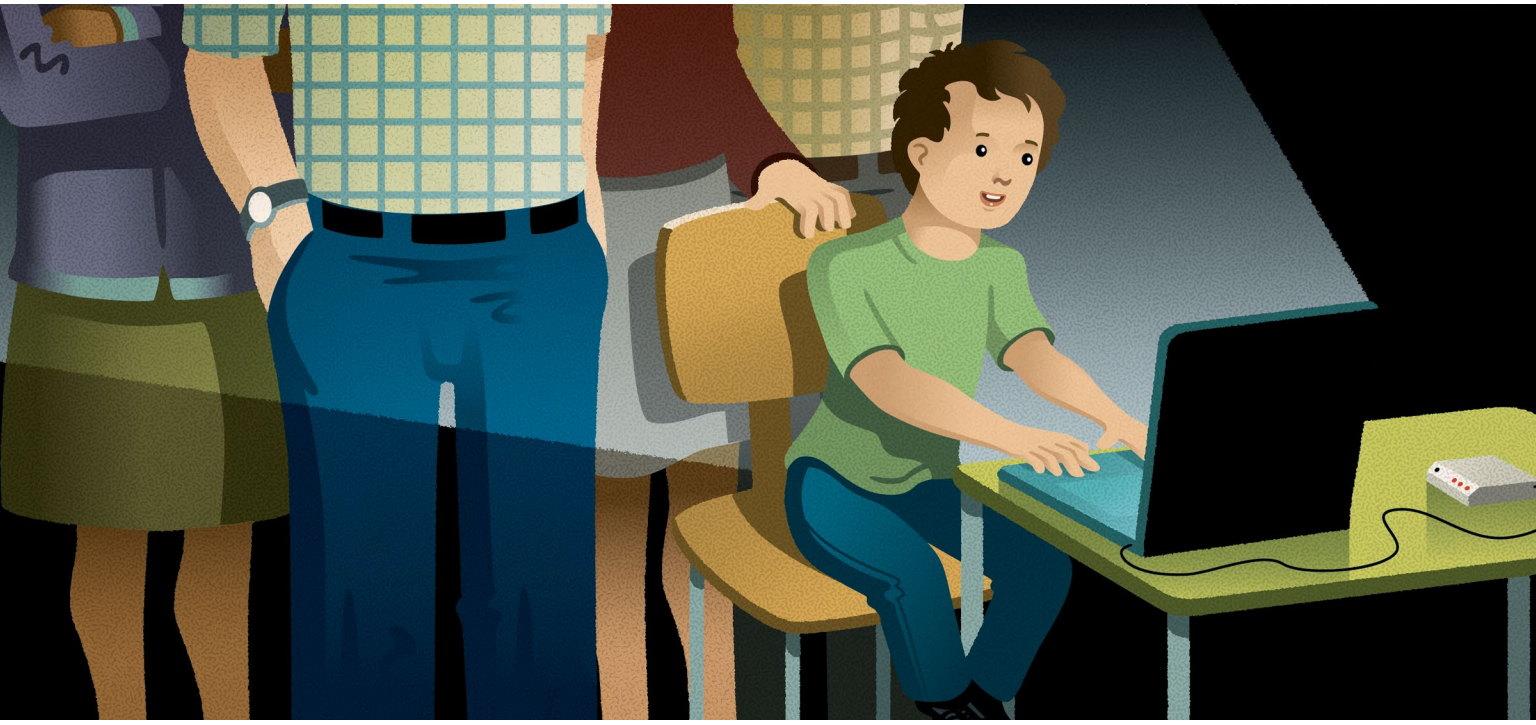
‘WHY JOHNNY CAN’T WRITE SECURE CODE’

BY **KEITH McMILLAN**



UNDERSTANDING TODAY’S AGILE SOFTWARE DEVELOPMENT CAN HELP MAKE SECURITY A BIGGER PART OF THE APPLICATION DEVELOPMENT PROCESS.

ILLUSTRATION BY PETER AND MARIA HOEY

ON THE AGENDA at a recent security conference was a session titled “Why Johnny Can’t Write Secure Code.” The presenter designed the title to be tongue-in-cheek, but it underscores a constant, growing issue: the lack of security embedded into the software development process. This is particularly problematic for those who design web applications that are ripe for exploitation.

Last year’s Imperva Web Application Attack Report showed a threefold increase in SQL injection attacks and a 2.5 times increase in cross-site scripting attacks. The most recent Open for Web Application Security Project (OWASP) list of the 10 most critical web application security vulnerabilities also highlights how easy exploitation remains and shows the dire state of application security.

©2016 (ISC)² Incorporated. All rights reserved.

There are, however, opportunities for improvement. They start with a more information security-centric understanding of what it means when teams and organizations adopt agile software development practices. And to do that, information security professionals should also work to better understand today's application development processes.

UNDERSTANDING AN INCREASINGLY AGILE WORLD

"Agile" software development refers to a style rather than a singular defined methodology. The "Manifesto for Agile Software Development" summarizes agile development's basic tenets, and the "12 Principles Behind the Manifesto" expands upon those core principles.

Based on that manifesto, an agile style of development will include:

- A lightweight empirical process;
- Extensive and regular customer collaboration;
- Self-organizing and self-managing teams;
- Rapid feedback;
- Interpersonal communication;
- Just-in-time work;
- Limited documentation; and
- Measuring success based on working software.

There is no "one true agile"; multiple development processes can and have truthfully claimed that they are agile.

GETTING INTO SCRUM WORK

According to Version One's 10th Annual State of Agile Survey, Scrum is currently the predominant form of agile that companies have adopted (58 percent), with another 10 percent using a Scrum/XP (Extreme Programming) hybrid.

Scrum is a "process framework for complex product development." It focuses solely on project management and says nothing about how the developers have actually written the software. That's one reason developers frequently augment it with XP development practices.

In a Scrum/XP project, work to be done is described as a "user story," a short sentence describing the desired functionality in business terms. It contains a role, a goal and a reason. An example could be, "As a bank customer, I want to withdraw funds from my checking account so I can increase my cash on hand." In a Scrum/XP project, user stories replace all the traditional forms of requirements, such as system functional specifications, business requirements documents (BRDs) or detailed requirements documents (DRDs). Some teams identify user stories before they kick off a project, but they will identify additional stories continuously throughout the project and add these to the backlog. Since a user story describes a piece of functionality, the team can develop and deliver it independently of other stories.

Sometimes described as "a reminder to have a conversation," user stories are placeholders for later collaboration. Teams collect user stories in a "product backlog" and sort them by order of priority. A business user of the system, or their proxy, acts as a "product owner" and is responsible for which user stories are in this backlog and for ordering them based upon their business value. The product backlog contains "everything that the team might work on." All the development work the team will undertake takes the form of a user story in the backlog or as part of a user story. There is no separate project plan or requirements document, aside from the product backlog.

READY TO SPRINT

To deliver software early and continuously, scrum teams work in "sprints," or short iterations. Most teams today are using one- to two-week sprints. In that time, the team agrees to a scope of work for the sprint (based on which user stories from the backlog they agree to deliver). Then team members design, write code, test and deliver the user stories as working software during the same sprint.

Because the user stories are light on specifics, the team needs to have continuous access to the product owner to discuss possible ways to achieve the business goal and to verify that the completed function works to the business' satisfaction.

"Agile approaches leverage 'last responsible moment' decision-making, allowing the latest, complete information and data to guide us," says Lowell Lindstrom, past managing director of the Scrum Alliance and founder of the Oobeya Group, an agile transformation services firm.

"As teams learn to use agile approaches, the skill and discipline of how to defer a decision, and when not to, are critical."

At the start of the sprint, the team holds a sprint planning ceremony and creates a sprint backlog. This sprint backlog contains the user stories from the product backlog the team is committing to deliver at the end of the sprint. As part of sprint planning, the team will discuss with the product owner what the user story entails, and many groups will choose to document these as acceptance criteria. Once the necessary parties commit to the scope of a sprint, the scope does not change.

During the sprint, the team holds a daily stand-up meeting, or "scrum," at which time each person answers three questions: "What did you do yesterday?" "What are you going to do today?" and "Do you have any blockers?" The last question refers to where the team needs help and what kind.

©2016 (ISC)² Incorporated. All rights reserved.

The daily scrum lasts no more than 15 minutes, and it's the primary coordination meeting for the team. It allows everyone to know who's working on what and what obstacles the team is encountering. A person in a formal Scrum role (the "scrum master") takes ownership of resolving any blockers and is responsible for making sure the team follows the scrum process—including protecting the team from outside interference during the sprint. The scrum master, product owner and team are the only roles in a scrum project.

At the end of the sprint, the team holds a sprint review, also known as a "showcase" or "sprint demo meeting," for project stakeholders. The purpose is to show the user stories they have completed, to preview upcoming work, and to gather feedback from the stakeholders. As the representative of the business on the team, the product owner is typically responsible for this ceremony, although that person may call on the team to assist.

More mature agile organizations are moving toward scaling agile practices beyond the team. Many of these approaches, such as the Scaled Agile Framework (SAFe®), incorporate Scrum/XP practices at their core. Understanding how to incorporate security requirements into a Scrum/XP project is therefore foundational for today's security professional.

INCORPORATING SECURITY REQUIREMENTS INTO A SCRUM PROJECT

User stories describe the entirety of work the team will undertake, so user stories must incorporate security requirements.

In a traditional style of development, a security requirement might take the form of a stand-alone functional

©2016 (ISC)² Incorporated. All rights reserved.

requirement, such as "all access to individually identifiable patient data must be logged." Using Scrum/XP, the user story describing where such data is accessed will need to include that sort of functionality.

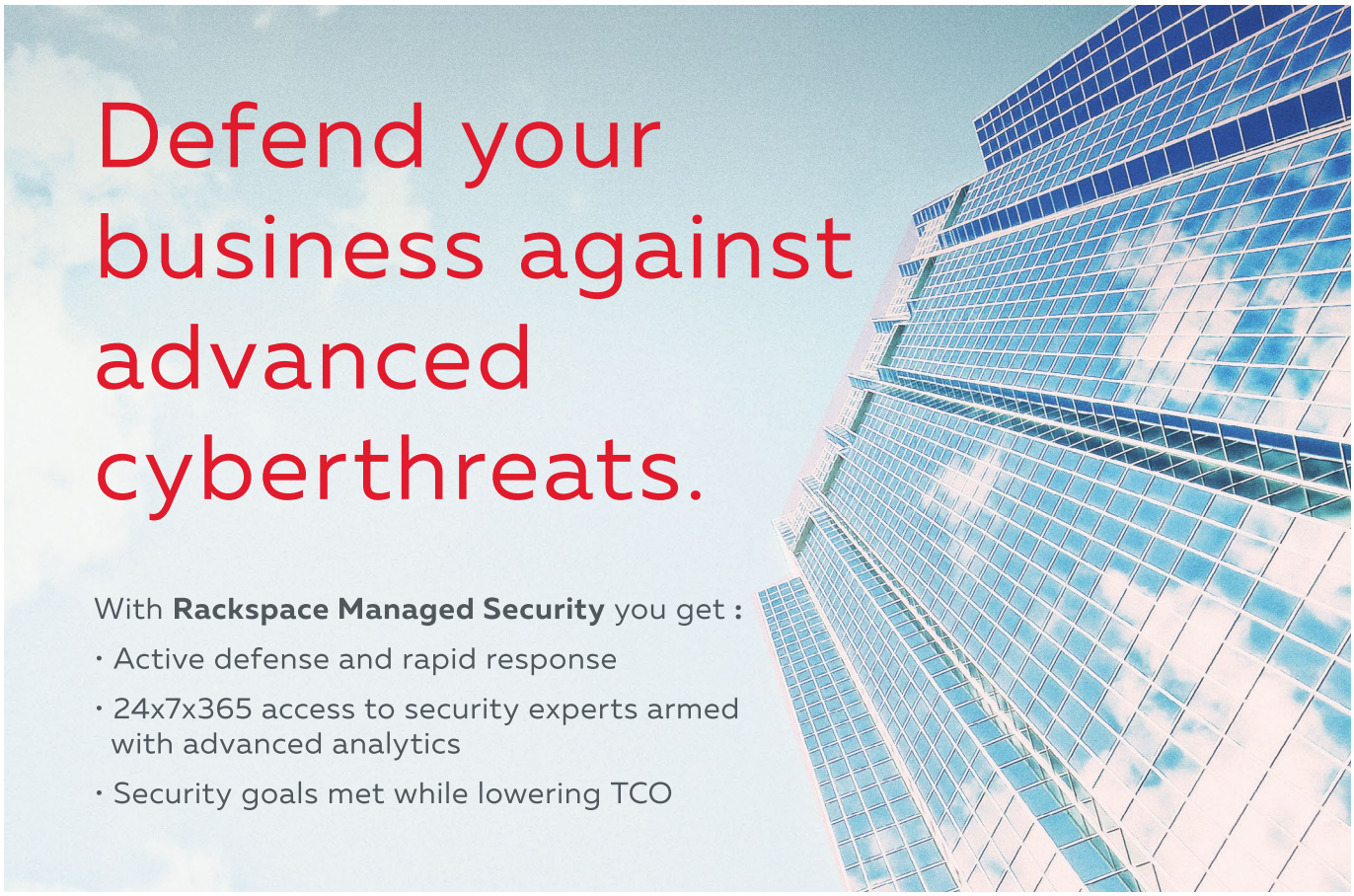
Because the product owner is responsible for user stories—including their priority in the backlog and what it means to be "done" with them—the person filling this role must have an appreciation of the security requirements.

Some teams find value in documenting nonfunctional requirements, such as logging and performance, in a common location for easier reference and updating. These should not be thought of as stand-alone requirements but rather as a reference the team should review as part of the discussion of any user story and its acceptance criteria. Sometimes a given requirement will apply to the story in question. Sometimes the requirement may be slightly different from standard performance requirements. And sometimes it's not in there at all.

When considering how to incorporate security requirements into user stories, it's useful to think of a two-level scheme for security requirements. At the upper level are "capabilities" the system under development should have, such as role-based access control, nonrepudiation or confidentiality of data at rest.

When discussing a user story, the product owner and team also should consider what a given capability means in the context of this story. For instance, they should ask, "Does this story require nonrepudiation?" Any such requirements then become part of the acceptance criteria for that story, and the team will need to verify them before they consider the story complete.

"The development team can automate and test for many of these broader, nonfunctional requirements through a continuous integration process, allowing the security team



©2016 (ISC)² Incorporated. All rights reserved.

AGILE DICTIONARY

BY KEITH McMILLAN

Agile. Any of a number of styles of software project management that emphasizes lightweight empirical process; extensive and regular customer collaboration; self-organizing and self-managing teams; rapid feedback; interpersonal communication; just-in-time work; limited documentation; and measuring success based on working software.

SAFe (Scaled Agile Framework). An agile project management approach that scales scrum/XP across multiple teams and up through business portfolios.

Scrum. An agile project management approach formalized by Ken Schwaber and Jeff Sutherland.

Scrum Master. Role on a scrum team. Each scrum team has one scrum master. The principal responsibilities of the scrum master are protecting the team from outside interference, taking ownership of blockers raised by the team, and assuring that the team follows the scrum process.

Sprint. A short iteration in scrum. Typical industry practice is one to two weeks. All work undertaken by a team happens in the context of a sprint.

Sprint Backlog. The work items from the product backlog that a scrum team agrees to complete during a given sprint. Creating the sprint backlog from the product backlog is the first activity that occurs during a sprint.

Story Point. A unitless measure of relative complexity of a user story.

INVEST. Acronym describing the characteristics of a good user story: (I)ndependent of other user stories, (N)egotiable in terms of how the work may be accomplished, or in terms of how it might be divided into smaller stories, (V)aluable to the business, (E)stimatable for level of effort, (S)mall enough to be worked on in a sprint, (T)estable as to whether the software works as intended.

Iteration. See Sprint.

Product Backlog. The entirety of the user stories that a scrum team may work on. All requirements, ordered by their value to the business. The product owner is responsible for the product backlog.

Product Owner. Role on a scrum team responsible for representing the business. Each scrum team has one product owner. The product owner owns the product backlog and is responsible for working with the team to answer questions about desired functionality and ways it could be achieved and for reviewing and approving complete user stories.

Retrospective. Scrum ceremony at the end of every sprint during which the team reflects on how they can tune and adapt their behavior to become more effective.

Velocity. A measurement of how much work a team can undertake successfully in a sprint based on how much work the team has undertaken in previous sprints. Typically, the average of the number of story points delivered successfully in the previous three sprints.

XP (Extreme Programming). Software development practice teams frequently use to augment scrum, as scrum contains no actual software development practices of its own. ●

to scale across projects," says Ed Bellis, CTO at Kenna Security and former CISO at Orbitz.

"There are hooks into CI servers to use tools such as Gauntlet that allow the development team to perform security testing of these requirements during integration," he continues. "A security team will need to rely on both tools and appointed developers within the scrum in order to scale."

Kathy Marshak, agile transformation coach and SAFe program consultant trainer at Icon Technology Consulting, offers some tips for integrating security-related requirements into a Scrum or SAFe project.

"Ensure that security experts understand agile fundamentals so they can participate in discussions about definitions of 'done' and contribute to reusable acceptance criteria," she says. "Ensure that coding standards integrate security practices [and that] such standards facilitate XP's 'collective code ownership' practice. Encourage developers who are skilled in writing secure code to pair with other developers to spread the secure coding practices."

Because they are business experts, some product owners are not technical and may not understand what security capabilities a system should possess and in which stories they should appear. While a scrum team has a single product owner, it makes no statements about who can provide input to the product owner. In fact, the development team frequently calls upon product owners to take input from multiple stakeholders, and security is another voice providing input.

The product owner and team can call upon security professionals to provide additional information as necessary; however, this requires the product owner to appreciate security's importance in the development process. It's our job as security practitioners to make that case.

By incorporating security capabilities into the user stories in a Scrum/XP project, we dovetail with the way modern application development works. This avoids disruption to the process and incorporates security requirements early and throughout the development process. ●

KEITH McMILLAN, *CISSP, is principal consultant at Adept Technologies. He provides agile coaching, comprehensive start-up CTO services and application security architecture. He is a Certified Scrum Practitioner and a Certified Scaled Agile Framework Program Consultant (versions 3 and 4). He can be reached at kmcmillan@adeptechllc.com.*

©2016 (ISC)² Incorporated. All rights reserved.