Software Assurance Practice at Ford: A Case Study

Dr. Nancy R. Mead
Software Engineering Institute

Dr. Dan Shoemaker
University of Detroit Mercy

Jeffrey A. Ingalsbe
Ford Motor Company

Software pervades our technological society, handling our financial transactions, managing power transmission, facilitating most forms of communication, and keeping us safe. This makes defects in software one of the most potent threats to our national security, and turns identification of best practices in software development, acquisition, and long-term use the highest national priority. This article presents the best practices employed by the Ford Motor Company to develop and maintain their software assets.

Defects in software are among the most potent threats to our national security. That is because these defects—whether by malicious agent placement or faulty manufacturing—represent avenues of attack for any potential criminal, terrorist, or enemy adversary.

Therefore, it is the highest national priority to find and employ the right set of practices in software development, acquisition, and long-term use that will prevent those defects. This is a noble goal, but the fact is that there is very little agreement on what exactly is the optimum set of best practices. And worse, we have almost no idea what the current state-of-the-practice is in business and industry.

Essentially, there are no concrete points of reference to guide us in affecting change to the noticeably unsuccessful way that industry has currently approached the problem. It's like "Alice in Wonderland" when Alice doesn't care where she goes. The Cheshire Cat responds that in getting "somewhere" that "it doesn't matter which way you go ... you're sure to do that ... if you only walk long enough." So it seems like the alternatives are to develop a clear understanding of the current state-of-the-industry best practice, or to continue to do a lot of pointless walking.

One Company's Best Practice Example

With no clear practices currently in place, a case study of current industry practice, especially from a *Fortune* 10 company, is an extremely valuable and useful tool for people who are interested in changing the state of practices in the software community. This article presents an overview of the control processes employed by the Ford Motor Company to develop and maintain their software assets, a brief history of those processes to provide context, and a discussion of their future.

What Has Gone Before

The IT Security and Controls organization at Ford dates back to 1998, years before an agreed-upon international standard for information security management systems like ISO/IEC 27001:2005 [1] existed. Since that time, the organization has developed control processes for applications (first) and infrastructure (second). These are the application control review (ACR) process and the infrastructure control review (ICR) process, respectively. Later, they added a control process called systems control review (SCR), to be conducted yearly to assess the effectiveness of controls that had been specified by the ACR or ICR process and to address the results of internal audits. Ethical hacking and static code analysis (on select applications or infrastructure) were instituted to identify vulnerabilities (read defects) in code that was already written. Threat modeling (on select applications and infrastructure) was instituted to prevent vulnerabilities before code is written. Finally, the data from ethical hacking and static code analysis was fed back into a training and awareness program to help developers understand common errors being inserted into code.

Current State Control Processes

Ford's current control processes span the entire asset life cycle (software or hardware) from conception to retirement. They are the ACR process, the ICR process, threat modeling, ethical hacking, static code analysis, risk assessment, and the SCR process. As a set, these processes work together to ensure the overall security and integrity of Ford's software. Figure 1 shows when these processes are typically executed in the life cycle but does not show how often they are executed. For example, all applications must compose application control documentation using the ACR process before launch, but not all applications are threat modeled. Threat modeling is performed only on those applications that score highest on a risk assessment or are deemed strategically important (a small subset of the entire portfolio). Ethical hacking is performed based on an independent assessment of the criticality of the system or infrastructure. After launch, a yearly risk assessment is performed that determines whether a more detailed SCR is required. All applications and infrastructure scoring high and a semi-random sample of those scoring medium or low are required to perform the SCR. The following sections give an overview of the processes used.

ICR Process

The ICR process is responsible for ensuring that the overall IT infrastructure is correct and that adequate controls exist.

Any system activity planning to use a piece of infrastructure must ensure that an ICR has been done prior to its inclusion in the application design. More importantly, a reference to the infrastructure's ICR must be included in the ACR of every application using the infrastructure.

The ICR is initiated by an asset owner and performed by a designated internal control coordinator (ICC), with assistance from a security control champion (SCC). Next, a meeting is conducted with the person who is accountable for managing any identified risk (the infrastructure owner). The meeting might also include the technical support staff or subject matter experts of the infrastructure owner. For purchased or commercial off-the-shelf solutions, the vendor's technical support staff may be included as part of the infrastructure team. In this meeting, infrastructure components and the risks applicable to them are identified and a *risk matrix* is developed.

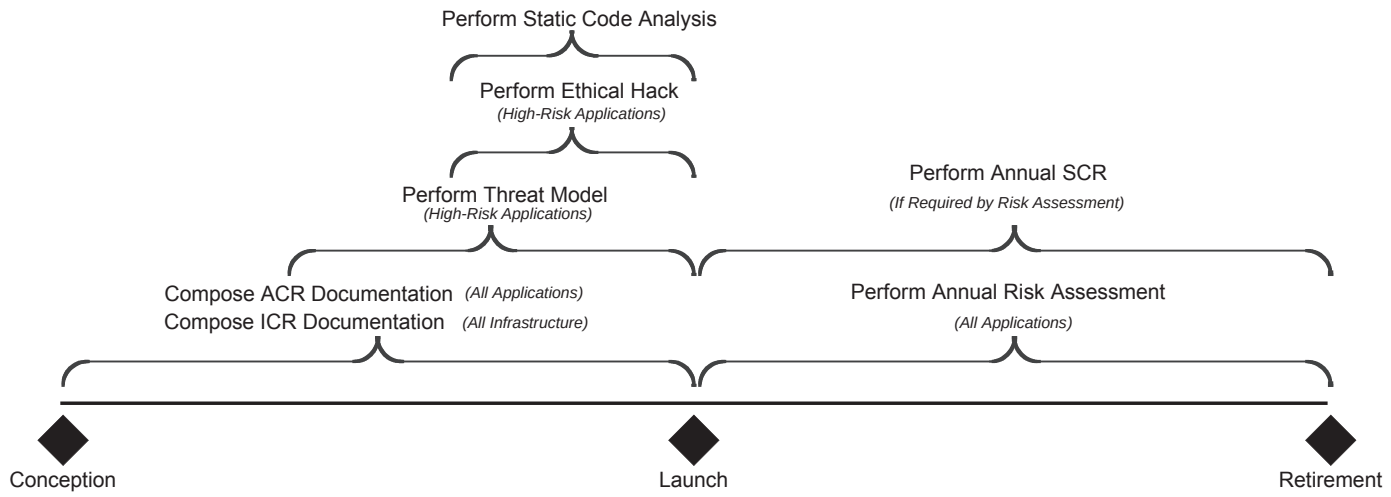

Figure 1: *Ford's Control Processes*

The risk matrix is then used to develop the ICR package. The infrastructure owner, ICC, and SCC prepare this package. It includes an overview of the technology being employed, a network diagram, and a data/process flow diagram. The two diagrams document where the infrastructure is being deployed, identify the hardware used to deploy it, and show how data flows through the infrastructure. The infrastructure risk matrix is also cross-referenced to the tangible controls that have been put in place to address each threat.

Finally, all infrastructure components are explicitly itemized along with all threats included in the infrastructure risk matrix. At this point, the controls used to reduce the risk of a specific threat to that component are itemized in terms of who is performing the action, what they are doing, and what threat is being reduced by performing that action. A specific disaster recovery plan is also specified along with the roles and responsibilities for the critical job functions needed to implement and support the infrastructure once a change has been made.

ACR Process

The ACR process operates at the level of individual applications, which is the level that would be of the most interest to people concerned with secure software assurance best practices.

The basic goal of the ACR process is to reduce the risk associated with information technology applications. It does that by ensuring that appropriate controls are implemented for each application and that those controls are functioning properly.

The ACR process assesses all significant new and changing services, processes, operations, and control processes. The ACR process applies to all IT applications, including commercial off-the-shelf, independent of whether the application resides internally (on the internal Ford network) or externally (hosted by an external provider).

ACR/ICR Roles and Responsibilities

The ACR and ICR processes involve the application/infrastructure owner, the IT organization, a designated ICC, an SCC, and (sometimes) an auditor. Telecommunication services is involved for external facing assets and externally hosted assets.

The application or infrastructure owner is responsible for ensuring that adequate controls exist and that the controls mitigate risk at a reasonable cost. The IT organization is responsible for assisting the application or infrastructure owner in defining adequate controls and incorporating them into the application or infrastructure. The ICC is responsible for overseeing the ACR and ICR for the relevant application data, programs, and infrastructure. The ICC is responsible for reviewing and approving the application classification and also provides advice on business controls and coordinates the control review. The SCC is responsible for guiding application owners in performing the specified ACR and ICR processes and also assists with the completion of the ACR/ICR documents (as required). The SCC conducts risk assessments and pre-implementation reviews for compliance with corporate policies and verifies that the documentation is valid based on current technical and policy information.

ACR Process Steps

The first step in the ACR process is to classify the application using the three basic security objectives: confidentiality, integrity, and availability (CIA). The application is classified along a sliding scale from low to high impact (numerically from 1 to 3) on each of these objectives[1]. From the CIA rating, a list of questions is automatically generated. Each question addresses a particular control.

The application owner, IT organization, SCC, and ICC work together in a series of meetings to prepare the materials required for completion of the ACR. The required materials include a non-technical description of the application that describes the purpose of the application, how it will function when installed, products, hardware, and software needed, and what activities use the application. The review materials also include a process flowchart of the application that identifies and describes the sources of input, master files, outputs, and major processing programs. It must describe the application to someone not familiar with it and graphically highlight where controls are required. Also required is a network flowchart, which is a detailed depiction of the network and the various connected host computers. The aim of this flowchart is to facilitate an understanding of the controls architecture. It depicts the flow of application data and the direction of the data flow. If connectivity is required from an outside vendor, the diagram should also depict the access-control points (firewalls, routers, or virtual private network devices) and the network flow though these devices.

Once the materials are complete, the ICC will determine that the ACR is ready for a formal review. The product of the formal review is a statement concerning the adequacy of controls. The application owner is then responsible for revising the controls and corresponding ACR documentation based on the outcome of this review.

SCR Process

The SCR process is performed annually on those applications or infrastructure that score *High* on the annual risk assessment as well as a semi-random sample of medium- and low-risk applications or infrastructure. It comprises a set of checklists, testing, and evidence-gathering techniques that (together) are used to assess whether all of the controls that were documented by the ACR and ICR processes are being performed. It essentially validates that the planned controls are working and that additional controls are put in place if anything has changed since the last review.

The IT Policy Manual, which guides all Ford IT work, requires an annual risk assessment for each application and infrastructure component. Ford requires this in order to prioritize SCR process work each year. The SCR process ensures that all of the necessary controls are adequate and are functioning effectively. This helps managers to identify control weaknesses before they can be exploited. It also provides a mechanism and process for developing corrective actions and tracking closure of the weaknesses that may be found.

Threat Modeling

Threat modeling is performed on strategically important projects and those that score high on a risk assessment. It is typically performed in the design phase and is focused on prevention of vulnerabilities that may have a high business impact if exploited. In its simplest form, threat modeling involves six steps:

1. Identifying assets within the application or infrastructure (e.g., data or processes).
2. Identifying people involved (e.g., administrators or users).
3. Identifying high-level or meta-use cases (these are not the same as Unified Modeling Language use cases).
4. Identifying threats to assets.
5. Classifying threats using what is called DREAD—**D**amage potential, **R**eproducibility, **E**xploitability, **A**ffected users, and **D**iscoverability [2]—and then assigning a 0 to 5 value to each of the five types[2].
6. Choosing whether to accept, transfer, avoid, or mitigate the risk posed by those threats.

Threat modeling typically takes four to five working sessions, and should involve both IT (because they understand threats) and the (internal) business customer (because they understand value).

Ethical Hacking

Ethical hacking is performed on strategically important projects and by request. It is typically performed after implementation and is focused on the detection of vulnerabilities. The ethical hack team uses a combination of tools, processes, and acquired skills to perform the hacks. Detailed metrics are kept on the results of the hack and given to the requestor. Metrics on classes of vulnerabilities (e.g., buffer overflow, cross-site scripting, Structured Query Language injection) have been used to develop a training course for developers.

> *"With no clear practices currently in place, a case study of current industry practice ... is an extremely valuable and useful tool for people who are interested in changing the state of practices in the software community."*

Static Code Analysis

Static code analysis is performed by request when an application owner feels that there is a higher risk associated with the project. It is typically performed after implementation and is focused on the detection of vulnerabilities. A commercial off-the-shelf tool is used to analyze the code. False positives are eliminated and the report is given back to the requestor.

Future State

The control processes outlined in this article have been instituted over a period of 10 years. The older processes have an information assurance (IA) *feel* to them (i.e., information protection) while the newer processes have a software assurance feel (i.e., prevention of coding vulnerabilities). The acceptance and popularity of ISO/IEC 27001 and the desire to make decisions based on business value and risk has prompted Ford's IT Security and Controls organization to begin aligning their processes with the international standard. The work is in its infancy, but it is already clear that quantifying asset value and risk—and using them to make decisions—is the right course.

Observations

This is a case study, so conclusions should not be drawn. However, some important observations can be made based on what has been reported here.

First, Ford clearly pays considerable attention to the security of its applications and its associated infrastructure. The degree of detailed rigor of their processes, the time spent, the documentation produced, and the obvious bureaucratic coordination and control requirements support that observation. Additionally, they pay particular attention to anything that might fall under Sarbanes-Oxley Act[3] purview by requiring that those applications are always addressed with the highest degree of rigor. Their control perspective, until recently, was oriented toward classic IA principles rather than those of software assurance. Specifically, the risks are rated on the CIA scale, which is typical of IA risk assessments. While IA concerns are to some extent founded on secure software assurance, the things that threaten information are not the same as those things that might threaten software. In fact, it is perfectly possible for a piece of software, both under development and in actual use, to have all of the necessary controls to assure any set of regulatory requirements while still being full of exploitable holes.

Second, anecdotal evidence suggests that there may be an evolutionary progression an organization goes through as they move from *no security focus* to *best in class*. First, they get things like antivirus, incident response, and forensics in place. These things are served up centrally and (of course) have an IA focus. Next, they move on to securing applications and infrastructure. However, since they have probably not plugged into those organizations yet, it is natural for them to show up just before launch or at gate review to render their opinion on the security of the application or infrastructure. This would tend to be adversarial. Next, they would get involved early in the software development process and help developers classify the importance of the information being created/deleted/modi-

fied/transmitted by the application. They would then prescribe controls based on that score and do things like ethical hacking to ensure that those with the highest scores don't have big vulnerabilities. Some companies stop at this point. More introspective companies may begin to engage their development community in order to build better software and ask questions like, "Is there a need to protect things at different levels?" Such companies would then do things like look at their penetration test results and take the top 10 vulnerabilities back to the development community, then working with them to change coding practices in order to eliminate those vulnerabilities and establish better coding practices. This is where Ford is right now. They have rolled out threat modeling and are engaging their development community in order to develop more secure coding practices.

The fact that a company well-known for its competence and effectiveness in IT security has recognized (and is acting upon) the need to move beyond IA to software assurance bodes well for others who are not nearly as far along the evolutionary scale. However, it may indicate that we have a ways to go in popularizing the importance of software assurance best practice in conventional IT security operation.

It is possible and (perhaps) probable that many of the practices that the software assurance community would recognize as secure software assurance are taking place on an ad-hoc basis within the actual application development and maintenance function within IT itself, which suggests where the next study should be focused. However, it is also clear that the only way to ensure that those practices are institutionalized is to provide both the incentive and the guidance to get large companies to implement secure software assurance practice as part of their conventional application security operation.◆

References

1. ISO/IEC 27001:2005. "Information Technology – Security Techniques – Information Security Management Systems – Requirements." Distributed through the American National Standards Institute. 23 Aug. 2007.
2. Howard, Michael, and David LeBlanc. Writing Secure Code (With CD-ROM). Microsoft Press, 2001.

Note

1. For example, an application that handles the most confidential data with the highest integrity requirement and the highest availability requirement would be rated 3,3,3. An application that handles the least confidential data with the lowest integrity requirement and the lowest availability requirement would be rated 1,1,1.
2. By assigning a 0 to 5 value for each part of DREAD, a final risk value is obtained that allows threats to be compared. Despite the fact that a *formula* is used to calculate a risk value, it is important to understand that it is still subjective. A significant effort by participants needs to be made to be consistent.
3. The Sarbanes-Oxley Act of 2002 (Pub.L. 107-204, 116 Stat. 745, enacted 30 July, 2002) is a federal law that describes specific mandates and requirements for financial reporting.

About the Authors

Nancy R. Mead, Ph.D., is a senior member of the technical staff in the Networked Systems Survivability Program at the SEI. She is also a faculty member at Carnegie Mellon University. Mead's research interests are in the areas of information security, software requirements engineering, and software architectures. She is a Fellow of the IEEE and the IEEE Computer Society and is also a member of the Association for Computing Machinery. Mead received her doctorate in mathematics from the Polytechnic Institute of New York, and received bachelor's and master's degrees in mathematics from New York University.

SEI
4500 5th AVE
Pittsburgh, PA 15213
E-mail: nrm@sei.cmu.edu

Dan Shoemaker, Ph.D., is the director of the Centre for Assurance Studies. He has been professor and chair of computer and information systems at the University of Detroit Mercy for 24 years, and co-authored the textbook, "Information Assurance for the Enterprise." His research interests are in the areas of secure software assurance, information assurance and enterprise security architectures, and information technology governance and control. Shoemaker has both a bachelor's and a doctorate degree from the University of Michigan, and master's degrees from Eastern Michigan University.

Computer and Information Systems – College of Business Administration
University of Detroit Mercy
Detroit, MI 48221
Phone: (313) 993-1202
E-mail: shoemadp@udmercy.edu

Jeffrey A. Ingalsbe is a senior security and controls engineer with the Ford Motor Company. He is involved in information security solutions for the enterprise, threat modeling efforts, and strategic security research. Ingalsbe also serves as an expert industry panelist on two national working groups within the DHS's Cybersecurity Division. He has a bachelor's degree in electrical engineering and a master's degree in computer information systems from Michigan Technological University and the University of Detroit Mercy, respectively. Ingalsbe is currently working on his doctorate degree in software engineering at Oakland University.

Ford Motor Company
17475 Federal DR
STE 800-D04
Allen Park, MI 48101
Phone: (313) 390-9278
E-mail: jingalsb@ford.com