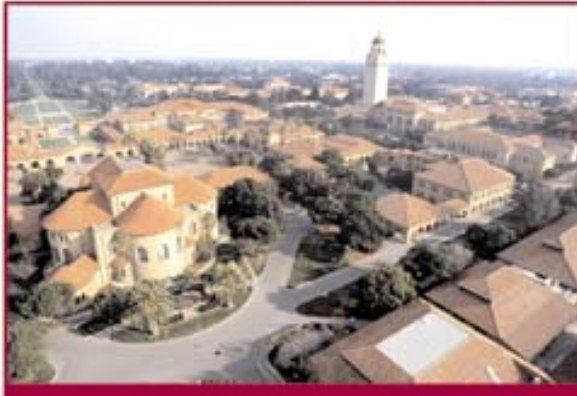


# Welcome!



*Bridging Stanford and Industry*

## INCOSE November, 2006

**Neil Daswani**



Stanford Center for  
Professional Development



# Outline

- ❖ What is security?
- ❖ Why is it important?
- ❖ How to design for security?
- ❖ How to organizing for security?
- ❖ Emerging Threats



## Security Goals: What is security?

- ❖ Authentication
- ❖ Authorization
- ❖ Confidentiality
- ❖ Data / Message Integrity
- ❖ Accountability
- ❖ Availability
- ❖ Non-Repudiation



# Why security?

- ❖ “Hurting customers”
  - Privacy Considerations
  - Vulnerable software
  - Stolen accounts
- ❖ Bad Press / Reputation Risk
- ❖ Financial Risk (including valuation)
- ❖ SOX / HIPPA / EU Compliance
- ❖ Waste of company time / resources



## Business / Valuation Risk

- ❖ Can ruin business (i.e., DoubleClick, CardSystems)
- ❖ On average, 0.63% decrease in valuation on the day of disclosure (self or third-party)  
For MSFT, -> ~\$1.7B loss in valuation
- ❖ (Based on security economics research. See [http://infoecon.net/workshop/pdf/telang\\_wattal.pdf](http://infoecon.net/workshop/pdf/telang_wattal.pdf) )



## Other Risks

- ❖ Financial Risk
  - Security bugs in financially-related software may result in direct monetary loss
- ❖ SOX Compliance
  - Without proper internal controls in place, executives, and others might go to jail!
- ❖ HIPPA / EU Compliance



## Waste of time

- ❖ Security vulnerabilities & exploits mean:
  - Incident response
  - Possible press response
  - Management time (coordinate patch)
  - Engineering time (fix, test, debug, rollout)
  - Damage assessment
  - Possible executive attention



# How do you get security?

## Approaches & Trade-offs:

- ❖ Risk Management
- ❖ “Designing-In” Security
- ❖ Security By Obscurity
- ❖ Open vs. Closed Source
- ❖ Threat Modeling
- ❖ Convenience vs. Security

## Principles:

- ❖ Least Privilege
- ❖ Fail-safe Stance
- ❖ Defense in Depth
- ❖ Secure Weakest Link
- ❖ Simplicity
- ❖ Usability
- ❖ Security Features vs. Security



# Risk Management

- ❖ How do we think about security?
  - All systems insecure: how insecure?
  - What is the cost to “break” the system?
  - For every \$ that attackee spends, how many \$ does attacker have to spend?
  - If (Cost to “break” system  $\gg$  Reward to be gained)
    - then system is secure
    - otherwise system is NOT secure
  - “Raise the bar” high enough





# Design In Security

- ❖ Design products with security in mind.
- ❖ Hard to “add-on” security later.
- ❖ Don’t add security as an afterthought.
- ❖ Define concrete, measurable security goals.
  - Only certain users should be able to do X.
  - A log entry should be written every time that a user does X.
  - Information output by feature Y should be encrypted.
  - Feature Z should be available 99.9% of the time.
- ❖ Security is an important “feature” itself, but it is not just a feature– it is about the process by which the product is developed, tested, deployed, and maintained.



## Security By Obscurity

- ❖ Kerckhoffs' doctrine (1883)
- ❖ "The method used to encipher data is known to the opponent, and that security must lie in the choice of key."
- ❖ Compromised key can be changed without re-designing system.





# Open vs. Closed Source

- ❖ “Is open-source software secure?”
- ❖ Open:
  - Some people might look at the security of your application (if they care)
  - They may or may not tell you what they find
- ❖ Closed:
  - not making source code available does not hide much
  - security-aware code reviews by diverse reviewers needed
- ❖ A business decision: Not a security one!



## Security Features vs. Security

- ❖ Using one or more security algorithms/ protocols will not solve all your problems!
  - Ex. Using encryption doesn't protect against weak passwords.
  - Ex. Using SSL doesn't protect against buffer overflows.

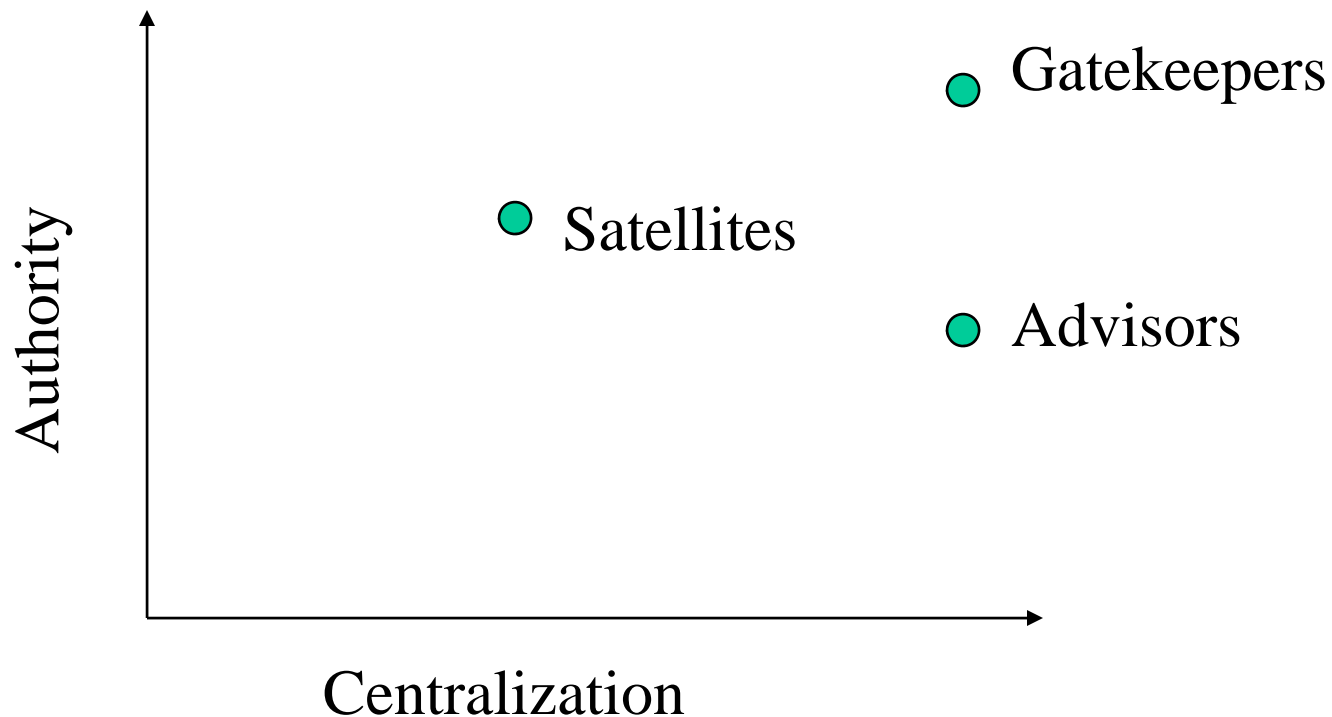


# Organizing for Security

- ❖ Security is a process; not a feature!
- ❖ Where is security expertise needed?
  - Operations (Network and OS security; firewalls, IDSes, etc.)
  - R & D (Architectural / Design)
  - Testing (Code Auditing, Penetration Testing)
- ❖ How to organize security for new product R & D
  - Gatekeepers
  - Advisors
  - Satellites



## Possible Organizations





# Gatekeepers

- ❖ Centralized Security Department with Approval Authority
- ❖ Security Dept accountable for every line of deployed code, and must provide explicit approval for every deployment.
- ❖ Pros:
  - High level of accountability
  - Tight control
- ❖ Cons:
  - Scalability
  - Could stifle innovation
  - Bottleneck
  - Development might become tight-lipped (or work-around security)



## Advisors

- ❖ Security Consulting Department with Escalation Authority
- ❖ Security Department provides feedback to product teams when requested, or can actively “probe”
- ❖ Pros
  - More openness to share risks
- ❖ Cons
  - Less accountability
  - Frequent escalation will de-sensitize executives



# Satellites

- ❖ Decentralized Security Staff / “Virtual” Department
- ❖ Put developers with security expertise on the product teams. Rotate if necessary.
- ❖ Or, train one of the developers on each product team to be “security czar.”
- ❖ Pros
  - Security recommendations more likely to be implemented
- ❖ Cons
  - Less flexibility in moving security engineers to most high risk projects fast.



# Emerging Threats

- ❖ Phishing / Pharming
- ❖ Spyware / Adware / Malware
- ❖ Extortion
  - Online gambling sites, banks
- ❖ Botnets
  - DDoS
  - Spam
  - Click Fraud
- ❖ Mobile Worms
- ❖ VoIP Security



# Phishing

- ❖ Impostor emails with links to “spoofed” web sites
- ❖ “Solutions”
  - Toolbars of various kinds (Cyota, etc.)
  - Google Anti-phishing Firefox Extension
  - PassMark
  - PasswordHash
  - Out-of-band callback



# Pharming

- ❖ aka DNS Cache Poisoning
  - Vulnerability in local DNS server
  - Virus adds to .hosts file
- ❖ Solutions
  - SecureDNS
  - Well-written DNS server (i.e., Nominum)



# Spyware / Adware / Malware

- ❖ Spyware: sends information outbound without informed consent
- ❖ Adware: displays ads to users with or without informed consent
- ❖ Some software (i.e., P2P programs) depend upon spyware/adware for their business
- ❖ Malware: keystroke logging, worms
- ❖ November 2004 AOL & National CyberSecurity Alliance
  - 80% of users PCs infected with Spyware
    - 89% of those users didn't know they were infected
    - 95% said they did not give permission
- ❖ Solutions:
  - Google Pack
  - Microsoft provides for free (beta until June '06), and free in Vista
  - Webroot SpySweeper
  - AdWare



# Extortion

- ❖ “Pay us or we will attack (DoS) your site”
- ❖ DoS = Denial-of-Service
- ❖ Solutions
  - Pay ransome
  - Overprovision
  - Change IP address
  - Signature-based detection
  - IP Traceback
- ❖ Latest: Distributed Reflected DNS attacks



# Botnets

## ❖ Applications

- DDoS
- Spamming (E-mail)
- Password Sniffing
- Keylogging (defeats SSL)
- Spreading malware / spyware / adware
- Click fraud
- Online polling / gaming
- Mass identity theft

## ❖ Largest: Dutch botnet – 1.5 million machines

- Threatened a US company



# Mobile Worms

## ❖ Analagous to desktop / PC worms, BUT

- Heterogeneous OS environment
- Many more attack “vectors”
  - Bluetooth (i.e., Cabir)
  - SMS / MMS
  - E-mail
  - 802.11, 802.16
- Solutions:
  - Could be deployed by carriers, handset manufacturers, or end-users




# VoIP Security

- ❖ Using Internet for voice phone calls
- ❖ Skype (P2P), Vonage (dedicated), etc.
  - Skype has tens of millions of users
  - Avg Skype call ~ 19 minutes vs. 2-3 minutes PSTN
  - Skype uses users' distributed PCs to setup & route calls
  - Work-day usage pattern
- ❖ Threats:
  - (inherits all Internet threats)
  - i.e., DDoS against ER as a complement to physical terrorist attack



# Questions and Contact

❖ <http://www.neildaswani.com/>



**Neil Daswani, PhD**

NEWS | EXPERIENCE | EDUCATION | PUBLICATIONS | PATENTS | PROJECTS

Check out my new book entitled "What Every Programmer Needs To Know About Security"  
(Publisher is currently accepting pre-orders!)

Learn about security via online, multimedia courses from Stanford!



## An experimental study of P2P VoIP

3/07/2006 08:23:00 AM

Posted by Neil Daswani & Ravi Jain, Google; and Saikat Guha, Cornell University

VoIP (Voice-over-IP) systems are one of the fastest growing means of communication on the Internet, enabling free or low-cost phone calls. But to date, researchers have had little data to work with to learn how to build VoIP systems better. Some of these systems are proprietary, and obtaining data about their operational characteristics has been particularly challenging. For instance, even though the Skype network has tens of millions of users, it has been hard for researchers to benefit from its commercial success.



Springer

Foundations of Computing

Select your subdiscipline

Select a discipline

> Home / Computer Science / Foundations of Computing



**What Every Programmer Needs to Know about Security**  
Daswani, Neil; Kesavan, Anita (Eds.)  
2006, Approx. 180 p. 20 illus., Hardcover  
ISBN: 0-387-30240-9

Not yet published. Available: June 2006

US\$59.95



[About this book](#) | [Table of contents](#)

## About this book

**What Every Programmer Needs to Know about Security** introduces software professionals to the mindset and techniques they need to know to build secure software systems. Software has become part of the world's critical infrastructure, but typically is not well protected from attacks. Programmers to date, have traditionally been taught to focus on performance and correctness, which is unfortunately not enough in a networked world of constantly-attacking hackers. This book teaches programmers how to also focus on safety, reliability, and security so that software can withstand attack. Once enabled with the knowledge presented in this book, professionals can start to alleviate some of the inherent vulnerabilities that make today's software so susceptible to attack.



# Questions and Contact

❖ <http://proed.stanford.edu/?security>

Stanford Center for Professional Development  
STANFORD UNIVERSITY



Course Information

- [Home](#)
- [Certificate Program](#)
- [Module Descriptions](#)
- [Instructor](#)
- [Instruction Methods](#)
- [Registration & Pricing](#)
- [FAQs](#)
- [News & Publications](#)
- [Related Offerings](#)
- [Other Programs](#)
- [Contact Us](#)

## Computer Security Certificate Program

### ONLINE

Computer security is a critical part of business strategy. Companies are now demanding new levels of risk management, assessment, reporting, and protection. Engineers with the ability to develop secure software from the ground up are valuable in today's market, producing more efficient, cost-effective and consumer-friendly programs.

▪ **Certificate Objectives**

Topics include software design principles, symmetric encryption and public-key cryptography, as well as strategies to defend software against adversaries such as worms and hackers. The certificate consists of three online modules, taught by [Neil Daswani](#), Stanford Ph.D. and a specialist in designing security into financial software and wireless networks..

▪ **Who Will Benefit**

Software programmers, architects, developers, and engineers who are interested in learning the ins and outs of designing secure programs. This program also is a great tool that IT managers, CIO's, and CSO's can use to educate their workforce about security issues.

▪ **Courses**

- Computer Security Principles
- Introduction to Cryptography
- Secure Programming Techniques



# Stanford Center for Professional Development



The Stanford Center for Professional Development collaborates with Stanford University faculty to offer graduate degrees, courses, and professional development short courses to meet the career-long learning needs of engineers, scientists, technology professionals and managers in industry. Courses are delivered via the Internet and local broadcast as well as on campus.



❖ Backup slides follow



# Convenience vs. Security

- ❖ Usually inversely proportional
  - More secure => Less convenient
  - Too inconvenient => Less secure
- ❖ If too inconvenient => unusable => users will workaroud => insecure
- ❖ Good technologies increase both security and convenience/usability



## Principle of Least Privilege

- ❖ Just enough authority to get the job done.
- ❖ Common world ex: Valet Keys
- ❖ Highly Elevated privileges usually unnecessary.



## Fail-Safe Stance

- ❖ Common world ex: Elevators
- ❖ System failure should be expected (and planned for)
- ❖ Ex: If firewall fails, let no traffic in
- ❖ Deny access by default



# Fail-Safe Stance

- ❖ Two possible designs for  
`int checkPassword (String username, String password)`
- ❖ Function could fail: what should function return?
  - 1) `ERROR_ACCESS_DENIED`  
`ERROR_PASS_FILE_NOT_FOUND`  
`ERROR_OUT_OF_MEMORY`  
`NO_ERROR_ACCESS_ALLOWED`
  - 2) `NO_ERROR`  
`ERROR`  
`int getError ()`



## Fail-Safe Stance

- ```
❖ int result = checkPassword ( ... )
  if (result == ERROR_ACCESS_DENIED) {
      abort();
  }
  else {
      // Complete login
  }
```
- ❖ Problem: `result != ERROR_ACCESS_DENIED` does not infer `ERROR_ACCESS_ALLOWED`
  - ❖ Result could have been `ERROR_PASS_FILE_NOT_FOUND` or `ERROR_OUT_OF_MEMORY` !



## Fail-Safe Stance

- ```
❖ int result = checkPassword ( ... )
  if (result == NO_ERROR) {
    // Complete login
  }
  else {
    int reason = getError();
    abort();
  }
```
- ❖ Much better— less error prone!
  - ❖ checkPassword failure occurs securely!



# Usability

- ❖ Will not read documentation  
(Enable security by default)
- ❖ Users are lazy  
(They ignore security dialogs)
- ❖ **Secure by default features in software forces users and vendors to be secure.**



# Usability for Security\*

Users need to:

- 1) be made aware of security tasks
- 2) be able to figure out how to successfully perform security tasks
- 3) NOT make dangerous errors
- 4) be comfortable enough with UI to continue using it

\*definition coined by Alma Whitten