that software has been scanned for flaws and security vulnerabilities using qualified tools o Provide a list of known flaws and security vulnerabilities o Provide test cases for their software o Provide more transparency or visibility into their development process o Provide information concerning the provenance of the code (e.g. where it was built, who it was acquired from, documentation of individuals who contributed to the code, etc.) o Other (please write-in) o Don't know/unsure o No additional information or actions by software vendors are needed	H	
Thesis Author	43. As a program manager, do you feel you are adequately informed by the Prime contractor of the risks involved with your system's software?	D8	
Thesis Author	44. Do you feel that your program has been provided with adequate resources to address the system's software assurance problem?	E4-5	

Source	Question	Maps to Best Practice #	
		Primary	Secondary
Thesis Author	45. Have you or your staff received a threat briefing regarding the capabilities and intent of adversaries to attack and subvert DoD systems and/or networks through supply chain exploitations or through other sophisticated techniques? If yes, did you find the briefing useful? From who or where did you receive this briefing? How long ago did you receive the briefing? If you have not received this type of briefing, do you think this type of briefing would be useful to you or your staff?	A	
Thesis Author	46. Do you have any recommendations on how the AF might improve the software assurance of its weapon systems?		
Thesis Author	47. Reflecting upon the issues and ideas that these questions may have brought up regarding your system's software assurance, is your answer to question 10(a) still the same? If no, please elaborate on which issue(s) changed your mind.		
Thesis Author	48. If the AF could provide your office with code analysis tools (software and licenses), installation assistance, and the training on how to use them, would your program office want them?		
Thesis Author	49. Are there any additional software assurance related incidents, anecdotes, comments, concerns, etc. not addressed above that you would like to relate, address, or add to?		

Appendix F: Detailed Content Analysis Results

The following Table includes the counts from the content analysis for each type of audit or report.

Table 3. Content Analysis Final Tally of Findings Matched to Best Practices

Best Practices	DSB	GAO	DoD IG	AFAA	Totals
Training & Education					***12***
Improve College-level courses					0
Train Developers					0
Train Managers	1				1
Train software architects/designers					0
Train software testers					0
Train quality assurance personnel					0
Train documentation personnel					0
Other			11		11
Build Security In					***125***
Use structured development model/method	1	4		2	7
Define Security Requirements Early	1	4	4	32	41
Design for security within application architecture		10	35	14	59
Develop Code with security built in; make conscious effort to avoid common errors like buffer overflows					0
Test and review code for defects	2	8	1		11
- Manually inspect or use code analysis tools throughout development			3		3
- Manually inspect or use code analysis tools at specific checkpoints, milestones, or "touchpoints"		1	2	1	3
- Use other robust testing methods (penetration testing, "fuzzing", etc.)			3		0
- Independently inspect/review or test code (Employ third-party)	1	1			1
Other					0
Security Readiness, Documentation, & Response					***89***
Assess/document known security risks and gaps		12	12	38	62
Develop security configuration and policy guide			3	3	6
Document dependencies and module functions		1	2	4	7
Ensure post-deployment incident response measures are in-place		1	6	7	14
Other					0
Code Pedigree/Provenance					***69***
Employ strict source code change mgt techniques		1	8	6	15

Best Practices	DSB	GAO	DoD IG	AFAA	Totals
Secure supply chain… protect against:			3		3
- tampering of physical media or electronic distribution methods			1		1
- threats posed by internally developed code		2			2
- threats posed by closed source code (COTS, Shareware, Freeware)		13		1	14
- threats posed by open source code					0
- threats posed by legacy source code		10			10
- threats posed by outsourced code		10			10
- threats posed by offshore code		12		1	13
- threats posed by mixed code					0
Ensure sample source code is secure					0
Require vendors to disclose usage of third-party software					0
Other	1				1
Research, Evangelism, & Management Involvement					***11***
Conduct secure software research	1				1
Stay current on secure software research					0
Share lessons learned					0
Leaders should make clear what priority security has in the organization			3	2	5
Managers should support security objectives	1		3	1	5
Other					0
Malicious Insiders					***32***
Employ mechanisms to mitigate threat/impact of malicious insiders		3	10	19	32
Other					0
Employee Quality					***1***
Evaluate/Test secure software development skills or knowledge level of current developers					0
Evaluate/Test secure software development skills or knowledge level of prospective new employees					0
Other	1				1
Supplier Security Controls			*1*		***78***
Assess supplier's security controls	1	16		9	26
- Include software assurance requirements in statement of work, contracts		15		36	51
Track supplier performance over time	1				1
Other					0

Bibliography

554 ELSW. (2007). *Application Security Assurance Pilot Final Report (Draft).* Gunter AFB, AL.

ACM Job Migration Task Force. (2006). *Globalization and Offshoring of Software.* (W. Aspray, F. Mayadas, & M. Y. Vardi, Eds.) Association for Computing Machinery (ACM).

Arbaugh, W. A., Fithen, W. L., & McHugh, J. (2000). Windows of Vulnerability: A Case Study Analysis. *Computer , 33* (12), 52-59.

Associated Press. (2007, July 23). *Researchers Seek Cash for Software Flaws.* Retrieved December 8, 2007, from FoxNews: http://www.foxnews.com/story/0,2933,290398,00.html

Balle, J. (2008, January 9). *Interesting Statistics from the Secunia PSI.* Retrieved January 11, 2008, from Secunia.com: http://secunia.com/blog/18/

Beznosov, K., & Kruchten, P. (2004). Towards Agile Security Assurance. *Proceedings of the 2004 Workshop on New Security Paradigms* (pp. 47-54). Nova Scotia, Canada: ACM.

BITS. (2007, October 1). *The Financial Institution Shared Assessments Program: An Overview.* Retrieved January 21, 2008, from BITS Financial Services Roundtable: http://www.bitsinfo.org/FISAP/index.php

Bullard, R., Collins, M. S., Devine, T., Donaldson, M., Edwards, W. H., Gill, J., et al. (2007). *NDIA Systems Assurance Guidebook (v1.2.2).* (C. R. Powell, D. Kleiner, & H. L. Schmidt, Eds.)

Buxbaum, P. (2008, 21 January). *Panel: DoD Software is at Risk.* Retrieved January 21, 2008, from Federal Computer Week (FCW): http://www.fcw.com/print/22_2/policy/151347-1.html

Caudle, S. L. (1994). Using Qualitative Approaches. In J. S. Wholey, H. P. Hatry, & K. E. Newcomer (Eds.), *Handbook of Practical Program Evaluation.* San Francisco, CA: Jossey-Bass Publishers.

Cavusoglu, H., Mishra, B., & Raghunathan, S. (2004). The Effect of Internet Security Breach Announcements on Market Value: Capital Market Reactions for Breached Firms and Internet Security Developers. *International Journal of Electronic Commerce , 9* (1), 69-104.

Chaki, S., & Hissam, S. (2006). *Certifying the Absence of Buffer Overflows.* Software Engineering Institute, Predictable Assembly from Certifiable Components Initiative. Carnegie Mellon University.

Chandra, P., Chess, B., & Steven, J. (2006, May/June). Putting the Tools to Work: How to Succeed with Source Code Analysis. *IEEE Security & Privacy* , 80-83.

Chess, B., & McGraw, G. (2004, November/December). Static Analysis for Security. *IEEE Security & Privacy* , 32-35.

Chess, B., & West, J. (2007). Static Analysis as Part of the Code Review Process. In B. Chess, & J. West, *Secure Programming with Static Analysis.* Addison-Wesley Professional.

Cox, J. (2008, January 15). Wireless LAN scan finds big security holes in NYC retailers' wireless nets. *NetworkWorld* .

Davis, N., Howard, M., Humphrey, W., McGraw, G., Redwine, S., Zibulski, G., et al. (2004). *Processes to Produce Secure Software* (Vol. I). (S. T. Redwine, & N. Davis, Eds.) National Cyber Security Summit - Software Process Subgroup of the Task Force on Security across the Software Development Lifecycle.

Devanbu, P. T., & Stubblebine, S. (2000). Software Engineering for Security: A Roadmap. *Proceedings of the Conference on the Future of Software Engineering* (pp. 227-239). Limerick, Ireland: ACM.

Devanbu, P. T., Fong, P. W.-L., & Stubblebine, S. G. (1998). Techniques for Trusted Software Engineering. *Proceedings of the 20th International Conference on Software Engineering* (pp. 126-135). Kyoto, Japan: IEEE Computer Society.

DHS NCSD. (2006, April 13). *Build Security In Project.* Retrieved February 18, 2008, from Build Security In: https://buildsecurityin.us-cert.gov/daisy/bsi/about_us/bsip.html?branch=1&language=1

DoD. (2004). *Elements of Defense Transformation.* Office of the Secretary of Defense, Office of Force Transformation. Washington, DC: Department of Defense (DoD).

DoD IG. (2002). *Information System Security: User Authentication Protection at Central Design Activities (D-2002-135).* Department of Defense Inspector General (DoD IG).

DoD. (2005). *The Implementation of Network-Centric Warfare.* Office of the Secretary of Defense, Force Transformation. Washington, DC: Department of Defense (DoD).

DSB. (2001). *Defensive Informaiton Operations, Volume II, Part 2, Annexes.* Defense Science Board (DSB), Office of the Undersecretary of Defense For Acquisition, Technology and Logistics. Washington, DC: Department of Defense.

Eglene, O. (2000, January). *Conducting Best and Current Practices Research: A Starter Kit.* Retrieved February 17, 2008, from Center for Technology in Government, University at Albany: www.ctg.albany.edu/publications/guides/conducting_best/conducting_best.pdf

Ellison, R. J. (2005). *Trustworthy Integration: Challenges for the Practitioner.* Software Engineering Institute. Carnegie Mellon University.

Epstein, J., Matsumoto, S., & McGraw, G. (2006, January/February). Software Security and SOA: Danger, Will Robinson! *IEEE Security & Privacy* , 80-83.

Fields, C. (1999, December 17). Report of the Defense Science Board (DSB) Task Force on Globalization and Security. Washington, DC, United States: Defense Science Board, Office of the Secretary of Defense.

florescent_beige. (2008, February 29). *Could a Cyber Attack Trigger a Real War?* Retrieved March 9, 2008, from Slashdot.org: http://interviews.slashdot.org/article.pl?no_d2=1&sid=08/02/29/1733222

Fogerty, K. (2006, October 11). *New CIO Executive Council Poll Reveals CIOs Lack Confidence in Software.* Retrieved September 9, 2007, from CIO Executive Council: https://www.cioexecutivecouncil.com/nb/?nid=24.36d.fd35b9b2

Friedman, T. L. (2000). *The Lexus and the Olive Tree* (First Anchor Books ed.). Anchor Books.

Galin, D., & Avrahami, M. (2007). Benefits of a Higher Quality Level of the Software Process: Two Organizations Compared. *Software Quality Professional (SQP) , 9* (4), 27-35.

GAO. (2007). *High-Risk Series - An Update (GAO-07-310).* United States Government Accountability Office (GAO).

Gengler, E. (2006, December 2006). *Overcoming the "Dangers of Foreign Software".* Retrieved September 4, 2007, from Sourcingmag.com Blogosphere: http://www.sourcingmag.com/blog/archive/overcoming_the_dangers_of_foreign_software.html

Gilbert, A. (2005, October 26). *Zotob damage deep but not widespread.* Retrieved February 17, 2008, from ZDNet News: http://news.zdnet.com/2100-1009_22-5915591.html

Goertzel, K. M. (2008, January 30). *Security in the Software Life Cycle, Version 2.* Retrieved from US-CERT Portal: https://portal.us-cert.gov/ (account required)

Goertzel, K. M., McKinley, H. L., & Winograd, T. K. (2005). *Department of Defense (DoD) Information Assurance (IA) and Computer Network Defense (CND) Strategies - A Comprehensive Review of Common Needs and Capability Gaps State-of-the-Art Report (SOAR).* Information Assurance Technology Analysis Center (IATAC). Fort Belvior, VA: Defense Technical Information Center.

Goertzel, K. M., Winograd, T., McKinley, H. L., Oh, L., Colon, M., McGibbon, T., et al. (2007). *Software Security Assurance: A State-of-the-Art Report (SOAR).* Information Assurance Technology Analysis Center (IATAC). Fort Belvior, VA: Defense Technical Information Center.

Hansen, M., & Nesbit, B. (2000, November). Final Report of the Defense Science Board Task Force on Defense Software. Washington, DC, United States: Defense Science Board, Office of the Secretary of Defense.

Hasan, R., Sion, R., & Winslett, M. (2007). Introducing Secure Provenance: Problems and Challenges. *Proceedings of the 2007 ACM Workshop on Storage Security And Survivability* (pp. 13-18). Alexandria, VA: ACM.

Herbsleb, J., Carleton, A., Rozum, J., Siegel, J., & Zubrow, D. (1994). *Benefits of CMM-Based Software Process Improvement: Initial Results.* Software Engineering Institute. Pittsburgh, PA: Carnegie Mellon University.

Hope, P., McGraw, G., & Antón, A. (2004, May/June). Misuse and Abuse Cases: Getting Past the Positive. *IEEE Privacy & Security* , 32-34.

Howard, M., & LeBlanc, D. (2003). *Writing Secure Code* (2nd ed.). Microsoft Press.

Howard, M., & Lipner, S. (2003, January/February). Inside the Windows Security Push. *IEEE Security & Privacy* , 57-61.

IC3. (2007). *Internet Crime Report: January 1, 2006 - December 31, 2006.* National White Collar Crime Center and the Federal Bureau of Investigation. Internet Crime Complaint Center (IC3).

INPUT. (2005). The Impact of Software Assurance on the Procurement Process. *TargetVIEW , 1* (10).

Irvine, C. E., & Levitt, K. (2007). Trusted Hardware: Can It Be Trustworthy? *Proceedings of the 44th Annual Conference on Design Automation.* San Diego, CA: ACM.

Jackson, D., Thomas, M., & Millett, L. I. (Eds.). (2007). *Software for Dependable Systems: Sufficient Evidence?* National Academies Press.

Jackson, W. (2008, February 22). *The common cold of IT security.* Retrieved February 25, 2008, from Government Computer News (GCN): http://www.gcn.com/online/vol1_no1/45864-1.html?topic=security

Jarzombek, J. (2007, January 17). Considerations for Quality & Assurance with Scale and Uncertainty. U.S. Department of Homeland Security.

Jarzombek, J. (2007, May 9). Security in the Software Lifecycle. U.S. Department of Homeland Security.

JTFGNO. (2008). *Information Assurance Vulnerability Management (IAVM) Program*. Retrieved February 18, 2008, from Joint Task Force Global Network Operations (JTFGNO): https://www.jtfgno.mil/bulletins/iava/iava_index.htm#2007

Keizer, G. (2004, November 30). Unprotected PCs Fall To Hacker Bots In Just Four Minutes. *Tech Web* .

Kelley, D. (2003, November 18). *Applying patches? Call a doctor.* Retrieved February 20, 2008, from ZDNet News: http://news.zdnet.com/2100-9595_22-5108482.html?tag=btxcsim

Kreitner, C. (2008, January 25). DoE IG Looks to Unify Cyber Security Practices (Editor's Note). *SANS NewsBites , 10*(7). SysAdmin, Audit, Network, Security (SANS) Institute.

Kresge, S. T. (2005). *Aircraft Accident Investigation, F/A-22 S/N 00-4014.* United States Air Force.

Landwehr, C. E., Bull, A. R., McDermott, J. P., & Choi, W. S. (1994). A Taxonomy of Computer Program Security Flaws. *ACM Computing Surveys , 26* (3), 211-254.

Leedy, P. D., & Ormrod, J. E. (2005). *Practical Research: Planning and Design* (8th ed.). Upper Saddle River, NJ: Pearson Education, Inc.

Lemon, S. (2007, September 03). *Expert finds 'stupid' vulnerabilities in Oracle 11g.* Retrieved September 11, 2007, from Computerworld: http://www.computerworld.com/action/article.do?command=printArticleBasic&articleId=9034078

Lemos, R. (2004, April 15). *Stampede for patches disrupts Microsoft update site.* Retrieved February 20, 2008, from ZDNet News: http://news.zdnet.com/2100-1009_22-5191796.html?tag=btxcsim

Levy, E. (2003, May/June). Poisoning the Software Supply Chain. *IEEE Security & Privacy* , 70-73.

Lewis, J. A. (2007). *Foreign Influence on Software: Risks and Recourse.* Washington, DC: Center for Strategic and International Studies.

Lewis, J. J. (2004). *Computers Quotes.* Retrieved March 12, 2008, from Wisdom Quotes: Quotations to inspire and challenge: http://www.wisdomquotes.com/cat_computers.html

Lipner, S., & Howard, M. (2004). The Trustworthy Computing Security Development Lifecycle. *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC'04)* (pp. 2-13). IEEE Computer Society.

Lipner, S., & Howard, M. (2005, March). *The Trustworthy Computing Security Development Lifecycle.* Retrieved February 11, 2008, from Microsoft Developer Network (MSDN) Library: http://msdn2.microsoft.com/en-us/library/ms995349.aspx

Long, L. N. (2008). The Critical Need for Software Engineering Education. (K. Johnstun, & C. Fortier-Lozancich, Eds.) *CrossTalk , 21* (1), 6-10.

McAfee Avert Labs. (2007). *Top 10 Threat Predictions for 2008.* Santa Clara, CA: McAfee, Inc.

McDonald, T. (2000, July 11). Report: Year's Hack Attacks to Cost $1.6 Trillion. *E-Commerce Times* .

McGraw, G. (2002). Building Secure Software: Better than Protecting Bad Software. *IEEE Software , 19* (6), 29-31.

McGraw, G. (2003, March/April). From the Ground Up: The DIMACS Software Security Workshop. *IEEE Security & Privacy* , 2-9.

McGraw, G. (1999). Software Assurance for Security. *Computer , 32* (4), 103-105.

McGraw, G. (2004). Software Security. *IEEE Security & Privacy , 2* (2), 80-83.

McGraw, G. (1998). Testing for Security During Development: Why we should scrap penetrate-and-patch. *IEEE Aerospace and Electronic Systems , 13* (4), 13-15.

McGraw, G., & Viega, J. (2000, February 01). *Make your software behave: Assuring your software is secure.* Retrieved from IBM developerWorks: http://www.ibm.com/developerworks/library/s-assurance.html

McNamara, P. (2007, November 16). *Half of stores invite wireless ID theft: survey.* Retrieved January 17, 2008, from NetworkWorld: http://www.networkworld.com/community/node/21995

Mead, N. R., & McGraw, G. (2005, July/August). A Portal for Software Security. *IEEE Security & Privacy* , 75-79.

Mead, N. R., Hough, E. D., & Stehney, T. R. (2005). *Security Quality Requirements Engineering (SQUARE) Methodology.* Software Engineering Institute (SEI), Networked Systems Survivability Program. Pittsburgh, PA: Carnegie Mellon University (CMU).

Merle, R. (2007, February 7). *Problems Stall Pentagon's New Fighting Vehicle.* Retrieved December 12, 2007, from Washington Post:

http://www.washingtonpost.com/wp-dyn/content/article/2007/02/06/AR2007020601997_pf.html

Mold, J. W., & Gregory, M. E. (2003). Best Practices Research. *Family Medicine , 35* (2), 131-134.

Nilsen, S. R. (2005). *Offshoring of Services - An Overview of the Issues.* Education, Workforce, and Income Security Issues. United States Government Accountability Office.

NIST. (n.d.). *CVE and CCE Statistics Query Page*. (National Institute of Standards and Technology (NIST)) Retrieved January 2008, from National Institute of Standards and Technology (NIST) National Vulnerability Database (NVD): http://nvd.nist.gov/statistics.cfm

Olsen, F. (2004, March 25). *Security needs better education for programmers.* Retrieved December 10, 2007, from Federal Computer Week (FCW): http://www.fcw.com/fcw/articles/2004/0322/web-secure-03-25-04.asp

Osinga, F. (2007). On Boyd, Bin Laden, and Fourth Generation Warfare as String Theory. In J. Olson, *On New Wars.* Institutt for forsvarsstudier.

Ozment, A. (2007, August 2). *Research.* Retrieved March 8, 2008, from Andy Ozment: http://www.andyozment.com/research/

PandaLabs. (2008). *Annual Report PandaLabs 2007.* Panda Security.

Polydys, M. L., & Wisseman, S. (2007, September 10). Software Assurance (SwA) in Acquisition: Mitigating Risks to the Enterprise. *Draft Version 1.0*. Department of Defense (DoD) and Department of Homeland Security (DHS) Software Assurance (SwA) Acquisition Working Group.

Polydys, M. L., Ryan, D. J., & Ryan, J. J. (2006). Best Software Assurance Practices in Acquisition of Trusted Systems. *Proceedings of the 10th Colloquium for Information Systems Security Education* (pp. 54-59). Adelphi, MD: University of Maryland, University College.

Potter, B., & McGraw, G. (2004, September/October). Software Security Testing. *IEEE Security & Privacy* , 32-36.

Powner, D. A., & Rhodes, K. A. (2007). *Cybercrime - Public and Private Entities Face Challenges in Addressing Cyber Threats.* United States Government Accountability Office.

Prentkiewicz, R. J., & Massey, A. J. (2006). *Air Force Equipment Management System Controls (F2006-0011-FB2000).* Air Force Audit Agency, Operations Directorate. Washington, DC: United States Air Force.

Pruessner, A., & Paternostro, C. (2006). Software Quality: From Legacy Codes to Web-based Enterprise Applications. *Software Quality Professional (SQP) , 8* (3), 4-11.

Rowlands, G. (2003). Software, Safety and SOUP. *Standards in Defence News* (187), 14.

RTI. (2002). *The Economic Impacts of Inadequate Infrastructure for Software Testing.* National Institute of Standards & Technology (NIST). U.S. Department of Commerce.

SAFECode. (2008, February). *Software Assurance: An Overview of Current Industry Best Practices.* Retrieved February 2008, from Software Assurance Forum for Excellence in Code (SAFECode): http://www.safecode.org/publications/SAFECode_BestPractices0208.pdf

SANS. (2007, March 24). *New Report Identifies the Three Programming Errors Most Frequently Responsible for Critical Security Vulnerabilities and Security Incidents in 2006.* Retrieved December 9, 2007, from SANS Institute: http://www.sans-ssi.org/top_three.pdf

SANS. (2008, January 18). *SANS NewsBites - Volume: X, Issue: 5.* Retrieved January 18, 2008, from SANS Institute: http://www.sans.org/newsletters/newsbites/newsbites.php?vol=10&issue=5

SANS SSI. (2007). *About the Secure Programming Skills Assessment (v0.91).* Retrieved December 9, 2007, from SANS Software Security Institute (SSI): http://www.sans.org/ssi

Schinasi, K. V. (2004). *Defense Acquisitions: Knowledge of Software Suppliers Needed to Manage Risks (GAO-04-678).* Acquisition and Sourcing Management. Washington, DC: United States General Accounting Office.

Schneider, F. B. (Ed.). (1999). *Trust in Cyberspace.* Washington, DC, United States: National Academy Press.

Schneider, T. (2006). Secure Software Engineering Processes: Improving the Software Development Life Cycle to Combat Vulnerability. *Software Quality Professional (SQP) , 9* (1), 4-13.

Schneider, W. J. (2007, September 21). Final Report of the Defense Scienc Board Task Force on Mission Impact of Foreign Influence on DoD Software. Washington, DC, United States: Defense Science Board, Office of the Secretary of Defense.

Sentrigo. (2008, January 14). *Survey of Oracle Database Professionals Reveals Most Do Not Apply Security Patches.* Retrieved February 18, 2008, from Sentrigo.com: http://www.sentrigo.com/press_releases-newsid-39.htm

Shah, J. (2007). Online Crime Migrates to Mobile Phones - As cell usage climbs, unfamiliar threats are heading our way. (D. Sommer, Ed.) *Sage , 1* (2).

Skibell, R. (2003). The Phenomenon of Insecure Software in a Security-Focused World. *Journal of Technology Law & Policy , 8* (2).

Songini, M. L. (2007, February 23). *Air Force Raptors cancel mission after software glitch.* Retrieved January 30, 2008, from Computerworld: http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=9011691

Sophos. (2008). *Security Threat Report: 2008.* Sophos.

Spafford, E. H. (2005, October 27). Testimony before the House Armed Services Committee Hearing on "Cyber Security, Information Assurance and Information Superiority".

Steven, J. (2006, March/April). Adopting an Enterprise Software Security Framework. *IEEE Security & Privacy* , 84-87.

Sullivan, M. (2008). *Best Practices - Increased Focus on Requirements and Oversight Needed to Improve DoD's Acquisition Environment and Weapon System Quality (GAO-08-294).* Acquisition and Sourcing Management. United States Government Accountability Office.

Taylor, D., & McGraw, G. (2005, May/June). Adopting a Software Security Improvement Program. *IEEE Security & Privacy , 3* (3), pp. 88-91.

Telos. (2007, October 9). *Air Force Signs $10.2M Contract with Telos to Provide Applicaiton-level Security.* Retrieved February 18, 2008, from Telos.com: http://www.telos.com/company/news/default.cfm?contentID=185

USAF. (2008). *Video: Cyber Dominance: It takes cyber dominance to defend America in a changing world.* Retrieved March 8, 2008, from U.S. Air Force: http://www.airforce.com/achangingworld/

van Wyk, K. R., & Steven, J. (2006, September/October). Essential Factors for Successful Software Security Awareness Training. *IEEE Security & Privacy* , 64-67.

van Wyk, K. (2007). Software Security - Setting the Stage. *Software Engineering Process Group Conference (SEPG 2007).* Austin, TX: Software Engineering Institute, Carnegie Mellon University.

Verduyn, B. (2006). *2005 FBI Computer Crime Survey.* Federal Bureau of Investigation. Houston, TX: U.S. Department of Justice.

Viega, J., & McGraw, G. (2001). *Building Secure Software: How to Avoid Security Problems the Right Way.* Addison-Wesley Professional.

Vijayan, J. (2008, January 16). *DBA's not to blame for Oracle patch application failures?* Retrieved January 19, 2008, from Computerworld: http://blogs.computerworld.com

Walker, E. (2005). Software Development Security: A Risk Management Perspective. *Software Tech News , 8* (2).

Woody, C. (2005). *Eliciting and Analyzing Quality Requirements: Management Influences on Software Quality Requirements.* Software Engineering Institute. Carnegie Mellon University.

Woody, C. (2007). *Process Improvement Should Link to Security: SEPG 2007 Security Track Recap.* Software Engineering Institute. Carnegie Mellon University.

Yeakley, C. L., & Fiebrich, J. D. (2006). Embedding Software Quality: Capitalizing on Available Resources. *Software Quality Professional (SQP) , 8* (4), 28-41.

Yin, R. K. (2003). *Case Study Research: Design and Methods* (3rd ed.). Thousand Oaks, CA: SAGE Publications.

Zayaraz, G., & Thambidurai, P. (2007). Quantitative Model for the Evaluation of Software Architectures. *Software Quality Professional (SQP) , 9* (3), 28-40.

Vita

Major Ryan Maxon was born in Kearney, NE, and graduated from high school there in 1993. He then attended the University of Nebraska at Lincoln (UNL), where he earned a Bachelor of Science degree in Computer Science, with a minor in History. While at UNL, he was in the Air Force Reserve Officer Training Corps and received his commissioned into the United States Air Force in January 1998.

After completing Basic Communications Officer Training (BCOT) in May 1998, he reported to his first assignment as the Chief, Air Traffic Control and Landing Systems Maintenance, 49th Communications Squadron (CS), Holloman AFB, NM. He also filled in as the Acting Flight Commander, Planning and Information Flight, and Special Projects Officer. In February 2000, he was assigned to the National Air Intelligence Center, Wright-Patterson AFB, OH, as an Information System Software Engineer. In January 2003, he was assigned to the HQ Air Force Weather Agency (AFWA), Offutt AFB, NE, where he held positions as the Chief, AFWA Web Services, Information Assurance Project Officer, Deputy Chief, Systems Branch, and Acting Chief, Communications Technology Branch. During his assignment to AFWA, he deployed to Al Udeid, Qatar, where he served as the Deputy Flight Commander, Planning and Implementation Flight, 379th Expeditionary Communications Squadron, in support of OPERATION IRAQI FREEDOM, OPERATION ENDURING FREEDOM, and CJTF HORN OF AFRICA.

In August 2006, he entered the Graduate School of Engineering and Management at the Air Force Institute of Technology, where he is completing his degree in Information Resource Management, with sequences in Information Assurance (NSA 4012 Certification) and Strategic Information Management. Upon graduation, he will be assigned to US Strategic Command, Peterson AFB, CO.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From – To)*
27-03-2008	Master's Thesis	August 2006 – March 2008

4. TITLE AND SUBTITLE

Software Assurance Best Practices for Air Force Weapon and Information Technology Systems – Are We Bleeding?

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Maxon, Ryan A., Major, USAF

5d. PROJECT NUMBER
If funded, enter ENR #

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way, Building 640
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GIR/ENV/08-M13

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Intentionally left blank

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

In the corporate world, "bits mean money," and as the Department of Defense (DoD) becomes more and more reliant on net-centric warfare, *bits mean national security*. Software security threats are very real, as demonstrated by the constant barrage of Internet viruses, worms, Trojans, and hackers seeking to exploit the latest vulnerability. Most organizations focus their resources on reactive defenses such as firewalls, antivirus software, and encryption, however as demonstrated by the numerous attacks that are successful, those post facto measures are not enough to *stop the bleeding*.

The DoD defines software assurance (SwA) as the "level of confidence that software functions as intended and is free of vulnerabilities, either intentionally or unintentionally designed or inserted as part of the software." SwA focuses on *baking in* security versus *bolting it on* afterwards. The Department of Homeland Security and DoD each have had SwA programs for a few years; however the Air Force (AF) just recently formed the Application Software Assurance Center of Excellence at Maxwell AFB-Gunter Annex, AL.

This research seeks to identify common issues that present challenges to the development of secure software and best practices that the AF could adopt as it proactively begins to heal the SwA problem.

15. SUBJECT TERMS

Software, Security, Information, Assurance, Best Practice, Provenance, Pedigree, Weapon Systems, COTS, Open Source, Third-party, Supply Chain, Closed Source, Air Force, Department of Defense, AF, DoD, OSS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dennis Strouble, PhD
U	U	U	UU	147	19b. TELEPHONE NUMBER *(Include area code)* (937) 255-3355, ext 3323 (dennis.strouble@afit.edu)

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18