

# Corporate vs. Product Security


Philip Watson

# *Corporate vs. Product Security*

*GIAC (GSEC) Gold Certification*

Author: Philip Watson, phil_work@yahoo.com
Advisor: Hamed Khiabani

Accepted: May 20, 2013

Abstract

The typical corporate security employee is tasked to defend and protect the enterprise, including the web servers, the corporate email, and corporate secrets. But there is a very different security focus for those who must design security into the products that a corporation produces, whether that product is an appliance, a software application, or a service. Identifying and separating *product security* teams from the *corporate security* teams is critical for their success. This paper will explain these differences, recommend a relationship between the two, and address scenarios when their borders overlap and recommend how to deal with them.

# 1. Introduction

When people hear "I deal with security" from any employee, the typical thought is that they are defending the enterprise, the web servers, the corporate email, and corporate secrets. But there is a very different security focus for those who must design security into the products that a business produces, whether that product is an appliance, a software product, or a service. These *product* security tasks, if they have even been identified, are often by default assigned to the corporate security teams. But doing so may not be the wisest choice, for conflicts and lack of attention to the product security tasks can result. The first step in dealing with this problem is to understand what the two different security needs are, and why it is often best to have two separate groups fulfilling them.

## 1.1. Corporate Security

The most common understanding today of any a security job is that the primary goal is to keep an organization secure, that is, to protect its secrets and its resources. Typically this involves building and maintaining protections around those secrets and resources. These can include walls and doors to prevent access except through controlled access points, access control systems to ensure that only authorized users have access, traffic monitoring systems in case someone gets past the access controls, and alarms. All of these types of protections can be either physical or network mechanisms, e.g. a concrete wall or a network firewall.

Some organizations split the physical and network security tasks between two teams, sometimes called *facility* security and *network* or *IT* security. The goals of these groups are still the same, to protect the organization's secrets and resources. The threat vectors they focus on are different, one on physical access, the other on virtual access via a computer network. While there may be conflict between these groups, perhaps due to the different culture that the training for each induces– the strong men versus the nerds– they should cooperate with one another. Often an attack will involve both vectors, such as physically penetrating an organization far enough so that a less restricted network access point can be utilized. And often by design physical security depends at points upon

network security, such as a fence surrounding a facility except for at the gate, where a networked access control system authorizes and records those who enter.

But conflicts between physical and network security teams is not the focus of this paper. They have the same goals, and clearly must cooperate with one another to achieve those goals. There often is another security team that has a wholly different goal.

## 1.2. Product Security

For businesses that produce a product, or operate a service, *product* security is the measure of how secure that product or service is. For products security can go beyond the traditional information security CIA triad–confidentiality, integrity, and availability (Perrin, 2008)—and may include safety. Products may have different security goals; for example, a smart phone needs to protect the user's information from thieves and malware; a webmail provider needs to protect the user's information from hackers, protect the user from phishing, and protect the user from malicious ads; a USB device should be free from malware (Krebs, 2008); an in-flight entertainment system needs to protect itself from malicious users and keep its financial transactions and digital content secure; a modern automobile needs to keep its remote monitoring/alerting/control system from unauthorized access; a modern aircraft needs to protect itself from unauthorized influence and control (Constantin, 2013). The list can go on and on. There is a strong expectation that products are secure, due in large part to the interconnectedness of products today, and someone should be ensuring that that is the case.

If an organization is enlightened enough to create teams charged with designing and testing for security in their products, they are most often different from the corporate security teams. Frequently they are simply part of the product engineering teams, but sometimes they are separated out and may be even given different job titles, such as security engineer, or security tester. This task is too often overlooked and given only an afterthought, perhaps after the first security event involving the product is reported from the field. It is best practice to consider security during the design process, ensuring security is engineered into the product from the start.

In his seminal work (Anderson, 2009, p.3), Ross Anderson defines *security engineering* as: building systems to remain dependable in the face of malice, error, or

mischance, and focusing on the tools, processes, and methods needed to design, implement, and test complete systems, and to adapt systems as their environment evolves. While this definition could be used for those ensuring corporate security, it is best applied to those ensuring product security, because product security is more concerned about *building* systems. The emphasis of corporate security is *protecting* information and resources, not building systems. Yes, to protect the corporate security teams may need to build protection systems, but that is not a constant task. Product security teams are always building secure products. That is what they always do, so the term *security engineer* is best applied to them.

Security as a job focus got it start in corporate security. Most books, classes, and training emphasize the corporate side (Campbell, 2011 and Allen, 2005). Too few emphasize product security. It is time that product security and security engineering rise to the front. Indeed, Bruce Schneier argued this same point in a recent blog post (Schneier, 2012), "We need to establish security engineering as a valid profession in the minds of the public and policy makers."

Having established the necessity of teams responsible for product security, should they be part of the corporate security teams or separated? Does it really matter that much?

# 2. Separated Product & Corporate Security Teams

When a physical product is being delivered to customers, the team responsible for product security should be separated from the corporate security team. This recommendation is based on experience, and will result in the best operation of the two teams.

## 2.1. Different Goals

As stated in the previous section, product security has a different focus than corporate security. The former is concerned with building and delivering secure products, while the latter is focused on protecting corporate information and resources. These different goals would lead to inevitable conflicts or compromise of one goal for the sake of the other during crises or periods of limited resources.

For example, two security incidents may be reported at nearly the same time, one from within the corporate network, and another arrived via the product support team. Both are urgent and could cause loss of customer and corporate secrets. If there is only one security incident response team, one incident will be left alone to fester while the other is addressed. Or for another perhaps more realistic scenario, should the security team spend time writing security requirements for their next product, or should they research and evaluate the next generation of networking products in order to upgrade their corporate network and avoid slowing down operations and development? Even with enough resources, a single team will tend to work together on one goal at a time, and a product may ship with latent security failures.

## 2.2. Different Skills

The skills required in the product security teams may be quite different from the skills in the traditional corporate security team. Producing a product involves all the phases of engineering, from requirements gathering, to design, to implementation and reviews, followed by verification testing and regression testing. Security engineers should be involved in all phases of this work, from writing security requirements, to code reviews, to testing products for residual, otherwise undetected, vulnerabilities. Corporate security teams rarely need to be involved in the engineering of products in this manner; most often they buy off the shelf appliances and the design is only in the integration of all the parts—network design. If a company is truly involved in designing and building products, it needs a dedicated product security team.

Some products require specialized skills or training. For example, if the product collects credit card data (such as a portable payment device, or a fixed payment terminal, or a smart phone application that does so), it is likely subject to requirements from the Payment Card Industry (PCI), in particularly the Payment Application Data Security Standard (PA-DSS). If so, the product security team should be skilled in the PA-DSS requirements and in the process of obtaining certification for the product, and may even wish to become a trained and certified qualified security assessor (QSA) who can conduct their own on-site assessments and be approved by the PCI Council. Unless your corporate infrastructure also processes credit cards (in which case it would be subject to the PCI-

DSS), the product team would have this unique requirement upon them. Best to dedicate a person with such skill and knowledge to the product security, not wasting it by sharing their time with corporate security tasks.

### 2.3. Different Budgets

When your product security team is merged with your corporate security team, they will share the same budget. Because of their different goals (see §2.1) they will encounter conflicting demands during purchasing decisions, such as, should we buy the bigger router or the automated security code reviewer tool? Of course ideally a security department would have the budget for all of that, but even given large amounts of money, the differing goals will result in conflicts resulting in one of the goals suffering for the sake of the other.

Separated teams allow the two budgets to be created and consumed without concern to the other. In fact, if the product security team is organized within the engineering department, it may be that they would enjoy a larger budget than the corporate security team, especially in a firm that emphasizes product development, that is, "run by engineers". This can lead to resentment from the slighted team, but it is better to have separated teams and budgets than one security goal that never gets any budget allocation.

## 3. Integrated Product & Corporate Security Teams

Having established why it makes organizational sense to separate your corporate and product security teams, there are still some situations for which integrated teams would be better. In general, this occurs when the product is either an Internet service, or when the product requires a networked service to operate, either all the time or periodically calling "home". In these cases it can become hard to differentiate the product security, or at least portions of it, from corporate security. When such lines are hard to draw, it is most efficient to integrate the product and corporate security teams.

## 3.1. Web-Based Product Example

A primary class or product ripe for joint corporate and product security teams is a web-based service offering, such as a cloud-based storage service, or a webmail product, or when the website of the company *is* the product. In such a case the servers for the product may be the same servers as for the corporate website, or at least share many components. In addition, the tools and techniques used to secure the product website are likely the same as those used to secure the corporate website, so the skills and purchasing decisions will significantly overlap between the product and corporate security teams. In such a case it is most efficient to make them the same.

## 3.2. Splitting Product Responsibility

However, the case for merged teams is not always so straightforward. In some instances it may be better to split the responsibility for the security of the product between two teams, one responsible for the part of the product that exists *outside* of corporate control, and another for the part, typically a server or back-end, that exists *within* corporate control. A product team for the latter part could be integrated with the corporate security team because of the skills and tools overlap, but the former may need to remain separate. A thorough review and decision should be made be management.

### 3.2.1. Client-Server Product

An example of such is a product that must communicate routinely with a common server base, such as a MMORPG, or an app front-end to a web-based service such as an email app. These products have two components, the client-side product and the common server-side, commonly called “the cloud”. Clearly the server-side product overlaps with corporate security skills and budget demands as examined previously. That part of the product at least should be located within the corporate security team.

But what of the client-side part of the product? Depending on the target client types, it may have a very different security context than the server side of the product, so one could argue that it should have a specialized product security team, maybe even one for each client type (ex. iOS, Android, Windows, MacOS, etc.). But it could be that the product and the client environments are similar enough, such as both run and are limited

to the same operating system, to warrant keeping the product security team within and under the corporate security team.

If the responsibilities for the two parts of the product are split between two teams, the security design for the communications between server and clients should be clearly identified as the responsibility of one of the two teams, with the other team subsidiary. To do otherwise would result in design conflicts between the two teams, or worse, if no team is identified and each side only looks at their "external" interfaces, there could be no security design given to the communications part resulting in unintentional information disclosure between the server and clients.

### 3.2.2. Calling Home

Another type of product to consider for how to split security responsibility between corporate and product teams is one that only periodically calls home, perhaps for updates or downloading performance metrics. Vehicle-borne systems are one example; a self-updating networking or entertainment appliance is another. For these, the overlap with corporate security is only in the portion that calls home. The reason that it calls home impacts the decision as to whether or not to make the server side the responsibility of the product or the corporate security team. If it is doing so to relay sensitive information that has special processing or certification requirements (ex. payment information, personal and private data) it would be best to keep the skills, knowledge, and tools to handle that within the product team by having them responsible for both the client and server sides. But if it is merely doing so to download logs, or check for software updates, the server side could be kept within the corporate team's responsibility.

Whatever the decision, it should be clear to both parties where responsibilities lie. A conscious decision should be made, not just assuming that all server components are the responsibility of the corporate team. Assumptions will result in conflicts between the teams, especially if one team is responsible for some certification process of a system while the other team is responsible for implementing the security measures that are being certified.

## 3.3. Organizational Recommendations

The first step to take is to recognize that product security is not necessarily the same as corporate security. Evaluate the special requirements for your products and decide if that warrants creating a separate team or teams. If any product differs enough from your corporate security environment, do create a separate team for it, and don't give them the same group or department name as the corporate one as it could lead to confusion having two "security teams". Leave corporate security with their established name, if they have one, and give the product team a unique name, perhaps "product security" or "security engineering". Forrester recommends even more (Kark & Dines, 2010), splitting into four security groups: security oversight, IT risk, security engineering, and security operations. This only makes sense for large organizations. Smaller organizations might merge oversight and operations with either of the two other groups, but do create two security groups.

Once you have decided for a separate product security team, the decision as to in what larger organization to place them will have a strong affect upon their chances of success. Don't place a product security team under your corporate security team. Doing so risks the conflicts mentioned earlier in §2. Similarly, don't place the two teams under the same management, or at least the particular risk of budgetary conflicts (§2.3) arises. The best organization is to place them within your engineering group, which is assumed to be separate from the corporate security team. The best product security team members are engineers after all, security engineers. It is here that they can influence the design and implementation of the product to build in security.

In his book by Cisco Press (Rajnović, 2011, chapter 8), Rajnović offers three locations for the product security team: engineering and development, testing, and technical support. Because the focus of Rajnović's book is in discovering vulnerabilities and responding to them, indeed half of the book being devoted to incident response, he offers these three locations rather than the one that results from a view of the whole product lifecycle. If your product security team primarily responds to issues discovered after delivery, then put them in technical support. If you want to catch the problems *before* release, put them into your testing group. However, it is best to focus on building

secure products the first time, so the best location to place them is in the engineering organization.

To avoid the dangers of self-policing the product security team should not be directly within software development team. The error of self-policing here would be to classify vulnerabilities as minor in order to ship the product on time. Placing the security team within the software team also risks them focusing solely on software errors, when there can be and often are manifested as vulnerabilities in the installation or operation of a product. It is best to give the product security team authority to both specify and check, that is to produce requirements for the rest of the product engineers and to test for security compliance. The testing should include authority to do source code inspection to catch coding errors, often security vulnerabilities. If your organization is large enough to have a systems engineering group, put them there, for here they can place a focus on security from the beginning of the product to the end.

If your organization is enlightened enough to have a Chief Security Officer (CSO), both groups should definitely report up to him. A CSO should be knowledgeable enough to balance the different security needs of IT, facilities, and product teams, so they could all be directly under her. On the other hand, an indirect, matrix-level reporting could work as well.

## 3.4. Process Recommendations

Now that the decision has been made to have a product security team, how and where should they be integrated into the product development processes? In short, they need to be involved at all phases, beginning from requirements development all the way through to maintenance.

During the requirements development phase the product security team needs to develop and provide security requirements. If this is done properly, it will minimize the work required in the successive phases. Don't allow them to simply say, "The product needs to be secure." Each requirement should be specific and testable. The team should research, document, and clarify any regulatory or external requirements. It's likely that the team, whose sole focus is security, will be aggressive and define many security requirements that some may view as unnecessary or overboard. To balance that the

product security team needs to be involved in the inevitable cost-risk tradeoff discussions that will occur with the hardware and software engineers, helping to decide how important and at what increased cost (size, space, heat, money, etc.) each requirement can be justifiably added in.

Research should be done on the proper way to implement certain requirements, and design guidance should be given to the developers. For example, without guidance the development team may not know the proper way to implement OATH (Goldshlager, 2013), or the proper method to hash authentication passwords in web applications (Skoudis, 2013). So the product security team should write detailed requirements down to the design level, and in addition should participate in design reviews to ensure proper implementation. Be sure to include them in the required attendee list for such reviews. By doing so, they should catch design implementation errors, which are often where vulnerabilities are created.

In addition, the product security team should participate in peer code reviews. This is not merely searching for common security coding errors—automated code checkers can do that and indeed should be utilized—but the product security team needs to look for errors in the design of the code. To do this they must of course have some software development training. If some of the security of the product depends on software, make sure that at least one member of the product security team has software coding skills.

It should be emphasized that functional testing alone will not catch many security vulnerabilities. Just because a product allows a user to log in doesn't mean the password was protected properly when sent to the server, or that the proper number of PBKDF2 iterations was performed ("On hashcat and," 2013; Steven, 2013). Having produced the detailed design requirements for security, product security team needs to do more than "toss them over the fence" and hope the engineers implement them correctly. They need to "track [their requirements] till closure…[which is] when implemented as per the security team's expectations" (arD3n7, 2013). So do ensure the product security team participates in design reviews, in order to save them the trouble of reverse engineering design errors after the fact.

Once the design reviews and done and developers believe the product is working, the product security team then enters into the next, more traditional, security task of checking, scanning, and trying to find vulnerabilities. The security team's testing could be independent, or could be integrated as part of the normal product testing activities. It should be emphasized that the goal of this testing is "test to pass" but to find out if the product can be broken, and if so, if that break can be exploited. To earn credibility for their test reports, some vulnerabilities should be exploited and "flags" (objects or information of value) obtained to prove the seriousness of the vulnerabilities. This is classic security penetration testing.

Finally, once a product is approved and delivered to customers, the product security team is not done. Now is when continuous vulnerability management occurs (Rajnović, 2011). If the product team has done its work in the earlier phases, very few new vulnerabilities will be found in service. But there will be some, especially if large public codebases are integrated into the product, such as third party operating systems. The product security team needs to analyze problem reports from the field and flag as security related those that should be. These along with any public vulnerabilities that impact the product should be assessed and categorized with a severity and priority in order to schedule if and when a fix should be made.

Don't forget to include the product security team in ongoing maintenance of a product, especially if any redesign is being performed. Consider redesigns as new products and restart the security design cycle.

## 4. Conclusions

If your company is designing products, do consider building security into the product before release. Don't wait for the public and embarrassing notice that you forgot to make your product secure. Then, recognize that the skill and task requirements for designing in security may be vastly different from what is needed by a typical corporate security team. As result you may need to build a separate security team dedicated to product security. Don't make the mistake of assuming that all security work is the same, because it is likely that one will overshadow the other, leaving either your product or

your corporate information insecure. Also recognize where the product and the corporate realms may overlap, and allocate responsibility accordingly.

It may be that you have different products with different security needs. Recognize these and if necessary create separate product security teams, or if there is sufficient skill overlap, assign responsibility for a certain product's security to your corporate security team. For some products, like purely web-based products, you may have sufficient overlap that you may not need a dedicated product team.

If you have created a separate product team, nurture it. The better they work, the less public incidents you will have. Build them into the entire product development lifecycle, from cradle to grave. Don't neglect that a security measure that was sufficient when it was designed may no longer be so after some time in service or after some external development (ex. a tool that attacks your encryption, faster computers, a new vulnerability discovered). Incorporate security lessons learned into the next product; don't repeat the same mistakes twice. To ensure this, treat your product security team well so that they would rather stay than take their acquired skills elsewhere.

By following this guidance, at least your organization can be spared any public embarrassment for security failings, and hopefully rather become known for producing well-made, secure products.

## 5. References

Allen, J. (2005). *Governing for Enterprise Security*. Informally published manuscript, Networked Systems Survivability, Software Engineering Institute, Carnegie Mellon University, Pittsburg, PA, Retrieved from http://www.cert.org/archive/pdf/GESppt.pdf

[Web log message]. (2013, April 16). On hashcat and strong master passwords as your best protection. Retrieved from http://blog.agilebits.com/2013/04/16/1password-hashcat-strong-master-passwords/

Anderson, R. (2009). *Security engineering, a guide to building dependable distributed systems*. (2nd ed.). Indianapolis: Wiley.

arD3n7. (2013, March 14). Building security in requirements [Web log message]. Retrieved from http://resources.infosecinstitute.com/building-security-in-requirements

Campbell, G. (2011). *Measures and metrics in corporate security*. Retrieved from https://www.securityexecutivecouncil.com/content/measures_metrics_mini_200801.pdf

Constantin, J. (2013, April 11). Vulnerabilities in aircraft systems allow remote airplane hijacking, researcher says. *ComputerWorld*, Retrieved from http://www.computerworld.com/s/article/9238320/Vulnerabilities_in_aircraft_systems_allow_remote_airplane_hijacking_researcher_says

Goldshlager, N. (2013, April 3). The unfix bug in Facebook OAuth - break security [Web log message]. Retrieved from http://www.breaksec.com/?p=6039

Kark, K., & Dines, R. A. (2010, May 10). *Security organization 2.0: Building a robust security organization*. Retrieved from http://eval.symantec.com/mktginfo/enterprise/articles/b-article_security_organization_20_building_a_robust_security_organization.en-us.pdf

Krebs, B. (2008, January 28). Best Buy digital photo frames shipped with virus. [Web log message]. Retrieved from http://voices.washingtonpost.com/securityfix/2008/01/bestbuy_digital_photo_frames_s.html

Perrin, C. (2008, June 30). The CIA triad [Web log message]. Retrieved from http://www.techrepublic.com/blog/security/the-cia-triad/488

Rajnović, D. (2011). *Computer incident response and product security*. Indianapolis, IN: Cisco Press.

Schneier, B. (2012, September 15). The importance of security engineering [Web log message]. Retrieved from https://www.schneier.com/crypto-gram-1209.html

Skoudis, E. (2013, April 05). Pass-the-hash web style [Web log message]. Retrieved from http://pen-testing.sans.org/blog/pen-testing/2013/04/05/pass-the-hash-web-style

Steven, J. (2013, March 20). *Password storage cheat sheet*. Retrieved from https://www.owasp.org/index.php/Password_Storage_Cheat_Sheet

Last Updated: November 27th, 2020

# Upcoming SANS Training

**Click here to view a list of all SANS Courses**

| | | | |
|---|---|---|---|
| **SANS Essentials Australia 2021** | **Melbourne, AU** | **Feb 15, 2021 - Feb 20, 2021** | **Live Event** |
| **SANS OnDemand** | **OnlineUS** | **Anytime** | **Self Paced** |
| **SANS SelfStudy** | **Books & MP3s OnlyUS** | **Anytime** | **Self Paced** |