Peering Behind the Curtain:
Evaluating Your Security Vendor

PENTCIRT Security Forum
August 5, 2003

Nate Lawson, Senior Security Engineer
nate@cryptography.com
(415) 397-0123

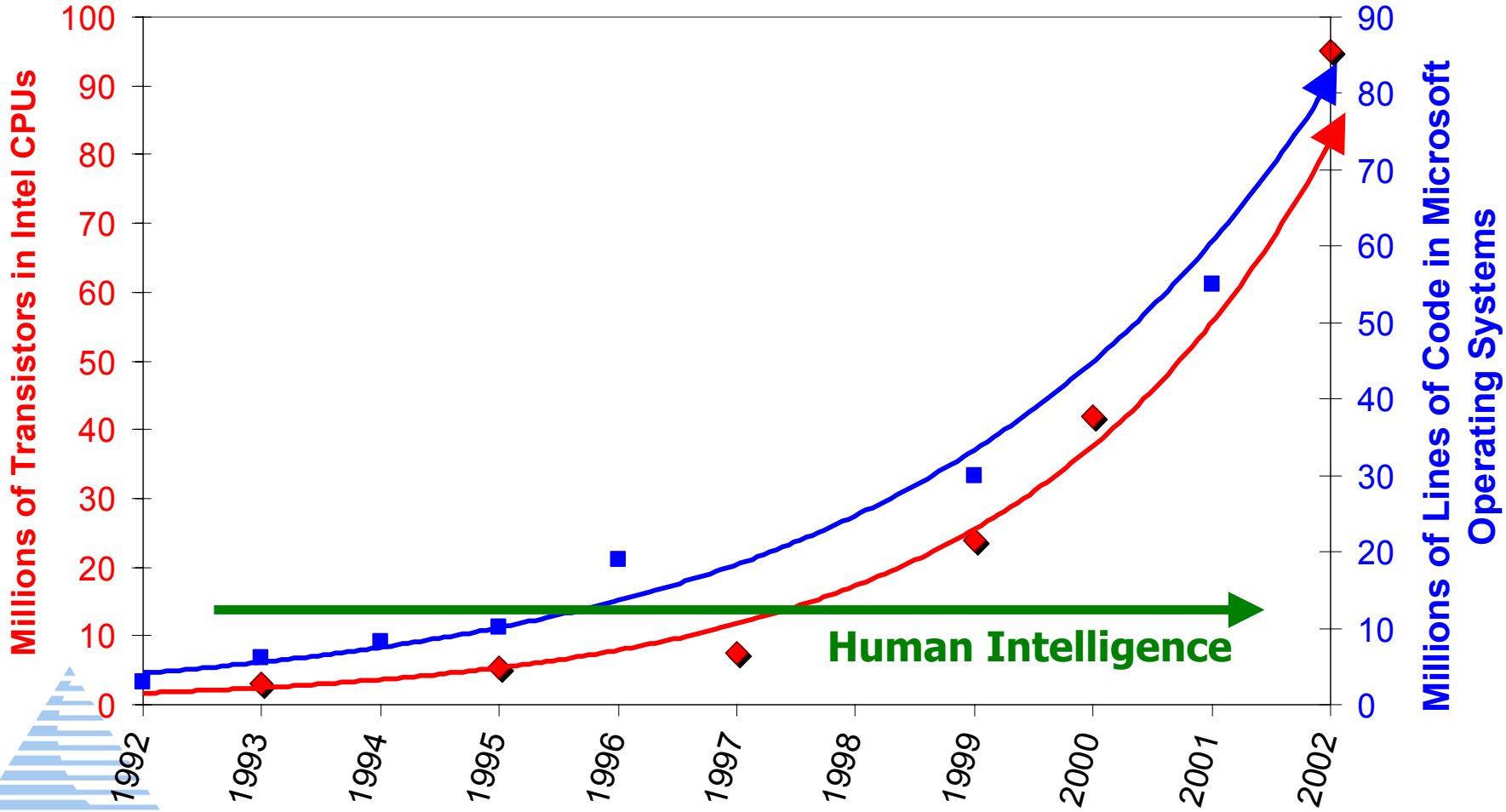
Cryptography Research, Inc.
www.cryptography.com
607 Market St., 5th Floor, San Francisco, CA 94105

Security is hard…

- Myriad of products to choose from
  - Firewalls, virus scanners, IDS, etc.
- Unclear vendor claims
  - What is "intrusion prevention" exactly?
  - Vendor's goal is to be just secure enough
- Inconclusive certifications
  - One Level 2 product may offer more assurance for your application than another
  - It's always what they didn't test that gets you (i.e. DPA)

- Solution: the Security Evaluation process gives insight into a product's capabilities, tailored for your risk profile.

Managing Complexity



CPU data courtesy Intel Corp.

About Cryptography Research

- Consulting, licensing & research
  - Consulting: Evaluation, implementation, design
  - Licensing: Tamper-resistance, content security, DPA, pay TV
  - Research: Real security problems & responses
- Emphasis on applied work
  - Practical, reliable solutions to real problems
  - Systems designed by CRI engineers protect >$50B annually
  - Most of our revenue from big companies with real losses
- Specialize in high-risk commercial systems
  - Highly technical; Hands-on + theoretical expertise
    - Crypto, risk management, hardware, networking…
  - Industries: Financial, content, pay TV, communications

About Nate

- Developed ISS RealSecure
  - First commercial network-based IDS
- Worked for InfoGard
  - FIPS 140 testing for IBM, Motorola, Netscape
- Developed Decru's SAN product
  - Prototype fibre channel encryption product
- Works for Cryptography Research
  - Evaluation: Neoteris, VIA
  - Design: Digi-Flicks, SPDC
- Moonlights on FreeBSD
  - Storage, ACPI, USB

Security evaluation

- Detailed security analysis
  - All information about the product is available
    - Specification, design documents, source code, pinouts
    - Vendor engineers answer questions
- Proven security evaluation team
  - Internal members provide application/risk insight
  - External members bring deep, expert technique
- Risk awareness
  - Address particular risks unique to desired application
- Often improves vendor product

Team Profile

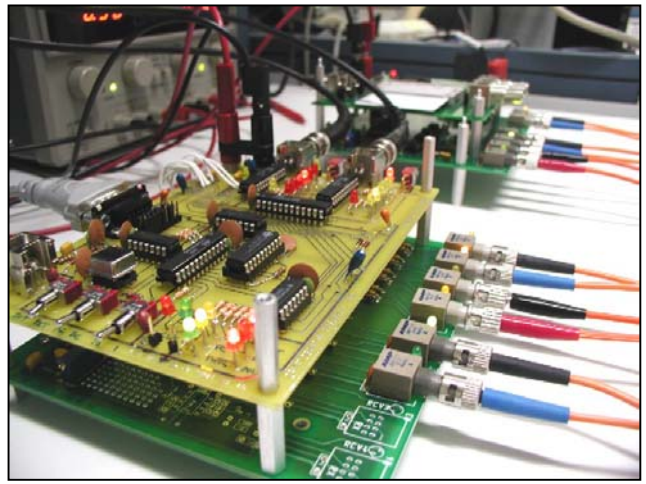
- Cross-discipline skills
  - Hardware, software, firmware
  - Soldering, mechanical, chemical
  - Documentation, specification, communication
- Proven results
  - Published research
  - Designed and analyzed widely-used systems
- Internal/External Composition
  - Internal people are more familiar with system design
  - External people are not entrenched in the system particulars

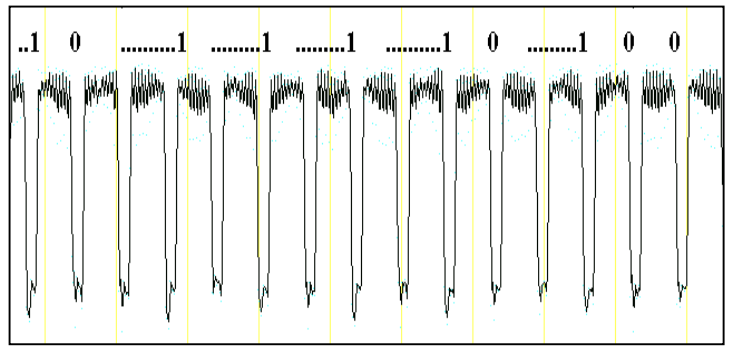
Blackbox testing is not an evaluation

- Vendor provides evaluator with the system but no documentation
- Why do it?
  - Need to prove something to effect a political change
- Why not to do it?
  - Success leaves unanswered questions
    - What other flaws are there?
    - What did the analysis actually prove?
  - Failure to compromise the system is inconclusive
    - Might have been successful with more time, equipment, creativity
  - 10X cost and time of a security evaluation

Example: Smart Card Evaluation

- Customer designing new smart card to be built by Vendor
  1. CRI contacted to evaluate design and implementation
  2. CRI reads vendor documentation and prepares a list of questions
  3. Vendor prepares answers. Repeat.
  4. Vendor addresses issues raised and provides sample implementation
  5. Customer requests CRI implement one possible attack
  6. CRI validates attack and vendor changes design according to advice

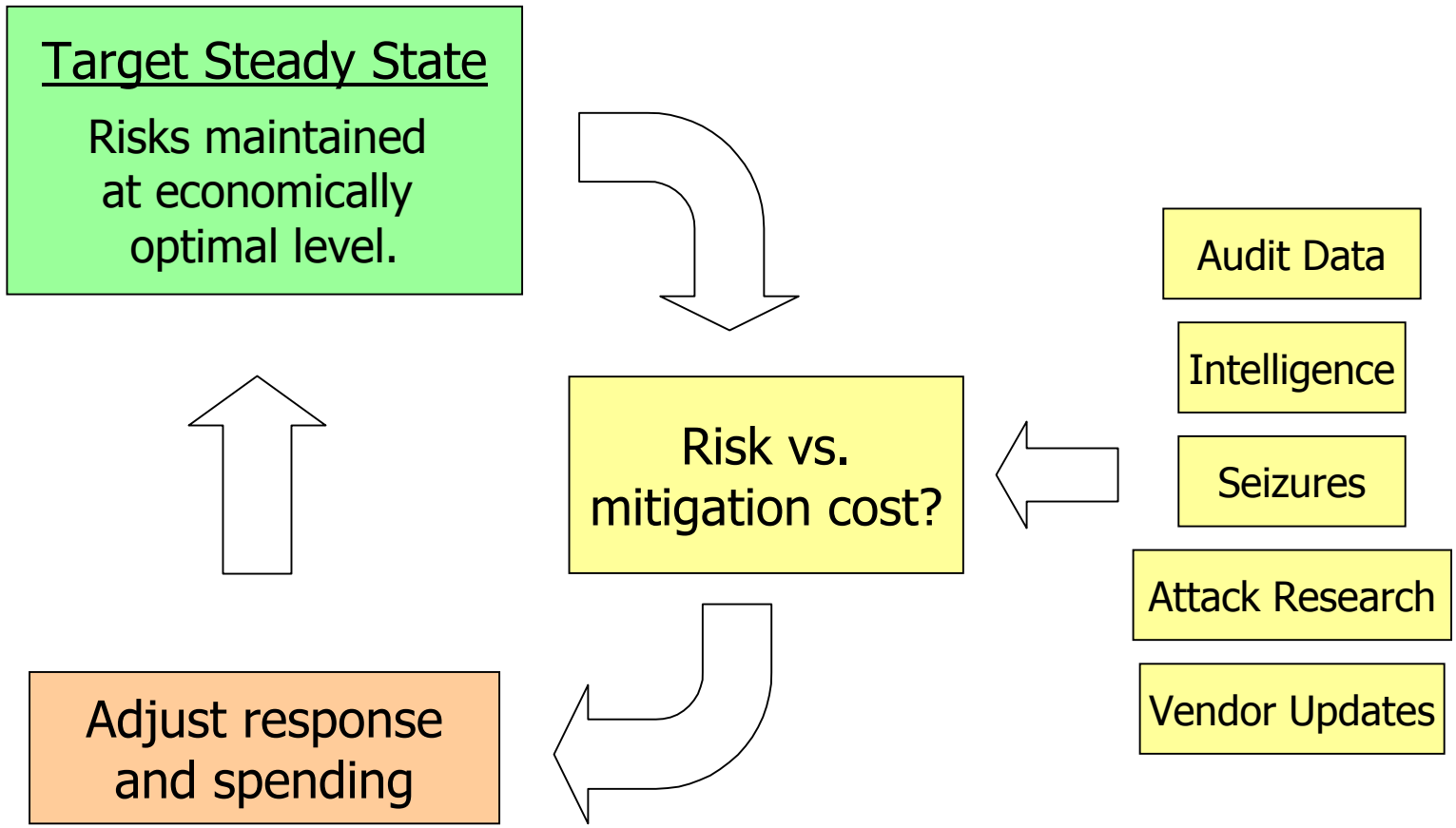




DPA test board and sample trace

CRYPTOGRAPHY RESEARCH

Risk Management: When Problems are Inevitable



CRYPTOGRAPHY RESEARCH

10 Suggestions for Evaluations

- Goal is higher assurance that system meets security requirements
- Remember: successful risk management requires a closed loop
  - Design
  - Implementation
  - Deployment
  - Monitoring (repeat)
- Security evaluation gives assurance at each stage

10 Suggestions

❶ View security in economic terms.

- Assign a dollar-value to your risk.
  - Get management support for the estimate.
- Spend before problems get out of control.

10 Suggestions

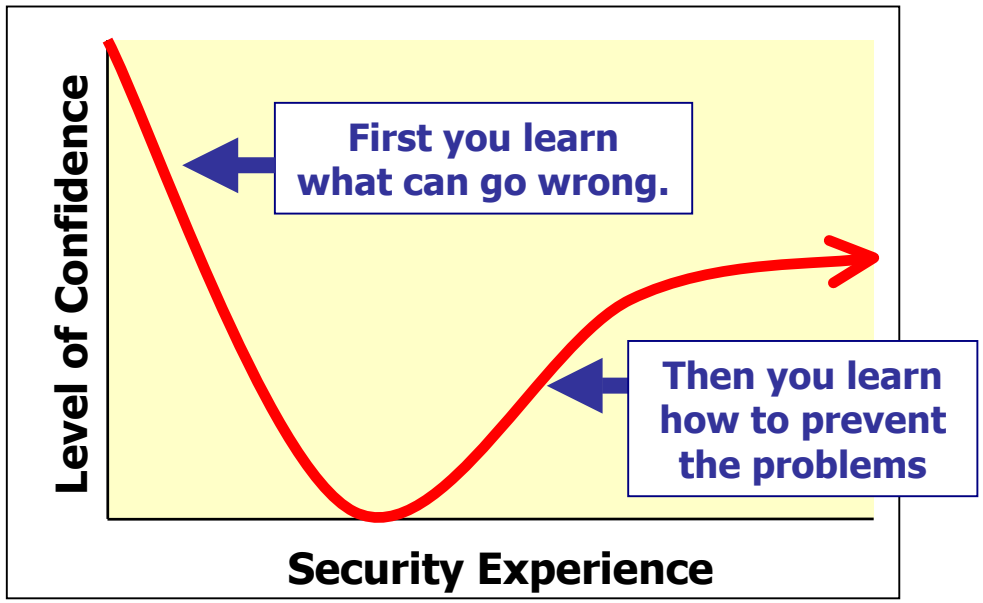
❶ View security in economic terms.

❷ **Think about how risk is allocated.**

- Where are the single points of failure?
- Will those who control your risk share it?
- Are the people you trust actually trustworthy?
  - What is their historical track record?
  - Do they make unsubstantiated claims of "security"?
  - Who can vouch for their work?

10 Suggestions

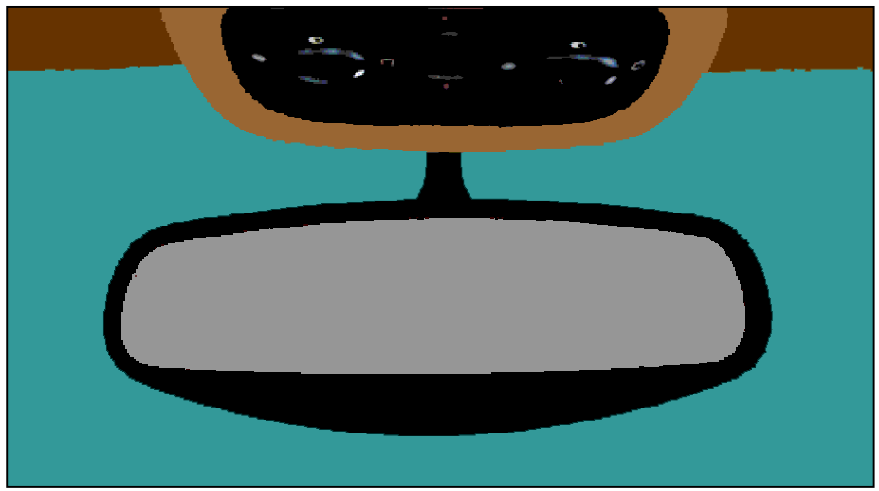
1. View security in economic terms.
2. Think about how risk is allocated.
3. **Be humble and know your limits.**
   - Don't mistake confidence for experience.
   - Encourage people to look for flaws in your work.
   - Don't assume attackers won't "figure it out".



CRYPTOGRAPHY RESEARCH

10 Suggestions

1. View security in economic terms.
2. Think about how risk is allocated.
3. Be humble and know your limits.
4. **Make realistic assumptions.**
   - Assume that users are lazy and gullible.
   - Assume that engineers make mistakes.
   - Beware of the rear view mirror.
     - Your greatest risk may not be what went wrong last time.



CRYPTOGRAPHY RESEARCH

10 Suggestions

1. View security in economic terms.
2. Think about how risk is allocated.
3. Be humble and know your limits.
4. Make realistic assumptions.
5. **Minimize complexity.**
   - Isolate critical components.
   - Beware of complex interfaces.
   - Have the courage to resist adding features.

Unnecessary complexity
is a security flaw.

10 Suggestions

1. View security in economic terms.
2. Think about how risk is allocated.
3. Be humble and know your limits.
4. Make realistic assumptions.
5. Minimize complexity.
6. **Spend more on evaluation than design.**
   - Evaluations can only prove insecurity.
   - Make sure evaluators are skilled and objective.
     - Don't impose unreasonable restrictions.
     - Requires creativity, experience, attention to detail.

10 Suggestions

1. View security in economic terms.
2. Think about how risk is allocated.
3. Be humble and know your limits.
4. Make realistic assumptions.
5. Minimize complexity.
6. Spend more on evaluation than design.
7. **Be a skeptic.**
   - Assume systems are insecure unless you have evidence to the contrary.
     - Avoid anything undocumented or untestable.
   - Ask tough questions and demand responses.
     - Don't be impressed by the line: "We can't tell you for security reasons."

10 Suggestions

1. View security in economic terms.
2. Think about how risk is allocated.
3. Be humble and know your limits.
4. Make realistic assumptions.
5. Minimize complexity.
6. Spend more on evaluation than design.
7. Be a skeptic.
8. **Plan for trouble.**
   - What happens <u>after</u> a breach?
     - Will you know if there was a breach?
   - Keep good audit records.
   - Are "impossible" attacks really impossible?

Image courtesy NTSB.

10 Suggestions

1. View security in economic terms.
2. Think about how risk is allocated.
3. Be humble and know your limits.
4. Make realistic assumptions.
5. Minimize complexity.
6. Spend more on evaluation than design.
7. Be a skeptic.
8. Plan for trouble.
9. **Use both internal & external expertise.**
   - Risks are much higher if you rely only on just one.
   - Get multiple opinions, especially if you fear: piracy, fraud, or espionage.

10 Suggestions

1. Design for testability
2. Avoid complexity
3. Isolate complex components
4. Think about trust
5. Spend money rationally
6. Focus on interfaces
7. Assume that users are stupid and lazy
8. Don't re-invent the wheel
9. Avoid single points of failure
10. **Study all layers of the system**
    - Transistors up to business objectives

Unsolved Problems

- Security vendors focused on one model
  - Monolithic, centralized IT organization
  - Enterprise-centric (low-value networks)
- Ignore many real-world needs
  - Automatic per-port filtering
  - Dynamic routing on LAN
- Big customers need to demand solutions

Next: some unsolved problems that really bug me…

Unsolved: Application Filtering

- Progression of the firewall
  - IP address, port filtering
  - Stateful inspection
  - Application proxy
- Problem: applications are proliferating but firewalls stopped evolving at the HTTP layer
  - Everything runs on port 80
  - Protocols change greatly between versions
- Proposal: developer tools output protocol specification
  - Client and server software use spec to format messages
  - Firewall uses protocol spec to disallow improper messages

Unsolved: Product Certification Aging

- Certification specifies testing conditions
  - FIPS 140: certificate shows tested configuration
  - Common Criteria: profile lists requirements
- Problem: No one has that exact configuration and soon the vendor releases a new version
  - Windows NT level 2 cert (but without network)
  - Netscape level 2 cert (but you need a sticker)
- Proposal: Use a questionnaire
  - Vendors answer standardized questions about their system
  - Customers (along with evaluators) use the answers to identify application-specific questions

Evaluation Approach/Guidelines

- Things to look for when…
  - Working with a security evaluator
  - Informally evaluating preliminary designs

Algorithms

- Don't bother with internals of good algorithms
- Most proprietary ciphers are bad
  - Inexperienced designers often unhelpful, paranoid, overconfident
  - Cryptanalysis is tedious – other attacks may be easier
- Quick tests
  - Is the ciphertext obviously nonrandom?
    - Compressable? Biased frequency distribution?
  - Is the scheme published? Broken?
  - Who designed it?
- Public key schemes
  - Based on a (believed) "hard" problem?
  - Is security truly equivalent? Is problem truly hard?

Good Algorithms Used Badly

- Implementation errors
  - DES S tables, SSH CRC, PGPDisk CAST bug
  - Stream cipher key reuse
- Marginal key sizes (e.g., DES)
  - Time/memory tradeoff
  - Same ciphertext under many keys
  - Key data shared with other algorithms
- Poor key management
  - Nonrandom / insecure derivation?
  - Is the correct value used as the key?
  - Are keys stored/managed securely?
  - Careless reuse of keys
- Poor modes of operation
  - 3DES with internal CBC
  - DES MACs
  - ECB, CBC with adaptive chosen plaintext

CRYPTOGRAPHY RESEARCH

Protocol Analysis

- Many approaches (intuitive to formal)
- Simple approach:
  - On three big pieces of paper:
    - ① Chart the protocol flow
      - Include every message that can be sent
      - Error messages, optional messages, etc.
    - ② List what can be discovered about each cryptographic value
      - Each crypto step generally reveals something new
      - List everything (helps catch unintended interactions)
    - ③ Diagram the state machine of each participant
      - Include negotiated options, failure states, crypto, etc.
  - Reconcile possible end states against objectives.
    - Check for missing "free" functionality, excessive complexity

Common Protocol Weak Spots

- Algorithm negotiation
- Version negotiation (backward + forward)
- Man-in-the-middle
- Message replay (within a session, multiple sessions)
- Message forwarding & impersonation
  - A connects to B, who connects to C pretending to be A
- Certificate handling & validation (or lack thereof)
- Out-of-sequence messages
- Error handling reveals information
- Denial of service
- Timing analysis
- Excessive complexity or lack of defined state machine
- Improper or inadequate use of hash functions
- Inefficiencies (round trips)
- Redundant information
- Management/debug functions (code upgrades, etc.)

Trust Boundaries

- Designers should isolate keys & critical components
  - Putting all your eggs in one basket is actually good
    - Risking all your eggs in many baskets is dangerous.
    - Fewer critical components means they can be tested better.
- Most products have poorly-defined boundaries.
  - Are the perimeters (or contents) too complex?
    - Typical Windows PC is too complex to secure internally.
  - What can cross the perimeter?
    - APIs, network protocols, chip I/Fs, control/audit/backup data…
  - Analyze single points of failure (inside & outside) [next]

Focus on Single Points of Failure

Examples

ROM/E^2/BIOS contents	Hard disk controllers	Data backup & redundancy
Key storage & metadata	Revocation systems	Crypto algorithms
Threat detection systems	Engineering personnel	Drivers
Executable program storage	Compiler correctness	CPU execution correctness
Sandboxes	Non-standboxed code	Security protocols
Input validation routines	Passwords & login procedures	Tamper resistance
Software update procedures	Master keys & passwords	

Tools

- Gathering Information
  - Crypto toolkits (Crypto++, CryptoLib, etc.)
  - Statistical toolkits (custom)
  - Bignum libraries (NTL for Lattice Reduction)
  - Compiler, system analysis tools, debugger, decompiler
  - Network traffic recorder (tcpdump)
- Brute force / disaster recovery
  - FPGA board, CPU farm
  - Password dictionaries
  - Hard drive imaging tools
  - Password recovery tools/services
- Tamper Resistance
  - DPA workstation
  - Oscilloscope
  - X-ray, Probe station, microscopes, e-beam, FIB

Actual Risk vs. Perceived Risk

- Real Quote:
  - "Smart cards with triple DES are three times as secure as those using single DES."

- Everybody knows this is wrong…
  - Attackers almost never waste time and money on brute force
  - Even when it's easy, there are easier attacks

What doesn't work

- Committee designs
- Obscurity
  - Increases cost for initial attack, but not repeat attacks
  - Reduces relying party's ability to gain assurance
- Fixed certification standards
  - Standardized evaluations only catch standardized attacks…
- Requirements that often go against security
  - More speed, more features, less cost, less development time
- "Hail Mary" security evaluations
  - Too late: Need security by design

How government can help…

- Our challenges going forward:
  - Convincing vendors to spend on prevention
    - Profits from fraud lead to more crime
    - Not enough pressure on vendors from most of private sector
  - Evangelizing
    - Always looking for companies that need help
    - Consulting + licensing (DPA, tamper resistance, etc.)
- Help preserve freedom
  - Support legitimate research
  - Lead by example with higher assurance approach

Contact Information

For more information, or to discuss how Cryptography Research can help with a security problem:

Nate Lawson

nate@cryptography.com
Tel: 415.397.0123

www.cryptography.com

Questions?