A Contemporary Look at Saltzer and Schroeder's 1975 Design Principles

Richard E. Smith | Cryptosmith

"The Protection of Information in Computer Systems" outlined a series of design principles for secure systems in 1975. Some of these principles have become staples of information security practice; others have failed to thrive.

The information security community has a rich legacy of wisdom drawn from earlier work and sharp observations. Not everyone is old enough or fortunate enough to have encountered this wisdom firsthand while working on groundbreaking developments. Many of us receive it from colleagues or through papers and textbooks.

The Multics time-sharing system (see Figure 1) was an early multiuser system that put significant effort into ensuring security. In 1974, Jerome Saltzer produced an overview of Multics security mechanisms, including a list of five "design principles" he saw reflected therein. The following year, Saltzer and Michael Schroeder expanded the article into a tutorial titled "The Protection of Information in Computer Systems."[1] The first section of the tutorial introduced the basic principles of information protection, including the triad of confidentiality, integrity, and availability along with a set of design principles.

In the following decades, these principles were used occasionally as guidelines for developing secure systems. Most of the principles found their way into the US Department of Defense (DoD)'s standard for computer security, the *Trusted Computer System Evaluation Criteria*, also called the "Orange Book." Saltzer and Schroeder's design principles were also highlighted in security textbooks, such as Charles P. Pfleeger's *Security in Computing*,[2] now in its fifth edition.

Different writers use the term *principle* differently. Some apply the term to a set of precisely worded statements, like Saltzer and Schroeder's 1975 list. Others apply it to a collection of unidentified but fundamental concepts. I focus on explicit statements of principles, like the 1975 list, which are concise and well stated on the whole.

In 2008, after teaching a few semesters of introductory information security at the University of St. Thomas, I started writing my own textbook for the course, which was designed to cover the topics required by selected government and community curriculum standards. Informed by an awareness of Saltzer and Schroeder's design principles but motivated primarily by the curriculum requirements, I produced a list of basic principles, which I recorded in the textbook *Elementary Information Security*.[3] In this article, I examine the mismatch between my list and the classic list and look at other efforts to codify general principles, including a recent textbook coauthored by Saltzer himself.[4]

Saltzer and Schroeder's List

Saltzer and Schroeder's 1975 paper listed eight design principles for computer security:

- *Economy of mechanism*. A simpler design is easier to test and validate.
- *Fail-safe defaults*. Outsiders can't enter a store via an emergency exit, and insiders may use it only in emergencies (see Figure 2). In computing systems, the safe default is generally "no access"; the system must specifically grant access to resources. Although most file access permissions work this way, Windows also provides a "deny" right. Windows access control list settings may be inherited, and the deny right lets users easily revoke a right granted through inheritance. However, default deny is easier to understand and implement because interpreting a mixture of permit and deny rights can be complicated.
- *Complete mediation*. Access rights are completely validated every time an access occurs. Systems should rely as little as possible on access decisions retrieved from a cache. Again, file permissions tend to reflect this model: the operating system checks the user requesting access against the file's access permissions. But what happens if the permissions change after a file is opened? Ignoring these changes might be efficient, but it poses major risks, especially with distributed, long-running processes.
- *Open design*. This principle reflects recommendations by the 19th-century cryptographic writer Auguste Kerckhoffs, as well as Claude Shannon's 1948 maxim: "The enemy knows the system." Even the US National Security Agency, which resisted open crypto designs for decades, now uses the Advanced Encryption Standard (AES) to encrypt classified information. Unlike the previous encryption standard the US government adopted, AES was refined using an open process that published the results of the tests and analyses that led to its selection.
- *Separation of privilege*. A protection mechanism is more flexible if it requires two separate keys to unlock it, allowing for two-person control and similar techniques to prevent unilateral action by a subverted individual. Classic examples include dual keys for safety deposit boxes and the two-person control applied to nuclear weapons and top-secret crypto materials. Figure 3 shows how two separate padlocks were used to secure the launch codes for a Titan nuclear missile.
- *Least privilege*. Every program and user should operate while invoking as few privileges as possible. This is the rationale behind Unix sudo and Windows User Account Control, both of which allow users to temporarily apply administrative rights to perform a privileged task instead of being continuously enabled.
- *Least common mechanism*. Users shouldn't share system mechanisms except when absolutely necessary, because shared mechanisms might provide unintended communication paths or means of interference. A classic example arises when two weakly shielded audio cables are bundled together: a loud signal on one might leak into the other, creating noise or even broadcasting the other cable's signal. Similar covert channels might arise when separate processes execute with shared CPU, RAM, or I/O resources.
- *Psychological acceptability*. This principle essentially requires the policy interface to reflect the user's mental model of protection, and notes that users won't use protections correctly if the mechanics don't make sense to them. Passwords fail this principle: users have been harangued for decades about picking hard-to-guess passwords, not sharing them, not using the same one in multiple places, and so on. User surveys and security breaches repeatedly illustrate that many people don't follow those recommendations.

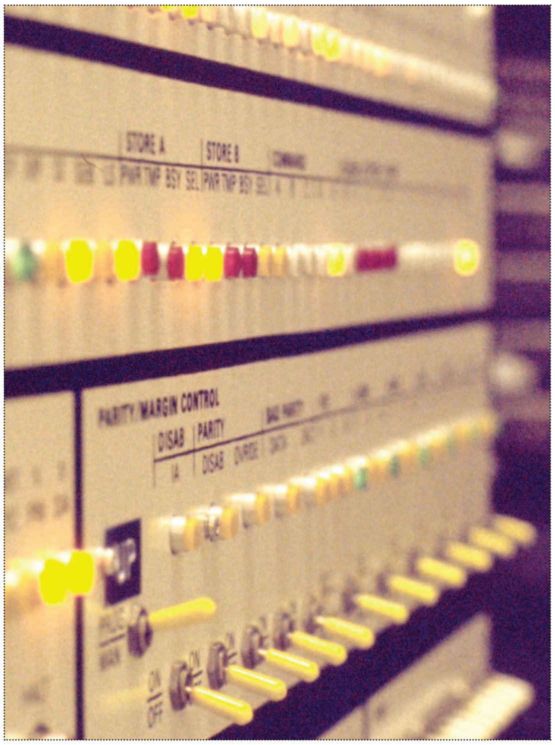

Figure 1. Multics time-sharing system. Jerome Saltzer produced an initial set of design principles to illustrate Multics security features. (Photo courtesy of Tom Van Vleck.)

Saltzer and Schroeder noted two additional principles as being familiar in physical security but applying "imperfectly" to computer systems:

- *Work factor*. Stronger security measures make attackers work harder. In some cases, we can quantify the security in terms of trial-and-error attempts that attackers must perform. Longer passwords and larger encryption

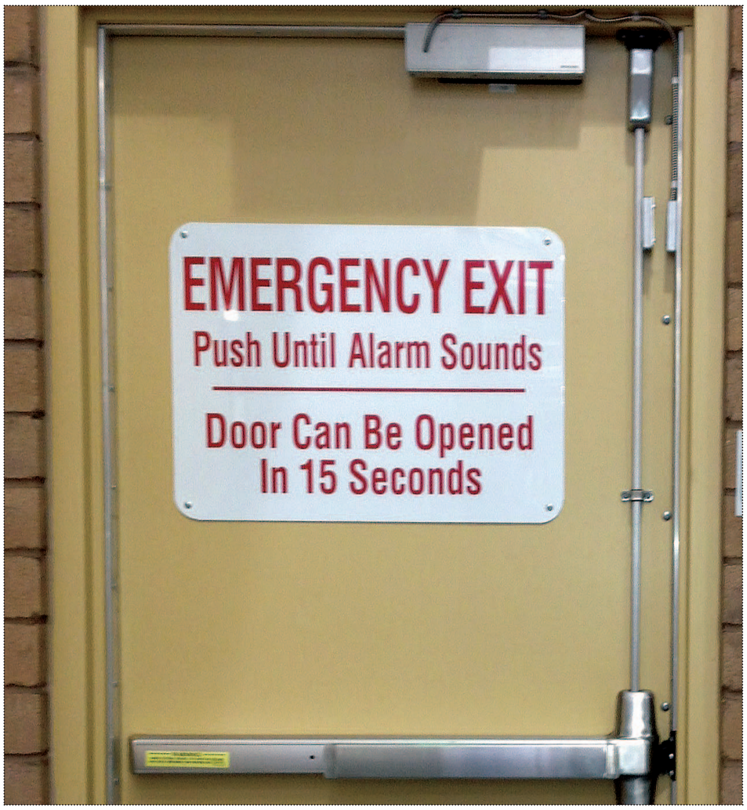

Figure 2. Principle of fail-safe defaults. This retail store keeps its back door closed and locked to prevent thefts. A "panic lock" allows the door to open during an emergency.

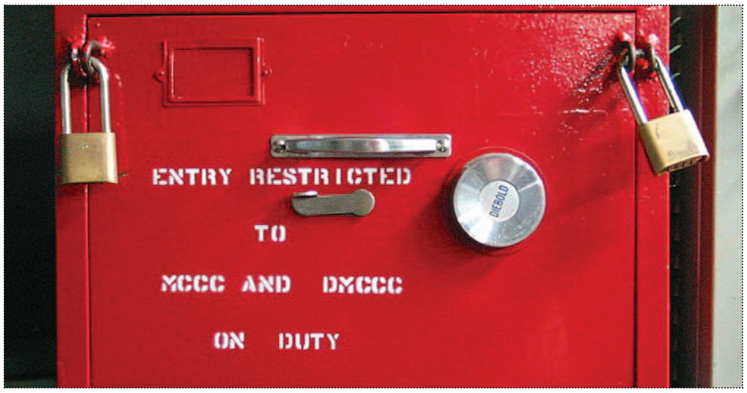

Figure 3. Principle of separation of privilege. Launch codes for Titan II nuclear missiles were stored in this safe. The two-person launch team had to work together to unlock the codes and launch the missile; neither could launch the missile independently. (Photo by the author, courtesy of the Titan Missile Museum.)

keys increase the attackers' work factor by increasing the number of trial-and-error attempts required to guess them. However, in the 1970s, attackers often penetrated systems by finding and exploiting vulnerabilities that didn't seem to rely on systematic trial-and-error attempts. This led Saltzer and Schroeder to question the work factor concept's relevance to computer security.

- *Compromise recording*. The system should keep attack records even if the attacks aren't blocked. Saltzer and Schroeder were skeptical about the benefit of such recordings. If the system couldn't prevent an attack that modified data, then the compromise recording itself might be modified or destroyed.

Today, most analysts and developers embrace these final two principles. Work factor remains the driving force behind the call for more complex passwords and larger encryption keys. The security community has also gained enough experience with other classes of vulnerabilities to apply work factor to a broader range of techniques. Compromise recording has also become an essential feature of every secure system in the form of event logging and auditing.

Security Principles Today

Security principles arise in several contexts. Numerous bloggers and other online information sources produce lists of principles. Many are variants of Saltzer and Schroeder's, including the list provided in the Open Web Application Security Project. In 1991, the US National Research Council released the study *Computers at Risk*,[5] which recommended drafting a standard set of security principles. The US National Institute of Standards and Technology and the International Organization for Standards both took up the challenge. These efforts yielded high-level lists of principles that have been largely abandoned.

Principles also arise in information security textbooks, more often abstract than concrete. Many authors avoid making lists of principles, which is clear from a review of 12 information security textbooks published during the past 10 years. I selected and reviewed these 12 books while searching for appropriate textbooks for an introductory information security course.

Even textbooks that include the word "principles" in the title tend to use the term in the abstract; few systematically identify specific security principles. Almost all textbooks recognize the least privilege principle and usually categorize it as such. Other design principles might use different terms; for example, some sources characterize separation of privilege as a control rather than a principle.

Charles P. Pfleeger and Shari Lawrence Pfleeger presented a set of four security principles—easiest penetration, weakest link, adequate protection, and effectiveness[6]—that encompasses a broader level of security thinking than Saltzer and Schroeder's design principles. Their text also reviews Saltzer and Schroeder's principles in detail. The remaining few textbooks that specifically discuss design principles generally focus on the 1975 list, including *The Craft of System Security* by Sean Smith and John Marchesini[7] and the textbook *Introduction to Computer Security* by Matt Bishop.[8]

Curriculum standards, notably those the US government uses to certify academic programs in information security, cite principles but don't list them. The US government's Information Assurance Courseware Evaluation (IACE) program validates postsecondary education programs against a set of standards maintained by the Committee for National Security Systems. All six IACE curriculum standards refer to principles in an abstract sense. A few of Saltzer and Schroeder's design principles appear piecemeal as concepts and mechanisms—notably least privilege, separation of privilege (called "segregation of duties" by the National Security Telecommunications and Information Systems Security Committee), and compromise recording.

A nongovernment curriculum standard of interest is the *Information Technology 2008 Curriculum Guidelines*, published jointly by the IEEE Computer Society and ACM, which recommends information assurance and security education.[9] It identifies design principles as an important topic, providing a single example—defense in depth.

Saltzer and Kaashoek

In 2009, Saltzer and Frans Kaashoek published *Principles of Computer System Design*.[4] This textbook isn't specifically on information security, but it does revisit many of the 1975 principles. The book lists 16 general design principles and several topic-specific principles, including six security-specific principles. Open design (a general principle) and complete mediation, fail-safe defaults, least privilege, economy of mechanism, and minimize common mechanism (security principles) were essentially inherited from the 1975 paper.

The following are new—or newly stated—principles compared to those described in 1975 (the first is a security principle and the last three are general principles):

- *Minimize secrets*. This thoughtful addition to the list could be prone to misunderstanding. Secrets should be few and changeable, but they should also maximize entropy, and thus increase an attacker's work factor. The opposite situation clearly poses a problem, because each secret increases a system's administrative burden. For instance, a fighter jet project designed in the late 1990s required dozens of separately managed crypto keys to comply with data separation requirements that had been added piecemeal. The result was a management nightmare.
- *Adopt sweeping simplifications*. This restatement acknowledges how hopelessly complex modern systems have become. In the 1970s, a Unix operating system could support a dozen separate users with a megabyte of RAM; a single user on a modern desktop easily consumes a gigabyte of RAM, much of it containing application programs.
- *Least astonishment*. A concise and much clearer restatement of the 1975 list's psychological acceptability principle, least astonishment aims to make security mechanisms comprehensible and fit efficiently into users' activities. This restatement inverts the problem: instead of promoting a hard-to-describe benefit, it focuses on a related, easy-to-describe problem. Although passwords might not be psychologically acceptable, most users are now familiar enough with their quirks. Passwords might be bothersome, but most users no longer find their behavior astonishing.
- *Design for iteration*. This is an important first step toward incorporating continuous improvement as a design principle.

Neither of the uncertain principles listed in 1975 made it into this revised list. But again, event logging and auditing remain a vital part of modern computer security, and work factor calculations continue to play a role in information security system design. Pfleeger and Pfleeger highlighted the weakest link and easiest penetration principles, which reflect the work factor concept.

Unfortunately, the work factor calculations for different vulnerability types often defy direct comparison. For example, an attempt to guess an encryption key requires a very different effort from a search for a vulnerable server through network scanning, even though both perform trial-and-error searches.

Elementary Information Security

In *Elementary Information Security*, I present a set of eight principles, using them as a pedagogic tool to highlight a few important concepts. Some of these principles directly reflect those in Saltzer and Schroeder's work; others reflect more recent terminology and concepts. Stating basic principles as brief phrases seemed like a natural choice for introducing students to a new field of study.

The textbook's contents were primarily influenced by two curriculum standards. The first was the *National Training Standard for Information System Security Professionals*.[10] Although this document clearly shows its age, it remains the ruling standard under the IACE program for training information security professionals. In February 2012, the IACE program certified the textbook as covering all topics required by that 1994 training standard. The second curriculum standard is the ACM/IEEE IT 2008 curriculum guidelines. The textbook covers all topics and core learning outcomes recommended in the guidelines' Information Assurance and Security section.

Eight Principles

To fulfill its instructional role, each principle had to meet certain requirements. Each needed to

- form a memorable phrase related to its meaning, with preference given to existing, widely accepted phrases in the computer security community;
- reflect the current state of the practice, not be simply a "nice to have" property; and
- be important enough to appear repeatedly as new materials were covered.

I introduced each principle when it played a significant role in a new topic, and no sooner. Students weren't required to learn a set of principles that they didn't yet understand or need.

This yielded the following eight principles:

- *Continuous improvement*. Continuously assess how well we achieve our objectives and make changes to improve our results. Modern standards for information security management systems, such as ISO 27001, are based on continuous improvement cycles. Such a process also implicitly incorporates compromise recording from the 1975 paper and design for iteration from the 2009 textbook. The textbook presents this principle with a simple six-step security process to use for examples and exercises.
- *Least privilege*. Provide people and other entities with the minimum number of privileges necessary to allow them to perform their role in the system. This repeats one of the 1975 principles.
- *Defense in depth*. Build a system with independent security layers so that an attacker must defeat multiple security measures for the attack to succeed. This echoes the least common mechanism but seeks to address a separate problem.
- *Open design*. Build a security mechanism whose design doesn't need to be secret. This repeats the 1975 principle.
- *Chain of control*. Ensure that trustworthy software is being executed or that the software's behavior is restricted to enforce the intended security policy. This is an analogy of the chain-of-custody concept in which evidence must always be held by a trustworthy party or be physically secured. A malware infection succeeds if it can redirect the CPU to execute its code with enough privileges to embed itself in the computer and spread.
- *Deny by default*. Grant no accesses except those specifically established in security rules. This is a more specific variant of Saltzer and Schroeder's fail-safe defaults that focuses on access control.
- *Transitive trust*. If A trusts B and B trusts C, then A also trusts C. In a sense, this is an inverted statement of the least common mechanism, but it states the problem in a simpler way. Moreover, this term is already widely used in computer security.
- *Separation of duty*. Decompose a critical task into separate elements performed by separate individuals or entities.

In retrospect, the list is missing at least one pithy and well-known maxim: "Trust, but verify." The book discusses the maxim, but it isn't tagged as a basic principle. Two additional Saltzer and Kaashoek principles might also belong in this list. The goal to minimize secrets could be a basic information security principle, either by itself or in some combination with open design. In addition, including the principle of least astonishment would have made a positive statement in favor of usable security mechanisms.

Omitted Principles

For better or worse, three of the 1975 principles don't play a central role in modern information security practice: simplicity, complete mediation, and psychological acceptability.

There's no real market for simplicity in modern computing. A private company releases product improvements to entice new buyers, and the sales bring in revenues to keep the company operating. The company remains financially successful as long as the cycle continues. However, each improvement increases the underlying system's complexity. Much of the free software community is caught in a similar cycle of continual enhancement and release. Saltzer and Kaashoek call for sweeping simplifications instead of overall simplicity, reflecting this change.[4]

Complete mediation likewise reflects a sensible but obsolete view of security decision making. Access control decisions in modern network systems are often distributed among separate firewalls, spam filters, and antivirus products. The site policy might call for blocking malicious email content, but no single device contains enough information to reliably detect and discard an embedded virus.

Psychological acceptability is an excellent goal, but it's honored more in breach than observance. As I noted, passwords can't seriously be considered psychologically acceptable. At most, users are resigned to them. Fortunately, users are rarely astonished by password behavior anymore.

Saltzer and Kaashoek's restatement of acceptability as the least astonishment principle might grudgingly find passwords acceptable, but it still raises the bar on other security mechanisms, notably access control mechanisms. The current generation of graphical file access control interfaces provides no more than rudimentary control over low-level access flags. A sophisticated knowledge of existing permissions is necessary to understand how a change in access settings might affect a particular user's access rights.

Observations

Least privilege, open design, fail-safe defaults, separation of privilege, and the least common mechanism are five design principles that still thrive. In some cases, authors use the exact phrasing as appeared in the 1975 paper. The least privilege principle typically appears with that phrasing, as does open design. Security environments that focus on secrecy or integrity often focus on access control and rephrase fail-safe defaults as a form of deny by default. Saltzer and Schroeder's original phrasing applies more generally to safety and control systems and to access control. Separation of privilege appears in many phrasings, sometimes morphing into "segregation of duties." Unlike fail-safe defaults, these different phrasings are often defined identically and described with identical examples.

Today, people often practice the least common mechanism by default, not intention. Many information security design decisions through the 1980s were driven by the high cost of individual computer systems. Mutually suspicious users had to share mechanisms because providing separate computing resources was too expensive. As computing systems evolved and prices fell, sharing turned expensive.

This evolution has also led the security community to ignore or abandon the remaining three design principles. A few experts might still point to them as nice features to have, but few widely used systems incorporate them. Economy of mechanism has been left far behind in both the profit-oriented and free software communities. Complete mediation has been nibbled to death by modern network security's distributed nature. Today, no single point of control can completely and accurately decide a network traffic stream's access rights. The "psychological acceptability" of security measures carries more weight today than it once did but is often trumped by cost concerns.

Is there any value in having a concise list of security principles? Security practitioners continue to recite specific Saltzer and Schroeder principles when justifying design decisions, even though the list doesn't encompass every relevant issue. Although the short phrases are easy to remember, they could promote an overly simplistic view of technical problems. Incorporating effective security into modern systems involves many complicated tradeoffs that a simple list can't reflect.

Perhaps such lists of principles should only exist as educational tools. Simplicity helps students build an understanding of a more complex reality. Such lists might serve a worthwhile pedagogical purpose, even if they don't encompass the entire scope of information security practice. As students become practitioners, they won't abandon least privilege, separation of duty, or their relevant siblings simply because the list of principles might never be complete. ■

References

1. J.H. Saltzer and M.D. Schroeder, "The Protection of Information in Computer Systems," *Proc. IEEE*, vol. 63, no. 9, 1975, pp. 1278–1308.
2. C.P. Pfleeger, *Security in Computing*, Prentice Hall, 1998.
3. R. Smith, *Elementary Information Security*, Jones and Bartlett, 2013.
4. J.H. Saltzer and M.F. Kaashoek, *Principles of Computer System Design*, Wiley, 2009.
5. "Computers at Risk: Safe Computing in the Information Age," Nat'l Research Council, National Academy Press, 1991; www.nap.edu/openbook.php?record_id=1581.
6. C.P. Pfleeger and S.L. Pfleeger, *Security in Computing*, 4th ed., Wiley, 2006.
7. S. Smith and J. Marchesini, *The Craft of System Security*, Addison-Wesley, 2008.
8. M. Bishop, *Introduction to Computer Security*, Addison-Wesley, 2005.
9. *Information Technology 2008: Curriculum Guidelines for Undergraduate Degree Programs in Information Technology*, IEEE CS and ACM, Nov. 2008; www.acm.org//education/curricula/IT2008%20Curriculum.pdf.
10. *National Training Standard for Information Security (INFOSEC) Professionals*, report NSTISSI 4011, Nat'l Security Telecommunications and Information Systems Security Committee, 1994; www.cnss.gov/Assets/pdf/nstissi_4011.pdf.

Richard E. Smith is a writer, educator, and principal of Cryptosmith, a consultancy. He has a PhD in computer science from the University of Minnesota. Smith authored *Elementary Information Security* and has developed a distance learning program based on the textbook. Contact him at rick@cryptosmith.com.

In my view, a defender who doesn't know how to attack is no defender at all.
—Earl Boebert