Holistic Approaches to Trustworthiness, Security, & Privacy

- -

Peter G. Neumann
Principal Scientist
SRI International ComputerSciLab
Menlo Park, CA 94025-3493
Neumann@CSL.sri.com
http://www.csl.sri.com/neumann
Tel 1-650-859-2375

Cybersecurity Summit 2008 for NSF Large Research Facilities
7 May 2008

Holistic Approaches

- **Holistic approaches consider systems and enterprises in their entirety, in the context of their environments, lifetimes, and total ranges of actual and anticipated uses.**

Trustworthiness is Holistic 1

- Trustworthiness involves many end-to-end emergent properties, some of which may be critical, such as security, reliability, system survivability, human safety. Also relevant are evolvability, usability, interoperability, ...
- Some emergent properties are undesirable and must be avoided such as flaws/incompatibilities/ covert channels ... that result from system compositions.

Trustworthiness Is Holistic 2

- **Trustworthiness is pervasive. Systems need to satisfy all critical requirements, not just security.**
- **Trustworthiness is highly multidimensional. It is not a local property, especially for applications. Total-system analysis is needed.**
- **Trustworthiness is a weak-link phenomenon. Today everything is a potential weak link.**

Trustworthiness Is Holistic 3

- **Security, reliability, and other critical requirements interact, and can be incompatible.**
- **Effects of flaws and bugs can propagate extensively (e.g., power and network outages, malware).**
- **Application security is easily undermined by poor OS security.**
- **Outages must be anticipated.**

Systems Demand Holistic Analyses

- Energy: long-term future-oriented/ short-term optimization
- Agriculture: natural/industrial
- Health care: prevention/“cure”
- Systems: principled/unprincipled

Holistic analyses are relevant for each, with economic, political, international, social, and other implications.

Interdisciplinary Understanding?

- **What might we learn from these other disciplines with respect to trustworthiness (remembering that reasoning by analogy can be misleading)?**
- **See PGN, Holistic Systems,** *ACM SIGSOFT Software Engineering Notes,* **November 2006, pages 4-5. http://www.csl.sri.com/ neumann/holistic.pdf**

Energy

- Renewable resources (solar, wind, biomass, hybrids) may be made viable if considered holistically.
- Fossil fuels are short-sighted, nonenvironmental, nonrenewable, contribute to global warming.
- Nuclear safety regulations may be strong, but enforcement is weak; waste disposal/recycling problems.

Agriculture

- Sustainable agriculture uses natural fertilizers/pest-controls, crop rotations. It is healthier for workers and consumers.
- Industrial agriculture causes soil depletion, toxic runoffs, worker and consumer health problems, and diminishes diversity.
- Animal agriculture generates more greenhouse gases than vehicles.

Health Care

- Preventive/alternative/traditional methods (orthomolecular, herbal, Oriental, Ayurvedic, exercise, diet, ...) can treat the whole person, and are environmentally aware.
- Conventional medicine seeks quick fixes that suppress symptoms rather than eliminating causes. It may be iatrogenic (exacerbating the disease, triggering bacterial mutations, and so on).

System/Network Development

- Holistic approaches, including principled system development, have many long-term benefits.
- Bad practices include inadequate requirements and architectures, nondecomposable systems, poor software engineering, sloppy software, unsafe languages, iatrogenic patches, ... These create many nasty problems.

Avoiding System Risks

- **Ted Glaser: "A modular system is one that falls apart easily."**
- **Modularity is not enough; we need encapsulation, compatibility, interoperability, noninterference, some sound formal bases, ..., to build constructively trustworthy systems with predictable composability and interoperability.**

Holistic Analysis Is Needed

- Principled development of trustworthy systems must be demonstrably cost-effective before it can become pervasive. How can this be accomplished?
- Consider the roles of principles, available source code, and formal methods, ..., all of which are increasingly applicable.

Principled Systems

- 1965 Glaser-PGN, Basic principles underlying Multics development
- 1969 PGN, Role of Motherhood
- 1975 Saltzer-Schroeder, Protection of Information in Computer Sys.
- 2004 PGN, Principles for predictable composability
- 2008 PJDenning, Great Principles

Saltzer-Schroeder Principles (1975)

- Economy of mechanism
- Fail-safe defaults
- Complete mediation
- Open design
- Separation of privilege
- Least privilege
- Least common mechanism
- Psychological acceptability
- Work factor
- Recording of compromises

Principled System Development

- **Holistic approaches to complexity: sound requirements, principles, structured architectures; good software engineering practice; design for trustworthiness, interoperability, evolvability, usability, administrability, ...; pervasive assurance analysis, formal methods, and lots more.**

Complexity

- **Simplicity is highly praised, but highly trustworthy systems are inherently complex.**
- **Oversimplifying creates problems. "Everything should be made as simple as possible, but no simpler." Albert Einstein**
- **Sound structures, composability, principles, discipline all can help.**

Principled System Design

- **Management of complexity through constructive architectures that modularly localize what must be trustworthy, such as separation kernels, virtualization, alternative approaches to multiple security levels, and so on.**

Principled System Implementation

- Predictably composable designs
- Sound software engineering
- Sound programming languages
- Property-preserving refinements
- Proactive code analysis
- End-to-end self-checking

Principled System Assurance

- **Pervasive assurance throughout development/use cycles.**
- **Assured composability, with with hierarchical closure, e.g., the Computational Logic stack, PSOS/Robinson-Levitt, Rushby-DeLong (new work).**
- **Assured multilevel security?**

Deja Vu All Over Again, Yogi Berra

- **Unfortunately, the same types of mistakes (design flaws, software bugs, operational errors) recur.**
- **There is much to be learned from many past mistakes. Education/training are crucial.**
- **Various examples follow.**

System Development Difficulties

- FAA Air traffic control redux
- IRS system modernization efforts
- FBI Virtual Case File
- Employment Eligibility (EEVS)
- Secure Global Information Grid
- 2010 Census handheld terminals
- CAL Deadbeat Dads Registry
- Customs entry database system
- Election systems (Fed and state)

Backup and Recovery Risks 1: Air-Traffic Control Failures

- **LA Palmdale ATC Jul 2006 power**
- **Reagan National Apr 2000 power**
- **LI NY ATC SW upgrade Jun 1998**
- **LA ElToro ATC 104 failures/day 1989 (no previous system saved)**
- **3 NY airports 1991 (on batteries)**

Backup and Recovery Risks 2

- -

More total system/backup failures:

- **Swedish central train res system**
- **Washington Metro Blue Line 1997**
- **SF BART SW upgrades Apr 2006**
- **Japanese stock exchange Nov 2005**

Cases of losses with no backup:

- **NY Public library references**
- **Dutch criminal mgmt system**

Propagation Risks 1: Widespread Network Outages

- **1980 ARPANET collapse: router memory errors, weak garbage collection of old status messages, memory overflow in every node.**
- **1990 AT&T longlines collapse: untested change in recovery code, repeated crashing for half a day.**

Propagation Risks 2: Widespread power outages

- -

- Northeast US, Nov 1965
- Lower NY State, Jul 1977, >26 hrs
- Ten Western states, Oct 1984
- Western US, Jul 1996, heat/tree
- Western US/Canada/Baja, Aug 1996
- Northeast, Aug 2003, >2 days

Propagation Risks 3: Power outages in 2006

- -

- Queens, NY, week-long, wiring
- Portland, Oregon, October
- Ems River, Germany, November. preventive shutdown failed to consider iterative implications (N-1), affecting 10 million in 6 countries from Austria to Spain.

Software Flaws 1

- **Buffer/stack overflows, missing bounds checks, type mismatches, and other flaws are ubiquitous and keep recurring. This seems rather ridiculous.**

Software Flaws 2

- Multics prevented stack overflows.
- Progr. languages are a mixed bag.
- Analysis tools: StackGuard (Cowan), buffer overflow analyzer (Wagner), lint family, Coverity (Engler), Fortify (Chess), MOPS (Chen); Microsoft: Spec#/Boogie, PREfast/PREfix, RaceTrack, ...

Some Privacy Problems

- Surveillance
- Accidental release/interception
- Misuse of personal information
- Identifiers and authentication
- Identity fraud
- Mistaken identities
- Spamming, phishing, ...
- Voter privacy, coercion, vote selling, ...

Some Systems with Privacy Issues

- Surveillance: FISA, Carnivore, DCSNet, ADVISE, PATRIOT Act, Protect America Act
- REAL-ID (slippery slope)
- US-VISIT (DHS)
- CAPPS II, No-Fly List (TSA)
- RFID chips (slippery slope)
- US Passports
- EEVS & E-Verify (DHS/SSA)
- FBI/UK DNA databases
- Electronic voting systems

Paradigmatic Example: Voting 1

- **Elections should have end-to-end integrity/reliability/accountability, nonsubvertible audit trails, uncompromised voter privacy, etc., throughout the entire process.**
- **Conceptually simple? Not really. Existing standards, systems, evaluations are seriously flawed.**

Voting 2: Weak Links

- Unfortunately, the entire process is vulnerable: voter registration, authentication, authorization; voting, counting, certifying, recounting, resolving disputes; depoliticizing the process, etc.
- Every step is a potential weak link; we have weakness in depth.

Voting 3: All-Electronic Systems

- **Today's all-electronic paperless systems are unauditable, lacking integrity and auditability under undetectable errors and fraud, proprietary code/data/evaluations, unable to resolve discrepancies, with many nontechnological problems.**
- **HAVA, EAC, voluntary standards, evaluations are still simplistic.**

Voting 4: Theory vs Practice

- Huge differences exist between research and practice/standards/evaluation/certification/...
- Many problems are nontechnical (absentee ballots, vote selling, voter coercion, politics, ...).
- Big opportunities for crypto and low-tech solutions.
- California Secretary of State's Top-to-Bottom review, 2007!!!

Lessons (To Be) Learned 1

- We need trustworthy systems, with dramatic improvements in system development practices.
- We need proactive attention to critical infrastructures.
- 20-200 foresight is much better than 20-20 hindsight.
- Priorities must be realistic.
- Progress in trustworthiness has been extremely sporadic.

Lessons (To Be) Learned 2

- **Reliance on misapplied technology usually increases risks.**
- **Privacy is often not appreciated until it is lost, and then may be impossible to recover.**
- **Privacy is difficult to ensure. Worse yet, it is often sacrificed in misguided hopes for security.**
- **Eternal vigilance is required, with proactive maintenance.**

Lessons (To Be) Learned 3

We need

- Better system architectures
- Better system engineering
- Better public-private cooperation
- Better education
- Privacy-aware crypto
- Intelligent incentives
- and lots more ...

Lessons (To Be) Learned 4

- **Attackers have many advantages over defenders. However, too often systems collapse on their own without provocation.**
- **Don't overendow technology. Every would-be technological solution has some risks.**
- **Let's not wait for disasters.**

Effective Forcing Functions?

- Market forces are inadequate.
- Open systems, interfaces, sources, and supporting incentives?
- Regulation, liability?
- Insurance and tax incentives?
- Better awareness of the risks of untrustworthiness; disasters?
- Maybe some of all of the above?
- But there are no easy answers.
- What can YOU do?

A Question To Ponder

Based on your own experience, does this talk seem

- **Heretical?**
- **Evolutionary?**
- **Revolutionary?**
- **Empirical?**
- **Timely?**
- **Impossible in the real world?**
- **Common-sense?**
- **Absolutely essential?**
- **Largely old-hat (1960s-1970s)?**

Conclusions 1

- -

- **We need long-term total-system life-cycle approaches, far-sighted optimization, and much more.**
- **Above all, we need far-sighted research, development, commitment to proactive scientifically sound approaches, and corresponding education. We should not trust our lives to untrustworthy systems!**

Conclusions 2

- Specific examples are perhaps less important than the fact that the the same types of problems keep recurring. Who's asleep at the wheel? Blame is distributed.
- Many common vulnerabilities can be relatively easily avoided with more principled approaches.

Conclusions 3

- **Computer development is mostly an incremental process, driven by marketplace forces. But security research and assurance are slow to be adopted, despite vital needs.**
- **Incentives are needed to make better use of past lessons.**

Conclusions 4

- **Better trustworthiness is urgently, needed, and should be approached holistically, with composable architectures and principled system developments.**
- **Development and operation of trustworthy critical systems require massive cultural changes.**

A Few Relevant References

- U.S. Government Accountability Office reports on DHS, US-VISIT, EEVS, etc., http://www.gao.gov
- National Academies Press, National Research Council, CSTB, www.nap.edu
  - ⋆ Toward a Safer and More Secure Cyberspace, 2007
  - ⋆ Trust in Cyberspace, 1998
  - ⋆ Computers at Risk: Safe Computing in the Information Age, 1990

References on Principles

- **J.H. Saltzer and M.D. Schroeder, The Protection of Information in Computer Systems,** *Proc. IEEE 63,* **9, 1278-1308, September 1975**

http://www.multicians.org

- **PGN, Role of Motherhood in the Pop Art of System Programming, ACM Symposium on Operating System Principles, October 1969**

multicians.org/pgn-motherhood.html

A Few Recent PGN References

- **Reflections on System Trustworthiness, Advances in Computing volume 70, 2007.**
- **Holistic Systems,** *ACM SIGSOFT Softw.Eng.Notes,* **Nov. 2006.**

http://www.csl.sri.com/neumann/holistic.pdf

- **Principled Assuredly Trustworthy Composable Architectures, 2004.**

http://www.csl.sri.com/neumann/chats4.html, .pdf, .ps

Old PSOS/HDM References

- **Neumann, Boyer, Feiertag, Levitt, Robinson, A Provably Secure Operating System: The System, Its Applications, and Proofs, SRI 1980. http://www.csl.sri.com/neumann/**
  psos/psos80.ps
  psos03.pdf
- **L. Robinson, K.N. Levitt, Proof Techniques for Hierarchically Structured Programs,** *CACM,* **Apr 1977.**

Two Privacy References

- -

- **S.M. Bellovin, M.A.Blaze, W. Diffie, S. Landau, PGN, J. Rexford, Risking Communications Security: Potential Hazards of the "Protect America Act",** *IEEE Security and Privacy, 6,* **1, Jan-Feb 2008, pp. 18–27.**
  http://crypto.com/paa.pdf
- **PGN, Security and Privacy in the Employment Eligibility Verification System (EEVS), House Subcomm. on Social Security, 7 Jun 2007.**
  csl.sri.com/neumann/house07.pdf

A Few More PGN References

- **Computer-Related Risks, Addison-Wesley, 1995**
- **ACM Risks Forum, www.risks.org**
- **http://www.csl.sri.com/neumann**