Design Flaw #6: Use Cryptography Correctly

Laurie Williams
williams@csc.ncsu.edu

Use Cryptography Correctly

Through the proper use of cryptography, one can protect the confidentiality of data, protect data from unauthorized modification, and authenticate the source of data.

Pitfalls in Applying - 1

- Rolling your own cryptographic algorithms or implementations.
- Misuse of libraries and algorithms
  - Incorrect assumptions
- Poor key management
  - hard-coding keys into software (often observed in embedded devices and application software)
  - failure to allow for the revocation and/or rotation of keys
  - use of cryptographic keys that are weak (such as keys that are too short or that are predictable)
  - weak key distribution mechanisms

Pitfalls in Applying - 2

- Randomness that is not random
  - Need random numbers with strong cryptographic randomness properties
  - Cannot reuse random numbers
- Failure to centralize cryptography choice within a team/organization
  - Different cryptographic algorithms often don't interact nicely.

… *work with an expert, if possible*

Related Principle: Don’t reinvent the wheel

http://www.codinghorror.com/blog/2009/02/dont-reinvent-the-wheel-unless-you-plan-on-learning-more-about-wheels.html

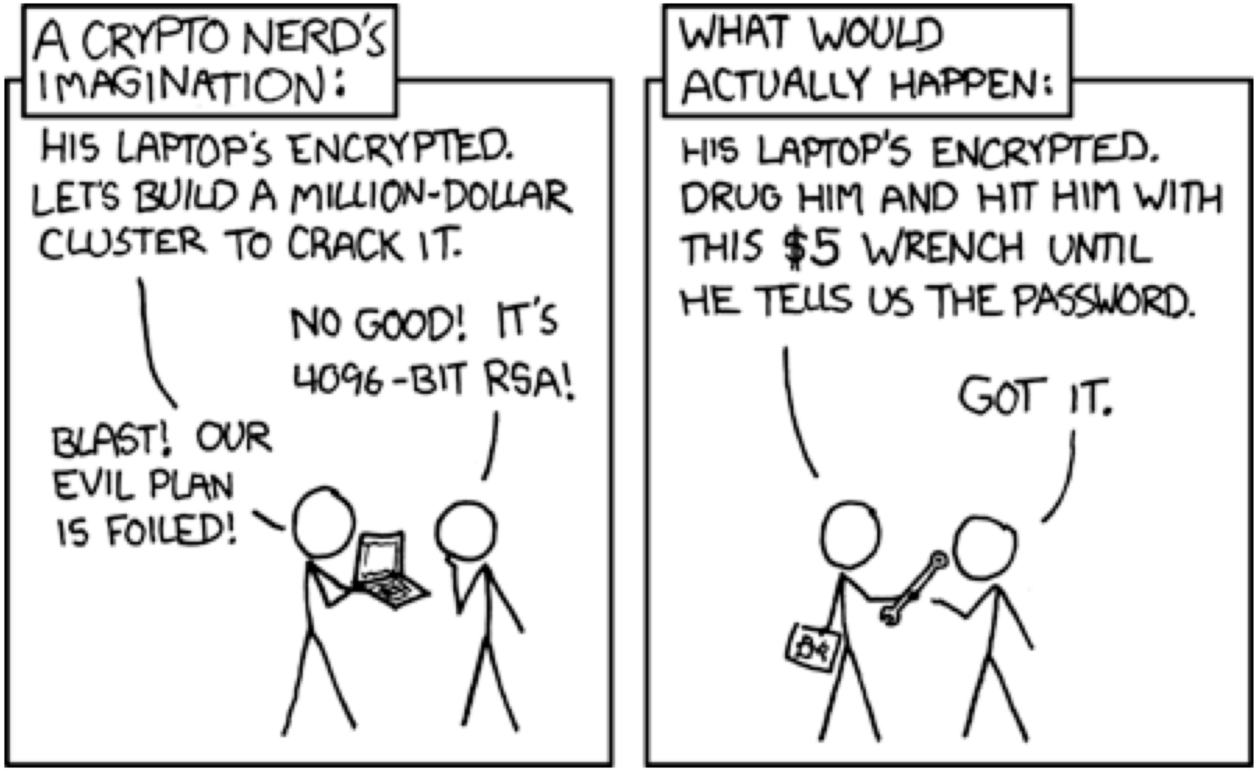
Respect their expertise

Hibernate

Credit card authorization

Encryption

Input validation frameworks


A CRYPTO NERD'S IMAGINATION:
HIS LAPTOP'S ENCRYPTED. LET'S BUILD A MILLION-DOLLAR CLUSTER TO CRACK IT.
NO GOOD! IT'S 4096-BIT RSA!
BLAST! OUR EVIL PLAN IS FOILED!
WHAT WOULD ACTUALLY HAPPEN:
HIS LAPTOP'S ENCRYPTED. DRUG HIM AND HIT HIM WITH THIS $5 WRENCH UNTIL HE TELLS US THE PASSWORD.
GOT IT.

http://xkcd.com/538/

Design Flaw #7: Identify Sensitive Data and How They Should be Handled

Laurie Williams
williams@csc.ncsu.edu

Identify Sensitive Data

- Comes from many sources
  - Users input
  - … and the Personally-identifiable information (PII) of users
  - Data computed
  - Sensors (e.g. geolocation, accelerometer)
- Must: Create a policy that explicitly identifies different levels of classification

Handling Sensitive Data - 1

- Procedures may be provided (a.k.a. non-negotiable) … though these are changeable over time
  - Regulation (e.g. HIPAA; Payment Card Industry (PCI); EU Data Protection Directive)
  - Company policy (e.g. privacy policy)
  - Contractual obligation
  - User expectation
    - More subjective than the rest

Handling Sensitive Data- 2

- Data sensitivity can be context-sensitive (temporal, location, situation)
  - Availability of medical data over confidentiality
  - “Break the Glass” scenario

Hurricane Katrina, New Orleans, 2005

http://en.wikipedia.org/wiki/Effects_of_Hurricane_Katrina_in_New_Orleans

Things to consider

- Access control mechanisms (including file protection mechanisms, memory protection mechanisms, and database protection mechanisms)
- Cryptography to preserve data confidentiality or integrity
- Redundancy
- Data at rest
- Data in transit
  - Trust and trust enclaves

Design Flaw #8: Always consider the users

Laurie Williams
williams@csc.ncsu.edu

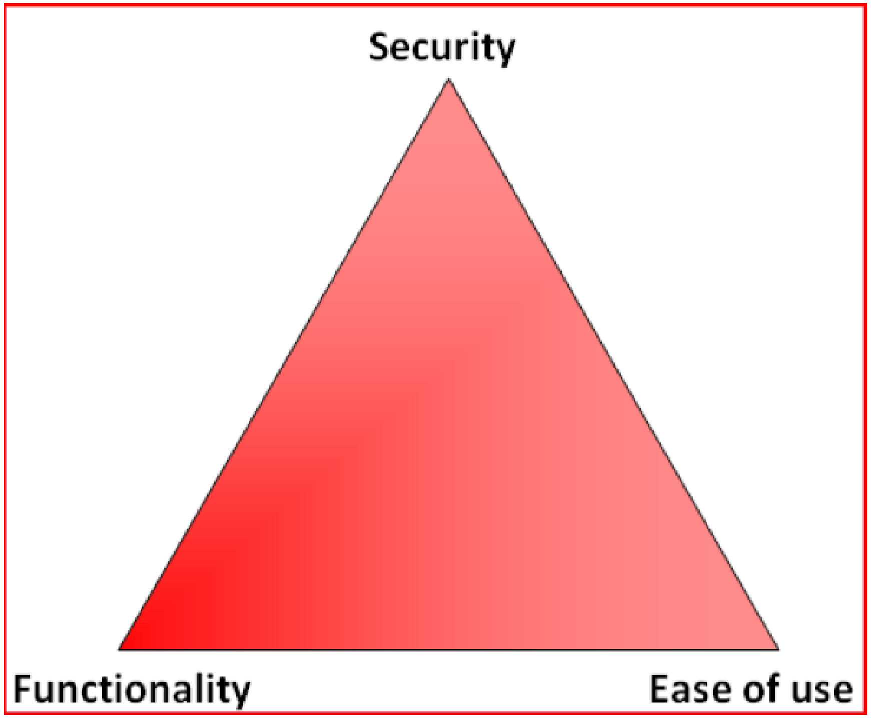
Users … Those Humans

No Grandma, Listen, Double-click the Internet Explorer Icon.

Users, attackers, developers …

h 302-grandma-hair/ and
h /chinese-hackers-now-hitting-major.html
h iow-to-become-a-programmer/
h xecutive-spotlight-joseph-cormier-of-gtec/

The Triad …



Focusing on one will severely impact the others

http://blog.infosanity.co.uk/2010/06/12/infosec-triads-securityfunctionalityease-of-use/

MORDAC, THE PREVENTER OF INFORMATION SERVICES.
SECURITY IS MORE IMPORTANT THAN USABILITY.
scottadams@aol.com
www.dilbert.com
IN A PERFECT WORLD, NO ONE WOULD BE ABLE TO USE ANYTHING.
© 2007 Scott Adams, Inc./Dist. by UFS, Inc
11-16-07
To complete the log-in procedure, stare directly at the sun.

Computer Science
NC STATE UNIVERSITY

Design Flaw #9: Understand How Integrating External Components Changes Your Attack Surface

Laurie Williams
williams@csc.ncsu.edu

What is an Attack Surface?

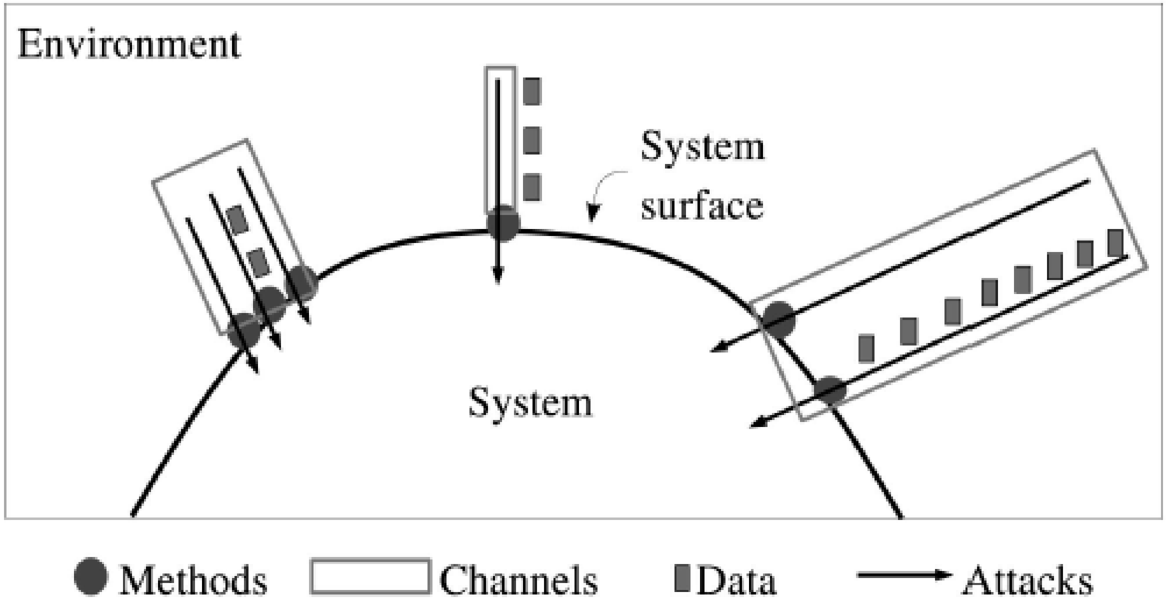


Fig. 2. A system's attack surface is the subset of the system's resources (methods, channels, and data) potentially used in attacks on the system.

Entry and exit points of a program/system

P. K. Manadhata and J. M. Wing, "An Attack Surface Metric," *IEEE Transactions of Software Engineering*, vol. 27, no. 3, pp. 371-386, May/June 2011.

Considering the attack surface

- the sum of all paths for data/commands into and out of the application;
  - the code that protects these paths (including resource connection and authentication, authorization, activity logging, data validation and encoding);
- all valuable data used in the application, including secrets and keys, intellectual property, critical business data, personal data and personally identifiable information (PII); and
  - the code that protects these data (including encryption and checksums, access auditing, and data integrity and operational security controls)

Attack surface analysis

- To understand and manage application security risks as applications and operating systems are designed and changed in a software system. The goal is to close all but required entry and exit points leading to and from system assets and to constrain others with access rights, monitoring, and response

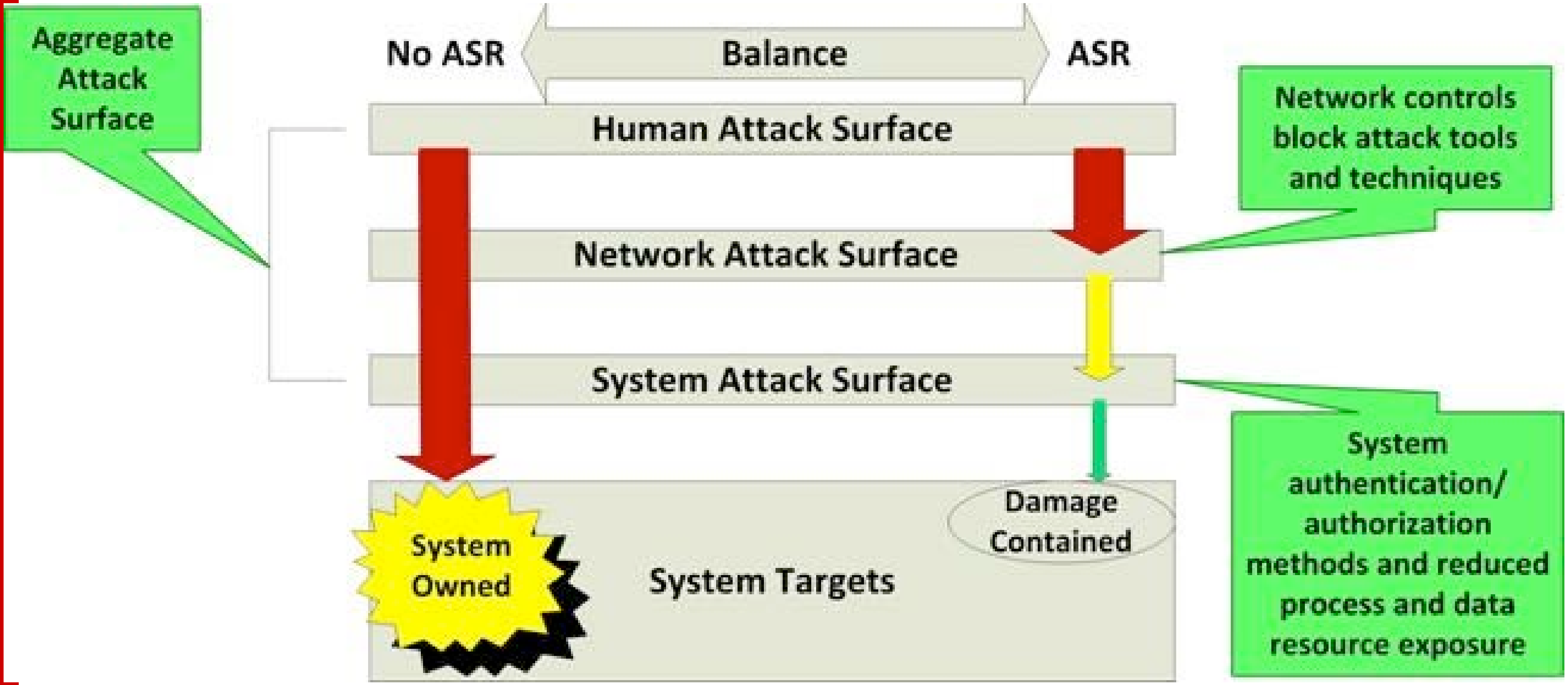
How to reduce the attack surface

- Keep entry and exit points to a minimum and allow users to enable functionality as needed.
  - open sockets (TCP and UDP)
  - open named pipes
  - open remote procedure call (RPC) endpoints
  - services
  - services running by default
  - services running in elevated privileges
  - dynamic content Web pages
  - account you add to administrator’s group
  - files, directories, and registry keys with weak access control lists

Attack Surface Comparison

High Attack Surface	Low Attack Surface
Features running by default	Feature off by default
Open network connections	Closed all unnecessary connections
System always on	System intermittently on, as needed
Anonymous access	Authenticated access
Code running will full admin privileges	Code running under “least-privilege” account
Uniform defaults	User-chosen settings, secure by default
Larger code	Smaller code
Weak Access Control Lists (ACLs)	Strong Access Control Lists (ACLs)

Defense in depth and the attack surface



http://resources.infosecinstitute.com/attack-surface-reduction/

Components change the attack surface

- OTS components, platform, applications
- Third party open source or proprietary libraries
- Widgets and gadgets loaded at runtime as part of a web project
- Software developed by a different team
- Software your team developed at a different point in time
- ….

… as binaries, source code, API …

What to do …

- Isolate components as much as possible
- Configure to only open functionality you will use
- If the component cannot be configured to comply with your security policy, don’t use it
- Look at vulnerability history in CVE database
- Maintain up-to-date components
- Maintain a healthy distrust
- Authenticate dataflow
- Consider data coming in untrusted

Design Flaw #10: Be Flexible When Considering Future Changes to Objects and Actors

Laurie Williams
williams@csc.ncsu.edu

http://www.zlien.com/articles/mississippi-lien-waivers-new-scheme/

Be flexible

- Software security must be designed for change, rather than being fragile, brittle, and static.
  - Design for secure updates
  - Design for security properties changing over time; for example, when code is updated.
  - Design with the ability to isolate or toggle functionality.
  - Design for changes to objects intended to be kept secret (e.g. password recovery)
  - Design for changes in the security properties of components beyond your control.
  - Design for changes to entitlements (i.e. authorizations).

https://infocus.emc.com/tim_layton/incident-detection-woes-because-of-security-skills-shortage/