



## There are no new vulnerabilities

Position paper for

ACSA Workshop on the Application of Engineering  
Principles to System Security Design (WAEPPSSD)

### Author

Gary Stoneburner  
National Institute of Standards and Technology (NIST)  
100 Bureau Drive, Stop 8930  
Gaithersburg, MD 20899-8930  
301-975-5394, Stoneburner@nist.gov

### ABSTRACT

The practitioners of computing security get it wrong when the phrase “a new vulnerability” is used. THE vulnerability is the same and has not changed in decades. THE vulnerability is use of poor quality systems. It is like a guy proclaiming that he has found a hole in his bowl and the best response to this proclamation is - “Well duh, your holding a sieve not a bowl!” The need is to recognize this, for the near-term to use IT more wisely, and for the longer-term to move the COTS market place toward higher-quality by the expectation of the application of engineering disciplines.



## There are no new vulnerabilities

Position: The practitioners of computing security get it wrong when the phrase “a new vulnerability” is used. THE vulnerability is the same and has not changed in decades. THE vulnerability is use of poor quality systems. It is like a guy proclaiming that he has found a hole in his bowl and the best response to this proclamation is - “Well duh, your holding a sieve not a bowl!”

Confusing “another symptom of the problem” with the problem itself is the computer analogy of treating the signs of a disease instead of the disease. Instead of addressing the problem the community continues to address results of the problem, and hence is in a no-win situation where the attacker has (and is likely to retain) the advantage. We expend a lot of resources, tell management that this process, mechanism, or whatever will fix it, and still get hacked. My recent experience on active duty with the US Army Information Operations red team has convinced me that if an Army unit (or garrison) is attacked by a world-class adversary - the Army loses - period. The use of commercial off the shelf (COTS) components for critical systems (and if you care about it, it is critical ☺) is at the core of the problem.

First recognize that more of what we have been doing will not help.

- Best Commercial Practice. There is an implicit, if not explicit, confusion of “best” with “good”. The common, best commercial practice results in complex systems with 10’s of thousands of flaws. This is based upon number of lines of code and common metrics for errors per line of code - see the following references:

[http://www.garynorth.com/y2k/detail\\_.cfm/640](http://www.garynorth.com/y2k/detail_.cfm/640)

<http://world.std.com/~tej/sweng.html>

<http://ftp.stsc.hill.af.mil/crosstalk/1996/jul/quantify.asp>

<http://www.bell-labs.com/news/2002/march/holzmann.html>

<http://www.embedded.com/story/OEG20020104S0084>

and many more ...

There is no feasible means of uncovering and patching all of the security-critical flaws in complex systems built to current best commercial practice. The ONLY measure of success that counts is whether the adversary is impacted by the patching done. It is not how many flaws were patched that is important, but how many remain! Patching 1000 or 2000 flaws might make it appreciably harder to find a flaw by uncovering the correcting the easy to find, leaving only the hard to find ones. Patching 1000 of 10,000 flaws is likely to be of much less benefit in terms of making it harder for the adversary to succeed. These two examples are the same number of patches, but radically different end states, 50% less versus 10% less flaws.



- Adversary knows flaws we don't. This obvious fact is enabled by the item above and is commonly ignored. Even if we were able (we aren't!) to keep our systems protected against the publicly known flaws, security practitioners seem to fail to stop and consider how flaws become public knowledge. Individuals and small groups without state sponsorship are routinely uncovering problems. We also publicly discover a flaw when a poorly crafted exploit gives itself away. Only a fool believes that this constitutes the sum total of known flaws. We know about the flaw only because the discoverer chose to tell us, or the attacker was incompetent or clumsy. If an individual can uncover a flaw in Windows or Internet Explorer or Solaris or ..., then certainly a state-sponsored or large organization sponsored can develop (or just hire) the expertise and laboratory needed to find previously unknown flaws in our operating systems, firewalls, routers, guards, and other components. We would be sadly mistaken if we think that all attackers are as bumbling as those we know about.
- Loss recovery model. COTS components are developed for and used in a mass marketplace that is implicitly based upon the loss recovery model of blame shifting. As long as I can successfully shift the blame elsewhere (stick it on someone else or get someone else to pay; e.g. insurance) the level of security quality is much less important than the level of base functionality provided and the cost to purchase and operate this base functionality. This loss recovery model is certainly NOT acceptable for a warfighter nor for critical infrastructure. Yet by using COTS in such systems we are implementing IT that is good-enough for one environment (benefit of doubt ☺), and not considering how radically different the needs of our environment are.
- Lack of engineering discipline. The development of complex IT within the COTS world is, by evidence the design documentation used and the resulting products delivered, done without adherence to well-known, been taught for decades, engineering principles. Microsoft for example, recently came to the conclusion that security is a problem. But Microsoft did NOT come to the conclusion that their design processes were at fault. No, just the opposite. The public presentation by a Microsoft representative at the recent NSA red-team conference included the information that Microsoft builds good stuff, just that they need to consider security more. This consideration was, according to this presenter from Microsoft, accomplished by a one-day class to developers followed by a two-month code scrub. They still do not get it! The need is for a cultural change resulting new design processes and procedures.

Second take steps to address the real problem

- How we use the system. For the near-term: the security quality of the system we have (the “hand we have been dealt”) is a given (see above). The primary risk mitigation tool available to us is decisions on how we integrate the IT into our mission or business process. Unfortunately, at this point in time the base



functionality of the system determines how the system is used. Mission impacts made possible because of this use are ignored, explained away as not going to happen, or whitewashed with a technical mechanism, policy, or procedure. The risk remains, the organization has no means to effectively protect against it, and the organization will not train it a realistic attack environment. To do such training would destroy the capability of achieving any other training objectives. The decision-maker sees the security concern as the problem, not the adversary security is concerned about.

- Apply engineering disciplines. Over the longer-term the goal is to move the COTS marketplace toward quality IT. The first step is the application of basic, known for decades engineering disciplines. For software this includes, among other items, modularity - not just having modules, but modules that are crafted to be largely independent. Key choices about where to draw module boundaries impact all subsequent attempts at a security-quality design.

Next is the definition and application of engineering disciplines directly related to the security quality of the system. Two NIST publications that provide a start in this direction are:

SP 800-33 Underlying Technical Models for Information Technology Security,  
<http://csrc.nist.gov/publications/nistpubs/800-33/sp800-33.pdf>

SP 800-27 Engineering Principles for Information Technology Security (A Baseline for Achieving Security),  
<http://csrc.nist.gov/publications/nistpubs/800-27/sp800-27.pdf>

In summary, there is the need to see the real problem. Until we do, our efforts will continue to be of little use. Having seen the problem, we need to address IT as it really is, not as we wish it to be. Use IT much more wisely. In parallel, develop security engineering principles, present them as expectations like we have expectations on mechanical engineering principles in the design of a bridge, and take advantage of standards bodies and government organizations like NIST to promote their use.