Mostly sunny with a chance of cyber[1]

David Flater, NIST, 2016-05-09

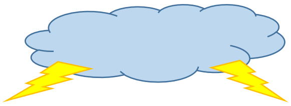
Counting known vulnerabilities and correlating different factors with the vulnerability track records of software products after the fact is obviously feasible. The harder challenge is to produce "evidence to tell how vulnerable a piece of software is" with respect to vulnerabilities and attack vectors that are currently unknown. This means forecasting the severity and the rate at which currently unknown vulnerabilities will be discovered or exploited in the future, given a candidate system and its environment.

Meteorologists can observe the present state of a weather system and assume that the future state must evolve from it through the application of known physics. Small features that are below the resolution of the radar are correspondingly limited in their impact, so the uncertainty can be bounded. But for computer system vulnerabilities, there are no analogous limits. High-impact exploits of tiny, obscure quirks that were not on anyone's "radar" appear with regularity. Although the resolution of that "radar" is continuously improved, the complexity of systems is increasing faster, so the relevant details are inexorably receding into the background.

Under these conditions, our best available predictors of future vulnerabilities in systems that were responsibly designed and implemented may be nothing more than metrics of size, complexity, and transparency. Unexciting as it may be, there is rationality to this approach. To develop a market for smaller, simpler, more verifiable systems would not be too modest a goal for a large government effort to attempt.

[1] Disclaimer: This statement reflects only the views of the author on the topics discussed, and does not necessarily reflect the official position that NIST may have about those topics.

Mostly sunny
with a chance of cyber



David Flater

dflater@nist.gov

2016-07-06

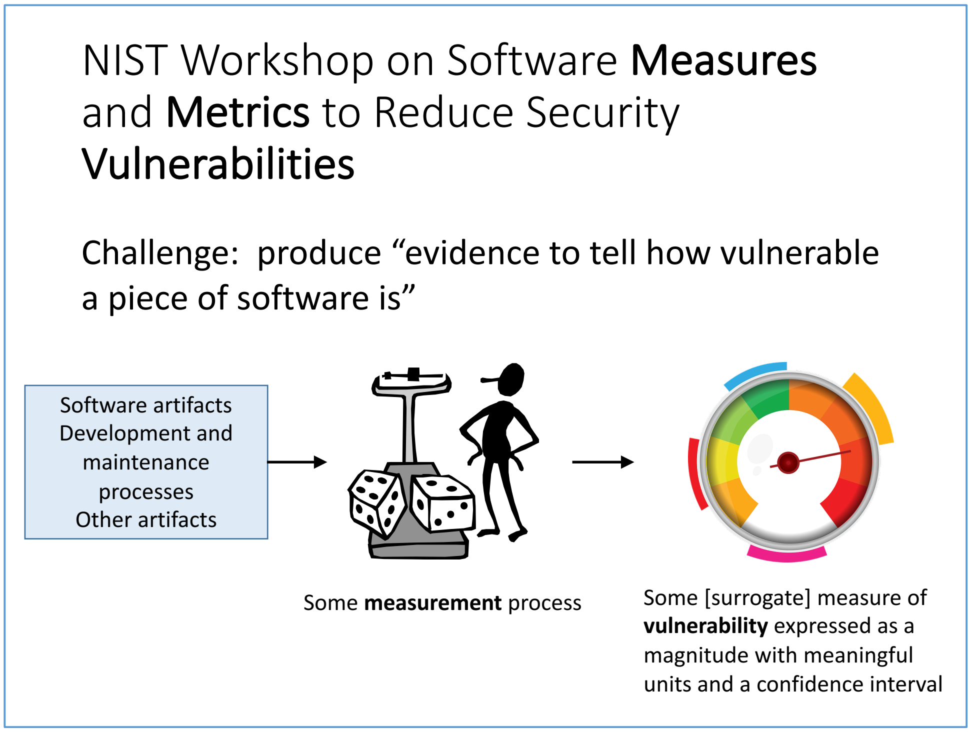
with notes added 2016-07-14

1. This presentation reflects only the views of the author on the topics discussed, and does not necessarily reflect the official position that NIST may have about those topics.
2. Identification of commercial products and entities is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the products or entities are necessarily the best available for the purpose.

I added these notes after the workshop to include important points that don't appear in the text of the slides.

Thesis

- The nature of the challenge is not measurement, but prediction
- Conditions are unfavorable for making a rational prediction
- Measuring what *is* measurable and applying empiricism will move us forward
- Measuring *cost* reveals a complication



NIST Workshop on Software **Measures** and **Metrics** to Reduce Security **Vulnerabilities**

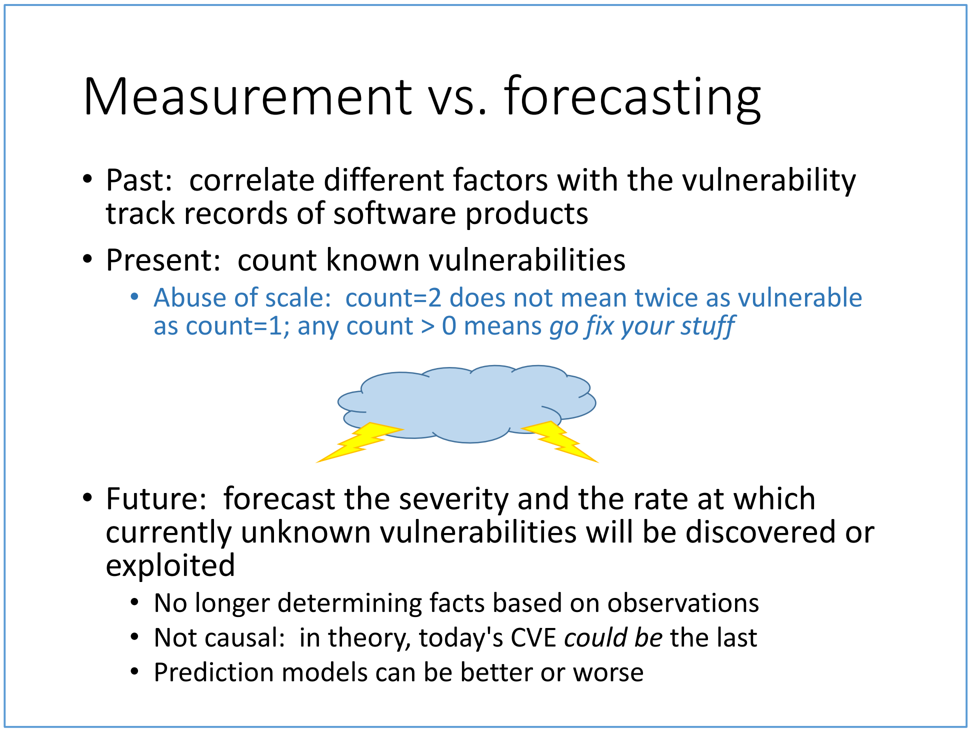
Challenge: produce “evidence to tell how vulnerable a piece of software is”

Software artifacts
Development and maintenance processes
Other artifacts

Some **measurement** process

Some [surrogate] measure of **vulnerability** expressed as a magnitude with meaningful units and a confidence interval

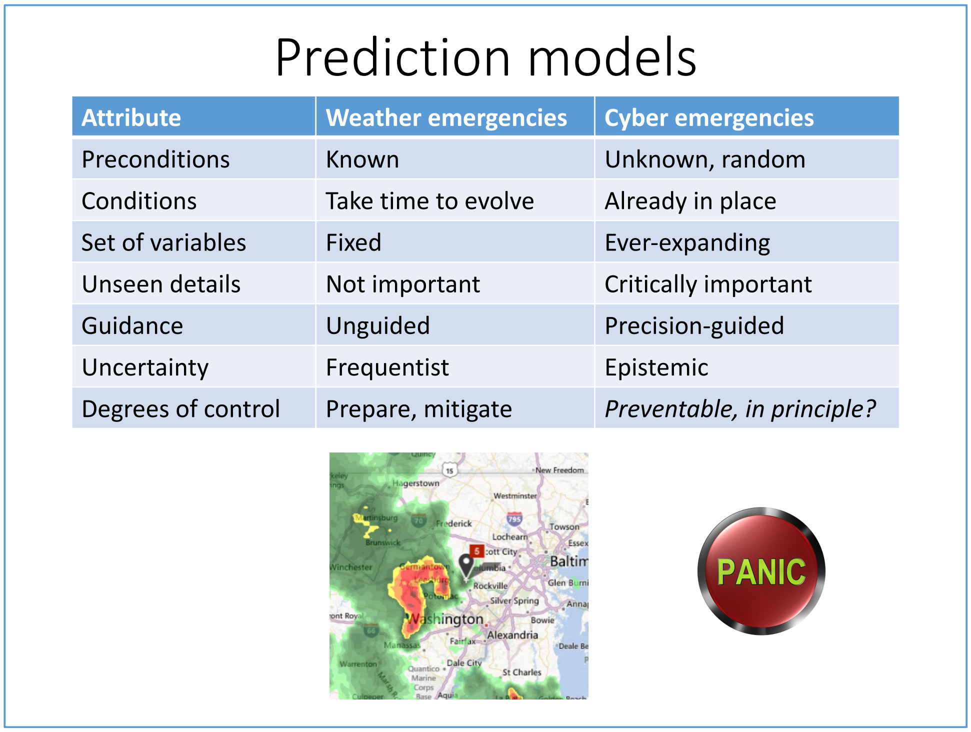
The metrology perspective is that measurement is about quantities. A quantity like 5 kg has meaning because it is defined as 5 times a standard reference, the unit. In most cases it would be nonsense to say that Software A is 5 times as vulnerable as Software B. Vulnerability is a quality, not a quantity. At best we may measure some quantity that helps us to characterize it better.



Measurement vs. forecasting

- Past: correlate different factors with the vulnerability track records of software products
- Present: count known vulnerabilities
  - Abuse of scale: count=2 does not mean twice as vulnerable as count=1; any count > 0 means *go fix your stuff*
- Future: forecast the severity and the rate at which currently unknown vulnerabilities will be discovered or exploited
  - No longer determining facts based on observations
  - Not causal: in theory, today's CVE *could be* the last
  - Prediction models can be better or worse

The count of known vulnerabilities is unsuitable as a surrogate measure of vulnerability. The future question is the most interesting one.

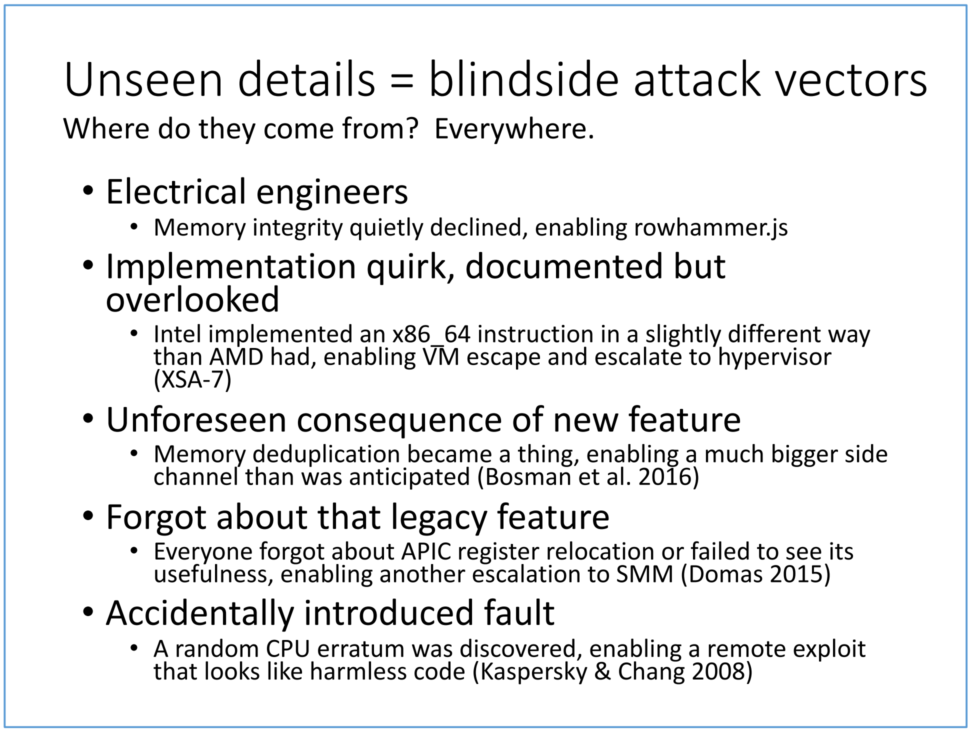


Prediction models

Attribute	Weather emergencies	Cyber emergencies
Preconditions	Known	Unknown, random
Conditions	Take time to evolve	Already in place
Set of variables	Fixed	Ever-expanding
Unseen details	Not important	Critically important
Guidance	Unguided	Precision-guided
Uncertainty	Frequentist	Epistemic
Degrees of control	Prepare, mitigate	*Preventable, in principle?*

In every respect but one (controllability), cyber emergencies are less predictable than weather emergencies. I will focus on the different impact of unseen details.

We can obtain an adequate prediction of impending weather emergencies even though the radar misses many small details. The butterfly effects do not matter as long as we can see the hurricane on its way with ample time to react. But for cyber emergencies it is exactly the opposite; it is the unseen details that are most likely to create an emergency with no warning at all.



Unseen details = blindside attack vectors

Where do they come from? Everywhere.

- Electrical engineers
  - Memory integrity quietly declined, enabling rowhammer.js
- Implementation quirk, documented but overlooked
  - Intel implemented an x86_64 instruction in a slightly different way than AMD had, enabling VM escape and escalate to hypervisor (XSA-7)
- Unforeseen consequence of new feature
  - Memory deduplication became a thing, enabling a much bigger side channel than was anticipated (Bosman et al. 2016)
- Forgot about that legacy feature
  - Everyone forgot about APIC register relocation or failed to see its usefulness, enabling another escalation to SMM (Domas 2015)
- Accidentally introduced fault
  - A random CPU erratum was discovered, enabling a remote exploit that looks like harmless code (Kaspersky & Chang 2008)

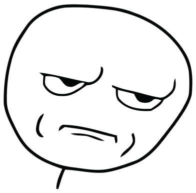
The threat model is of finite size. The unknown universe of potential attacks may be infinitely large. At least it is larger than our imagination, as we are consistently caught by surprise.

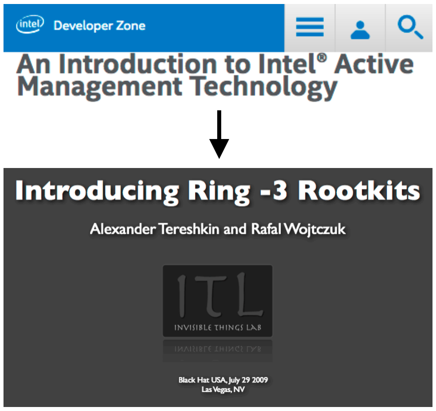
The idea that fully addressing the top 10 or top 25 attack vectors would cause there to be fewer successful attacks is an untested hypothesis. Past experience suggests that there is a large reserve of attack vectors that do not appear in the threat model. Perhaps attackers will simply move farther down the list and never run out of attacks.

Different perspectives, different metrics: the security industry sees progress in increasing the complexity of attacks, but the target sees no progress unless the frequency of attacks actually goes down.

The future will not be mitigated

- An assurance case is a fixed, closed-form expression up against an *evolving, open world*
- The unseen attack surface is vast and growing
- No opt-out





An Introduction to Intel® Active Management Technology

Introducing Ring -3 Rootkits

Alexander Tereshkin and Rafal Wojtczuk

ITL
INVISIBLE THINGS LAB

Black Hat USA, July 29 2009
Las Vegas, NV

Even if you had complete visibility into the system as it stands, there is the problem of future-proofing the assurance case. We are forced to upgrade in order to close the barn door on known vulnerabilities. Each upgrade comes with an expanded attack surface, which leads directly to new vulnerabilities.

Risk models vs. unknown unknowns

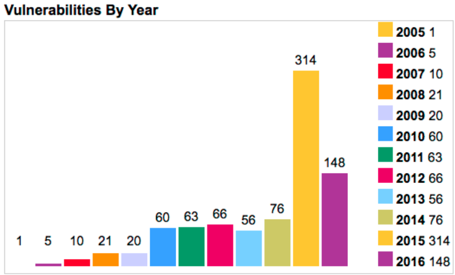
- "Risks"
  - Valid to estimate based on historical data
- "Structural uncertainties"
  - Follow from events that are rare or nonexistent in the historical record
  - Frequentist reasoning breaks down
- *"Unknowables"*
  - *Follow from inconceivable events*
  - *Bayesian reasoning breaks down*

Kees van der Heijden. Scenarios: The Art of Strategic Conversation. John Wiley & Sons, 2nd edition, 2005.

A risk model cannot do justice to unknown unknowns. We cannot possibly estimate the probability of something that, by definition, we know absolutely nothing about. Such a number is nothing but an arbitrarily chosen safety margin.

Growth models

- No evidence that security grows / vulnerability decreases over time (?)
- "Trivial forecast has some predictive accuracy" (Timm Grams, "Reliability Growth Models Criticized")
- Applicable to the frequency of vulnerability discovery

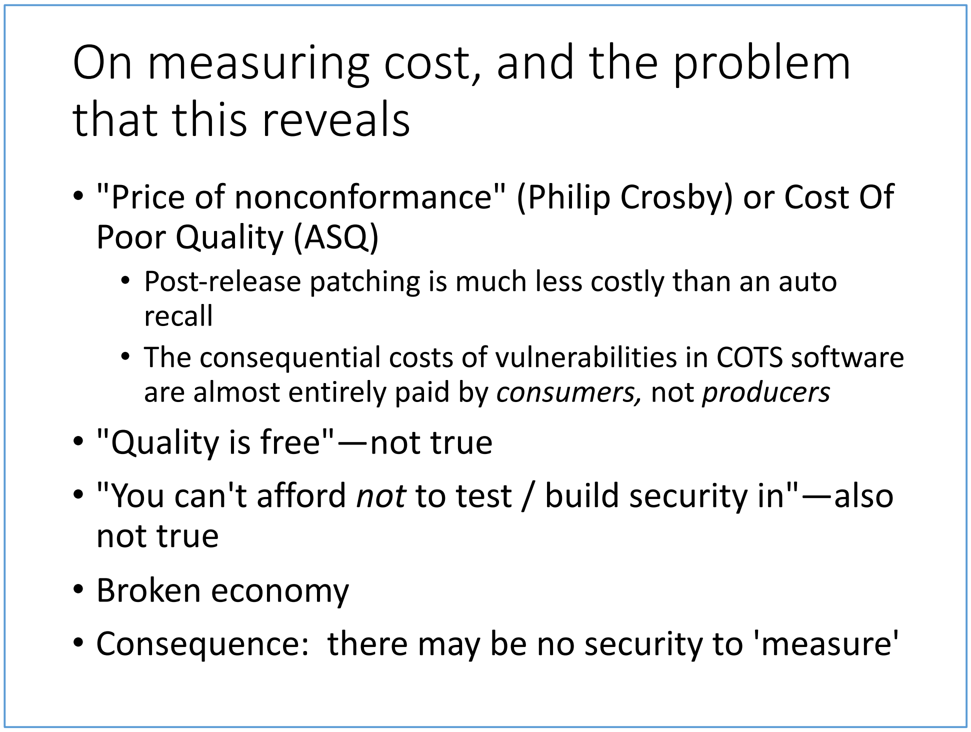


Security may grow over time in tightly-controlled systems, but the more typical treadmill of vulnerabilities and mitigations suggests that it does not grow over time in general. (Taking the target's perspective that the difficulty of exploits is irrelevant if they just keep on happening.)

What is measurable?

- Known quantities
  - Track record of fixed vulnerabilities
  - Known unfixed vulnerabilities
  - Measurable hardness of certain kinds of defenses
- Hypothesized indicators of unknown vulnerabilities
  - Measures of diligence
    - Test/analysis coverage
    - Hardening measures
  - Size & complexity
  - Area of attack surface
  - "Code smells" (operationalized)
  - Transparency (including amenability to analysis of whatever kind)

Inventing a metric is only the beginning. Hypotheses must be tested. Measurements must be validated.



On measuring cost, and the problem that this reveals

- "Price of nonconformance" (Philip Crosby) or Cost Of Poor Quality (ASQ)
  - Post-release patching is much less costly than an auto recall
  - The consequential costs of vulnerabilities in COTS software are almost entirely paid by *consumers,* not *producers*
- "Quality is free"—not true
- "You can't afford *not* to test / build security in"—also not true
- Broken economy
- Consequence: there may be no security to 'measure'

This argument is not valid for products whose primary customer is the government, for regulated industries, or for long-lifecycle software. It applies only to the mass market.

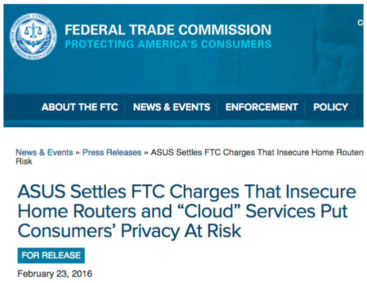
We are familiar with studies showing that the cost of correcting defects is less if they are detected and corrected earlier in the process. But as long as the market tolerates faulty software, the producer's cost can be lowered further by just never correcting the defects. A lot of software is being produced as a consumable (or as part of a consumable) rather than a durable good. Maintenance is minimized, and after a date certain the product is simply abandoned and the next product is rolled out.

Within the mass market, the cost of poor security may even go negative: a more secure product may be too difficult to configure, resulting in a competitive disadvantage. Even if the cost of building security in is reduced to marginal as the strategic plan envisions, the business case may remain broken.

This economic problem may overwhelm and obviate the measurement problem.

Conclusions

- There is value in correlating different factors with the vulnerability track records of software products after the fact
  - Hypothesized indicators
  - Programming languages
  - Development techniques
  - Quality processes
  - Formal methods….
- Engineering wasn't invented; it evolved
- Do what [apparently] works, but verify and track progress
- Goal: reliable predictors, best practices
- However, there also needs to be a business case
- Redistributing risk may be necessary to "significantly curtail software vulnerabilities" in the COTS market



News & Events » Press Releases » ASUS Settles FTC Charges That Insecure Home Routers
Risk

ASUS Settles FTC Charges That Insecure Home Routers and "Cloud" Services Put Consumers' Privacy At Risk

February 23, 2016

Empiricism is a useful strategy when we are overwhelmed by unknowns, but it must be used with great caution. Correlation is not causation. A good fit to past data does not ensure a good prediction. Hypotheses must be tested. Measurements must be validated. Apply science.

"Measure what is measurable,
and stop yer lyin' about the rest"

(Misquoting Galileo)

Software Metrology

David Flater

dflater@nist.gov

Not addressed: we also need software to be sufficiently functional running at least privilege that tricking users into granting excess permissions to trojans will no longer work.