SAND2015-7014

SANDIA REPORT

SAND2014-1889 1 J
Unlimited Release
Printed June 2015

Attributes of Securable Architectures

Christopher C. Lamb and Jesse P. Hatcher

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online ordering: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2014-1889 1 J
Unlimited Release
Printed June 2015

Attributes of Securable Architectures

Christopher C. lamb
Cyber Analysis Research & Development Solutions
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-0933
cclamb@sandia.gov

Jesse P. Hatcher
Engineering Infrastructure
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-0933
jphatch@sandia.gov

Abstract

As cyber-security is becoming more and more important in systems development, engineers have begun to recognize and understand the types of errors they can introduce through hurried coding technique and design. This overall trend is certainly moving the software industry in the right direction and can lead to developing higher quality software-centric systems. Unfortunately, we have barely begun to examine the results of poor architectural choices, nor do we have much insight into what secure and securable architectures look like. In this paper, based on the past 40 years of work identifying specific security principles, we create a taxonomy of principles that address the abstract cyber-security needs of systems. We then tie these principles to studies of insecure systems architectures to demonstrate applicability. We close the paper with a description of other cyber-security taxonomies, how they specifically differ from this presented taxonomy, and add new principles to address gaps shown in taxonomic comparisons.

Acknowledgments

Thanks to David Duggan for editorial guidance, and to Dr. Tod Amon, Steve Letourneau, and Mark Lynam for help with derivation and review of the presented taxonomy.

Contents

This page intentionally left blank.

Introduction

Securing underlying architectures are vital to the development of secure systems. Without appropriately securing the underlying foundation of a system, the developed system is relatively easy to undermine. We see specific examples of this kind of compromise frequently, in a variety of different types of systems including authentication protocols once used as the foundation of system user identities [16].

In 2014, Santamarta presents a small overview of communication vulnerabilities in today's satellite communication systems. He categorizes the severity of the problems they found, by vendor, product, and service. While he presents possible attack scenarios that could compromise the satellite communication equipment, he does not specifically release exploits [15].

Interestingly, his work does classify the possible exploits in a clear taxonomy. Overall, these kinds of devices are remarkably insecure, the result of a variety of bugs and design decisions impacting system confidentiality, integrity, and availability. Key design mistakes include hard-coded credentials and backdoors, homegrown, undocumented, and insecure protocols, and weak native security controls like password reset capabilities. These problems lead to attacker capabilities including communication compromise, system denial-of-service, and system geolocatability.

Koscher et. al. go into detail on their methodology in testing two late-model cars for security vulnerabilities within the car computational plant. They open the paper going into detail on why these networks exist in cars today, and how they originally were introduced. They then show what kinds of Electronic Control Units (ECUs) exist in typical cars, how the Controller Area Networks (CANs) interoperate, and how recent advances in customer focused computational integration are creating new exploitable attack surfaces for attackers. They then describe specific tools they created to help with this work, the systems they were able to compromise, how they were compromised, and how this compromise was verified [10].

These are specific examples of general problems today — we are beginning to understand how to build secure software, but we are not very good at assembling that software into secured systems. These examples both show how neglecting specific architectural principles result in insecure systems that may be running secure software. The first example shows how neglecting end-to-end perspectives, not providing inclusive authentication and encryption, and not following open design principles can result in easily compromised systems. The second shows how design pressures and systemic misuse of trust results leads to similar results.

The remainder of this paper will cover the most significant work defining architectural security principles. We will then organize the principles into a simplified taxonomy for easier application, compare that taxonomy to other existing taxonomies, and add specific principles based on this comparison that are currently un-addressed in the corpus describing cyber-security principles.

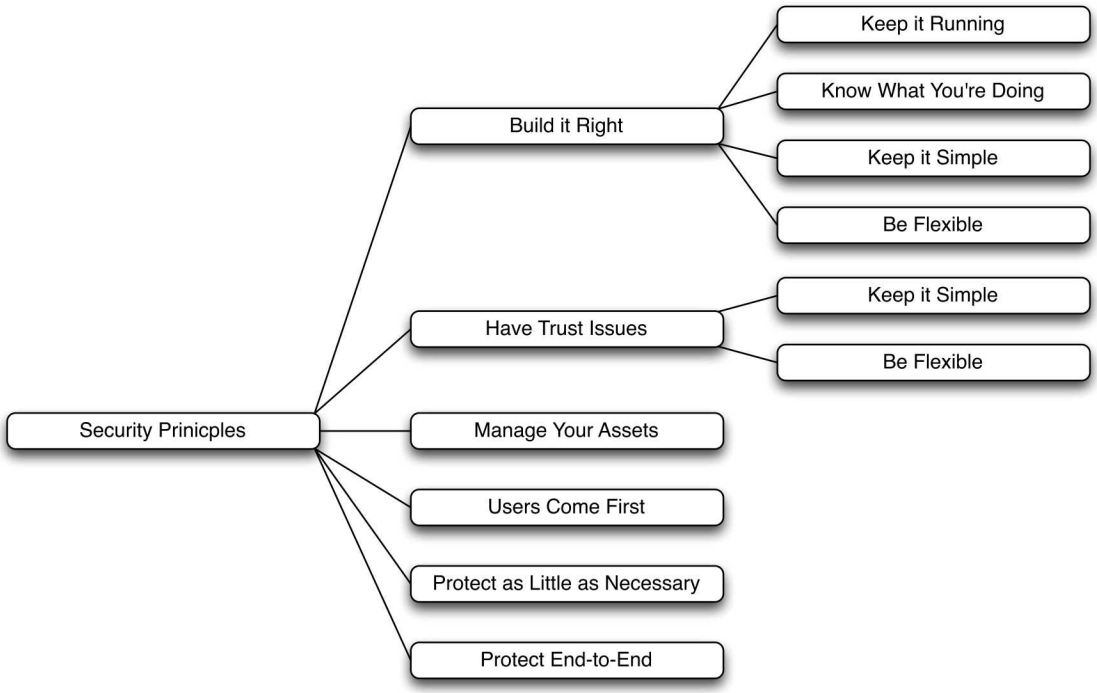

Figure 1: A possible taxonomy of security architectural principles. Here, we show the potential categories without specific principles.

Principles in Literature

Jerome Saltzer and Michael Schroeder wrote the first paper describing information protection principles in 1975 [14]. This paper contains not only the principles they outline, but initial definitions of terms in common security parlance today. They outline both ten principles for information protection as well as many examples of how those principles could apply. While the examples themselves are certainly dated by today's standards, the principles themselves have stood the test of time and have influenced almost every engineer attempting to outline relevant security principles.

Rein Turn and Willis Ware wrote a similar paper in 1975 addressing security and privacy in computer systems [1]. Here, Turn & Ware were more interested in how to secure and maintain the privacy of stored information, protecting it from various white-collar criminals with an eye toward fraud or embezzlement. They furthermore address the very real concern of privacy of personal information maintained by government and large private enterprises from both a confidentiality and privacy perspective, differentiating between the two.

Dorothy and Peter Denning identified the need to strongly secure data and data access in 1979 outlining the emerging threat to information from organizational insiders [7]. They correctly identified the growing future trend of online theft as well as it's large impact. They extended this thinking to identify and define groups of internal security mechanisms that could regulate cyber-system internals, protecting stored objects, information flow, and inferred information, and demonstrate the use of the proposed controls via specific hypothetical cases.

Charles and Shari Pfleeger's text *Security in Computing* was originally published in 1989 and is currently in it's fourth edition [12]. This text marks the transition of principle development from academia to more practically inclined audiences as previously developed principles become less a subject of academic research and more a point of day-to-day system development. The overall focus of this text is computer system security, and it covers subjects ranging from programs, to cryptography, to computer networks. More of a general-purpose than a specific, specialized text, *Security in Computing* still conveys a difficult subject well in some detail. The principles Pfleeger & Pfleeger outline have a slightly different perspective from other principles however as they are not system-centric, but rather take the perspectives of the attacker and defender into account.

Jerome Saltzer, the primary author of *The Protection of Information in Computer Systems*, and M. Kaashoek wrote *Principles of Computer System Design* in 2009 [13]. Though primarily focused on general computer system design principles and concepts, the text still addresses security, revisiting many of the original principles as well as introducing new ones.

In 2011 Richard Smith wrote *Elementary Information Security* to fully comply with NSTISSI-4011, an at that time new standard for federal cyber-security education. The resulting text covers topics ranging from file systems to identity management to networks and encryption [18]. In the following year, Smith reviewed the principles he used in the text and compared them to Saltzer's original list, also tangentally referencing some similar work from previous years [17]. The resulting list essentially adds a couple of principles to Saltzer's original list, and then contemporizes the

original principles that still seem to have direct application today. He does however take the somewhat contradictory stance that certain principles, like complete mediation, that are not in current use should be dropped even though they may contribute to other principles he does support, like Defense in Depth.

Finally, in 2014, the IEEE released a report on *Avoiding the Top 10 Software Security Design Flaws* [9]. Just released, this paper is the product of collaboration between industry and academia, and was released late in the year. As opposed to much guidance in industry today that addresses implementation, this report contains advice for system designers on how to design security into systems. Furthermore, while many papers describing principles trace their legacy back to the original Saltzer paper of 1975 [14], this collection of design points is more independent, though doubtlessly still influenced to some extent.

Overall, these are the key contributions to the area. We have other potential sources, but they generally parrot Saltzer's original work, with slight changes for context, and contributions from Smith or other authors [3, 8]. Overall, this gives us a variety of potential principles we can adopt for day-to-day use, but too many for realistic application. In fact, many of the principles themselves are less principles and more tactics. For example, if a principle is an underlying truth of architectures in general, it should be expressible in a variety of contexts in a variety of different ways. While Pfleeger & Pfleeger's principle of the weakest link has multiple possible expressions in different environments, from physical systems to cyber-systems, Turn & Ware's principle of Encryption does not. In order for these ideas to have wider applicability, they need to be transformed into something more memorable, more terse, and more clear.

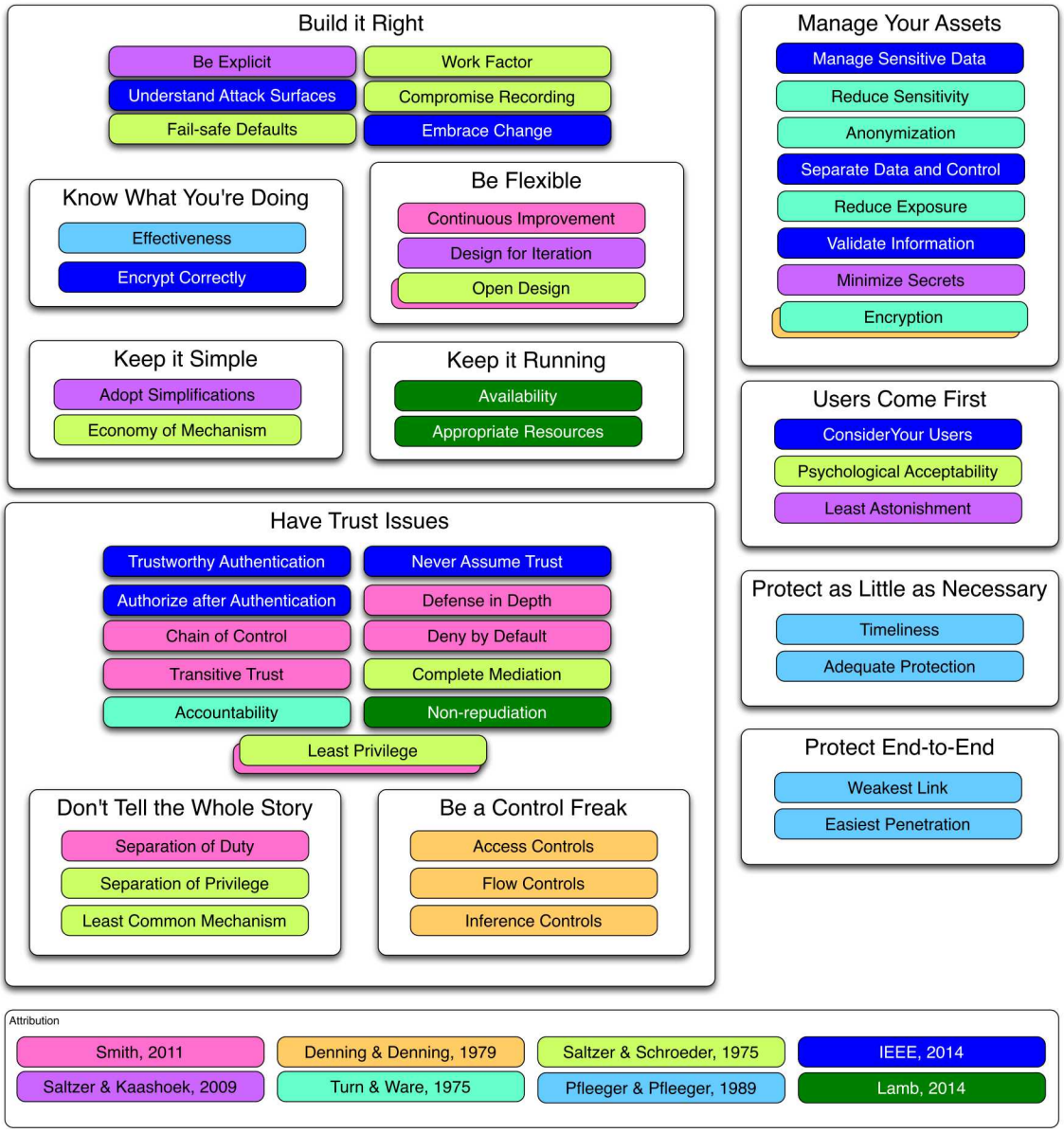

Figure 2: Taxonomy details, showing principle affiliations.

Taxonomy

Our proposed taxonomy is just one possible way to organize these principles. In this particular example, we've attempted to organize the principles by subject area. We have been able to partition the principles into six specific categories, with six sub-categories, as shown in Figure 1. We have mapped principles into these categories, and these mappings are shown in Figure 2.

Build It Right The first level of the taxonomy contains six primary elements. The first, *Build it Right*, addresses all the principles that specifically address construction. These include any principles describing how you should build systems to promote securability as well as how those principles should be applied for maximum protection. The overall direction for this category is to build systems that fail securely, have secure default settings, are as simple as possible, are able to support transparent design processes, and can evolve and improve over time. It is also vital that engineers understand how their systems can be attacked, and that the systems are able to appropriately track and log events for auditing and future forensics.

It's vital that designers know what they are doing. Systems frequently suffer from compromise because of adoption of ineffective controls or misapplication of otherwise secure encryption algorithms [11, 9]. If system designers do not have a clear, in-depth understanding of encryption and control application, they need to provision consulting that does. Encryption can frequently be problematic as valid, strong encryption algorithms can be rendered useless because of weak initialization vectors or pass phrases.

Simple systems are much easier to review, maintain, and understand than complex ones. They can usually be updated more easily, and can be tested more quickly. Complex systems hide vulnerabilities that can lie dormant for years prior to exploitation [5]. In order to keep systems simple, engineers must design them to be simplified over time as technology advances.

Finally, systems must be designed to be flexible and to to be updated regularly. This not only supports the ability to change the system functionally because of market pressures, but also provides engineers with the ability to rapidly change system components in response to security threats. After all, the security landscape can change rapidly and unpredictably, and systems need to change just as rapidly in order to stay uncompromised.

Have Trust Issues *Have Trust Issues* deals, not surprisingly, with how trust is managed and passed through systems. This category addresses how systems should extend, verify, and propagate trust. Generally, systems need to have a strong basis for trust, through trustworthy authentication and authorization primitives. They need to understand exactly what other components they should trust and why, and be able to authenticate and authorize other agents, whether those agents be other systems or users. Some of the approaches are contradictory, like *Complete Mediation* and *Chain of Control*, but allow engineers to apply different types of trust in different situations. By and large, this section advises designers to provide strong authenticaiton and authorization, to clearly understand what should be trusted, why, and what should not, to provide extensive internal controls

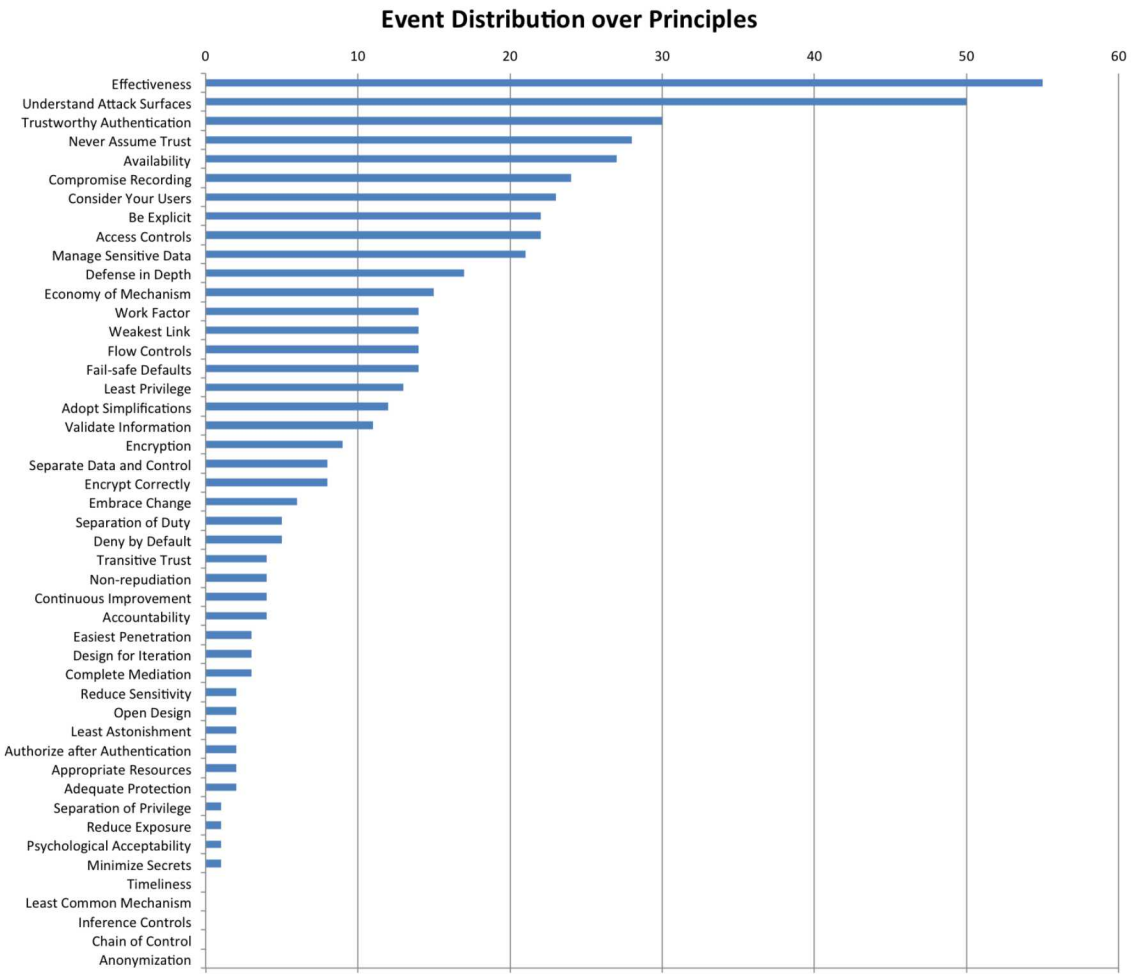

Figure 3: The distribution of events over the defined taxonomic principles.

rather than only perimeter defenses, and to clearly manage and minimize user privilege.

This category has two sub-elements. The first, *Don't Tell the Whole Story*, specifically addresses privilege minimization and common components. Users should be given the lowest level of privilege possible, while still allowing them to fulfill their responsibilities. This way, users are less likely to have privileges they can abuse to overreach their authorization. Likewise, if their accounts are compromised, the damage those accounts can do is minimized. Multi-tenancy is likewise risky. Shared infrastructure provides a vector through which attackers can impact other, unexploited systems.

Engineers need to understand the different types of controls they can use when applying principles like *Complete Mediation* or *Defense in Depth*. The general classes of controls identified in literature include *Access Controls*, *Flow Controls*, and *Inference Controls*

Manage Your Assets A large group of principles have been assembled over the past few decades that address how you should manage and control information assets, ranging from single assets to groups of data collected by an entire organization. Principles also address both data in storage and data at rest. Essentially, these principles address minimizing the amount of data you protect, anonymizing, desensitizing, and encrypting that which you must retain, and validating and partitioning all data flows. After all, the less you need to protect, the better you can protect it.

Users Come First Some common principles address user interaction and semantic acceptance. Interestingly these principles are commonly used in user interface design as well. In this case, they address not only user interfaces, but system interfaces of any type, including programming interfaces or other system interfaces that are only expected to be used by external components.

The primary focus of these principles is to ensure that security services are aligned with user preconceived expectations, and that they take very little effort, if any, to use. All users, whether they are using the system or developing software that takes advantage of exposed APIs are under pressure, short on time, and will cut corners. If security controls are too dificult to use, they will simply circumvent them or not use the system rather than spend significant time configuring them to work correctly. Security systems must be designed with this in mind, or they will be ignored or marginalized.

Protect as Little as Necessary Realistically, sensitive information need only be protected *while it is sensitive*. Likewise, it needs protection corresponding to its value. Providing protection to information beyond its value is irresponsible and inefficient, using resources to protect assets that need no protection when those resources could be better used to protect assets with real value. Information assets should be protected only while they need protection, no more, no less.

Protect End-to-End A ship is only as water-tight as the least water-tight spot on its hull. Likewise, a system is only as secure as its least secure component. That component could be a database, middleware, a client component, or communication system. A common problem is neglecting the security of the so-called ”last mile” of a distributed system [11]. In large-scale networks, this refers to the last link prior to a networked location, like a home or a business. In distributed applications, this refers to the last link (or some series of last links) with a termination at the consuming system. The consumer could be a software component using an API, or a client accessing a system via a browser. Granted, the severity of the breach can range from just exposing that user's information, to exposing credentials that can be used to attack the core of the system itself, but in either case, the security of the system as a whole is dependent on the security of this final link.

Taxonomic Validation

In mid-2015, we convened a group of subject matter experts to analyze the results of just under 400 events recorded from 65 separate red team engagement reports generated from work performed

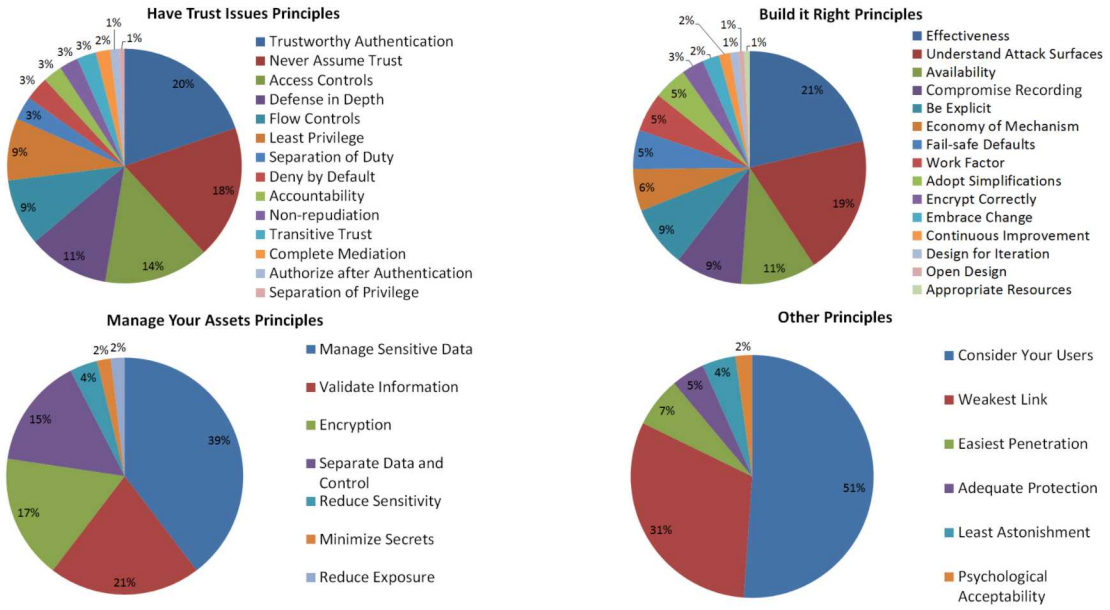

Figure 4: The distribution of principles over the examined security events.

by Sandia National Laboratories since 2008. We looked closely at the recorded vulnerabilities and risks found, and categorized them according to where the flaws fit within our proposed taxonomy. Each event could be associated with zero or more elements from the taxonomy.

Figure 3 shows the distribution of event classification. Interestingly, of the top five principles, four of them can be directly addressed via additional security education for engineers. Effectiveness is directly related to the correct application of controls, in a way that specifically addresses a possible vulnerability. Understanding Attack Surfaces is likewise a skill that comes from education and experience. Trustworthy Authentication and Never Assume Trust stem from misunderstanding how authentication should work in a computer system, and frequently arises from implementing homegrown, non-standard, or standardized but weak authentication schemes. Trust assumption problems, specifically, can arise when engineers assume successful authentication at a previous step and as a result infer that authentication in a current step is no longer needed.

In Figure 4, a few principles clearly dominate all of the categorized security events. In the Have Trust Issues category, three principles dominate the overall event distribution. Trustworthy Authentication, Never Assume Trust, and Access Controls cumulatively cover 52% of the total number of trust-related events. In the Build it Right category, 51% of the events fall into the Effectiveness, Understand Attack Surfaces, and Availability areas. Manage your Assets is dominated by Manage Sensitive Data and Validate Information, for a cumulative total of 60% of all related events. Finally, with respect to the remaining events, 51% of them stem from not considering users appropriately, and another 31% of them from weakest link issues within a possible group of attack paths.

During the validation process, we uncovered a potential new principle, Compromise Detection. While closely related to Compromise Recording, there were several findings in which the recording

of an event was insufficient due to inadequate processes to uncover the event in the first place. Detection would include both observing an event and providing an alert of its occurrence. We found twelve events that related to this principle.

It is likely that the utilization of events found in red team engagements skewed the validation results somewhat as well. For example, several of the principles deal with non-functional security measures (e.g. Appropriate Resources, or Timeliness). A red team approaches an assignment with the intent of finding functional security deficiencies, and does not usually uncover non-functional defects unless they are discovered as a side effect. Cyber-security vulnerabilities that are not functional, but rather related to attributes of protected resources or meta-attributes regarding security posture, are not likely to be noted in most cases.

Other Principle Taxonomies

This taxonomy is a way to organize the past few decades of work defining principles for secure cyber systems. It is not a detailed taxonomy of attacks, or security controls, or similar detailed work. Our intention in assembling this was to organize these principles in a memorable way to help other engineers to remember and apply them to systems they are building. Other organizations and authors have built extensive taxonomies at lower levels of detail. The National Institute of Standards and Technology (NIST) has assembled an extensive taxonomy of potential security controls for cyber-systems [2]. This taxonomy is immense, very detailed, and resonably flexible. As it comes from NIST, it is also essentially a compliance requirement for federal civilian information systems. Focused on security controls, the NIST taxonomy in Special Publication 800-53 does not apply well to principle organization.

In 2002, Chakrabarti and Manimaran assembled a taxonomy of internet infrastructure threats. This taxonomy organizes typical internet infrastructure attacks, including categories like *Denial of Service* and *DNS Hacking*. While an excellent way to organize attacks known at the time, this particular taxonomy is very specific to the internet and, again, does not address specific overarching cyber-security prinicples.

Simmons et al. established a more recent taxonomy of cyber attacks in 2014. While more relevant that Chakrabarti's work, primarily because of it's more recent publication, it again categorizes typical known cyber attacks. While complete and well focused, it does not address fundamental cyber-security principles, our focus with this taxonomy.

In fact, developed taxonomies today address attacks [4], risks [6], or known flaws [4]. This taxonomy addresses first principles that need consideration in systems, and does not specifically address technologies. This provides a more flexible way to guide systems security thinking, and one that is hopefully more flexible and applies in a variety of domains.

One other well known taxonomy for cyber-security controls exists; the common triple of Confidentiality, Integrity, and Availability, sometimes paired with Non-repudiation and Authentication. This simple group of principles is well suited to match against this taxonomy precisely because of

its high level of abstraction.

Matching these security goals against our grouped principles, we see that confidentiality and integrity are well represented. Many of the principles specifically address the need to maintain the secrecy of sensitive information, and to ensure that data is validated prior to use. We also have specific principles addressing authentication and authorization. Non-repudiation and availability are less well represented, however.

Non-repudiation principles should be included in the *Have Trust Issues* category. A simple principle of non-repudiation, *Non-repudiation*, would be sufficient. Availability could be included in *Build it Right*, under a sub-category entitled *Keep it Running*, with two contained principles, *Availability* and *Appropriate Resources*. The first principle, *Availability*, addresses the need to keep critical functions of systems available under a hostile conditions, whether those conditions are manufactured by attackers or other, more benign situations. This would address scalability under heavy intermittent load, for example, as this is needed to handle spikes in use as well as denial of service attacks. Just as availability is needed to keep systems running, not over-allocating resources is also important. Systems need to be designed to handle expected loads and attacks commensurate with the importance of their services. Providing unneeded redundancy introduces more complexity and cost with no real advantages.

Conclusion

In this paper, we covered the primary sources of cyber-security architectural principles over the past 40 years and described their contributions. We then organized those principles into a taxonomy, building a structure around the contributed principles to help facilitate application. Once the taxonomy was in place, we compared it to other proposed taxonomies from literature over the past 12 years, demonstrating how our proposed taxonomy differed from others. Finally, we compared our taxonomy to the widely used confidentiality, integrity, availability triad, and showed specific areas where principles were lacking.

With this taxonomy in place, we will begin to pivot toward providing design advice to system architects via defined principles. We expect to provide a way for system architects and designers to identify key system principles and from those principles navigate to specific design patterns they can use to implement software embodying those principles in developed systems. Our overall goal is to help systems engineers create software systems that are more secure by design rather than implementation, and allowing for easy principle-to-design navigation hopefully makes the development of these kinds of systems easier.

References

[1] Privacy and security in computer systems. Technical report, RAND, 1 1975.

[2] Security and privacy controls for federal information systems and organizations. Technical report, NIST, 4 2013.

[3] Category:principle, 2014.

[4] Making security measurible, 2014.

[5] Marco Carvalho, Jared DeMott, Richard Ford, and David A Wheeler. Heartbleed 101. *Security & Privacy, IEEE*, 12(4):63–67, 2014.

[6] James L Cebula and Lisa R Young. A taxonomy of operational cyber security risks. Technical report, DTIC Document, 2010.

[7] Dorothy E Denning and Peter J Denning. Data security. *ACM Computing Surveys (CSUR)*, 11(3):227–249, 1979.

[8] Micheal Gegick and Sean Barnum. Design prinicples, 5 2005.

[9] IEEE. Avoiding the top 10 software security design flaws. Ieee, Institute of Electrical and Electronics Engineers, Piscataway, New Jersey, USA, 2014.

[10] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 447–462, May 2010.

[11] Paul Krebs. The target breach, by the numbers, 2014.

[12] Charles P Pfleeger and Shari Lawrence Pfleeger. *Security in computing*. Prentice Hall Professional Technical Reference, 4 edition, 2006.

[13] Jerome H Saltzer and M. Frans Kaashoek. *Principles of computer system design*. Morgan Kaufmann, 2009.

[14] Jerome H Saltzer and Michael D Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, 1975.

[15] Ruben Santamarta. A wake-up call for satcom security. 2014.

[16] Bruce Schneier, Mudge, and David Wagner. Cryptanalysis of microsoft's pptp authentication extensions (ms-chapv2), 10 1999.

[17] Richard E Smith. A contemporary look at saltzer and schroeder's 1975 design principles. *Security & Privacy, IEEE*, 10(6):20–25, 2012.

[18] Richard E Smith et al. *Elementary information security*. Jones & Bartlett Publishers, 2011.

This page intentionally left blank.


Sandia National Laboratories