CSci530: Security Systems
Lecture 1 – August 25, 2017 – OHE122
The Security Problem

Dr. Clifford Neuman
University of Southern California
Information Sciences Institute
Administration

• Class home page
  http://ccss.usc.edu/530
  – Preliminary Syllabus
  – Assigned Readings
  – Lecture notes
  – Assignments
Who gets in

• If you wish to enroll and do not have D clearance yet, send an email to CSci530@usc.edu with:
  – Your name
  – If you meet the prerequisites
  – A phone number
  – Request to received D clearance

• I will assess and approve if appropriate.
Structure of lecture

• Classes from 9:00 AM – 11:50 AM
  – 10 minute break halfway through
  – Before or after break, 10 minutes for discussion of two current events, to be led by students.
    ▪ Groups of 1 to 3. Send two sentences on topic by Wed, selected Thu AM then prepare slides.
Administration

• Lab Component (see http://ccss.usc.edu/530L)
  – 1 of the 4 units
  – Instructor is David Morgan
  – Instruction 4:30-5:20 Fridays in OHE 122
    ▪ WebCast via DEN
    ▪ Today’s Lab instruction is only a 30 minute introduction
  – Hands on sections, choose from several sessions
    ▪ Provides an opportunity to do hands on work in OHE 406 lab.
    ▪ Some labs will be done remotely using DETER
    ▪ Must sign up for your preference of session.
    ▪ Details will be provided this afternoon.
Administration

• Class e-mail: csci530@usc.edu
• Instructor
  – Dr. Clifford Neuman – bcn@isi.edu
    ▪ Please be sure to include CSCI530 in the subject of the message
  – Office hours Friday 4:30 to 5:30 in RTH-512
  – Contact info on class web page
• TA
  – Yatin Wadhawan - ywadhawa@usc.edu
  – Hours and contact information to be posted
Administration

• Grading Base Grade
  – Reading reports: 5%, 5%, 5%
  – Exams: 25%, 30%
  – Research paper 30%

• Supplemental grade (can raise or lower base):
  – Lab exercises Pass(hi,lo)/Fail (adj 15%)
  – Class participation
    • up to 10% bonus
Desire 2 Learn

• Using the DEN Desire to Learn system
  – Follow Link to Lectures and Discussion forum from ccss.usc.edu/530
  – Contact webclass@usc.edu if you have difficulty gaining access to the system.
Class Participation

• Class participation is important.
  – Ask and answering questions in class.
  – Ask, answer, participate on-line
  – Presentation of current events

• Bonus for class participation
  – If I don’t remember you from class, I look in the web discussion forum to check participation.
    ▪ Did you ask good questions.
    ▪ Did you provide good answers.
    ▪ Did you make good points in discussions.
Academic Integrity

• I take Academic Integrity Seriously
  – Every year I have too many cases of cheating
  – Last year I assigned multiple F’s for the class
  – On occasion, students have been dismissed from program
• What is and is not OK
  – I encourage you to work with others to learn the material
  – Do not to turn in the work of others
  – Do not give others your work to use as their own
  – Do not plagiarize from others (published or not)
  – Do not try to deceive the instructors
• See section on web site and assignments
  – More guidelines on academic integrity
  – Links to university resources
  – Don’t just assume you know what is acceptable.
The Three Aspects of Security

• Confidentiality
  – Keep data out of the wrong hand

• Integrity
  – Keep data from being modified

• Availability
  – Keep the system running and reachable
Policy v. Mechanism

• Security policy defines what is and is not allowed
  – What confidentiality, integrity, and availability actually mean

• Security mechanism is a method or tool for enforcing security policy
  – Prevention
  – Detection
  – Reaction

• Among mechanisms are:
  – Mechanisms enforce policy.
  – Mechanisms may solve intermediate problems.
    ▪ Authentication, Audit
    ▪ Containment
Trusted vs. Trustworthy

• We trust our computers
  – We depend upon them.
  – We are vulnerable to breaches of security.
• Our computer systems today are not worthy of trust.
  – We have buggy software
  – We configure the systems incorrectly
  – Our user interfaces are ambiguous regarding the parts of the system with which we communicate.
• Trust is for a purpose and an assumed environment
**Terminology**

Trusted – Parts of a system that we depend upon for the proper enforcement of policies, whether or not the code is free of vulnerabilities (almost all systems have vulnerabilities). - as compared with

Trustworthy – our belief that a system is free of vulnerabilities that could result in the violation the relevant security policies.

Accreditation – A statement by a third party that a system or software has been found to be trustworthy with respect to a particular set of policies and for a particular operational environment.
Important Considerations

• Risk analysis and Risk Management
  – Impact of loss of data.
  – Impact of disclosure.
  – Legislation may play a role.

• Human factors
  – The weakest link
In The Shoes of an Attacker

• Motivation
  – Bragging Rights
  – Revenge / to inflict damage
  – Terrorism and Extortion
  – Financial / Criminal enterprises
  – Nation State Objectives

• Risk to the attacker
  – Can play a defensive role
  – Including effective attribution
Users of Published Attack Tools

• Attacker has specific tools
  – Casts the tool widely to see what can be caught.
  – Sometimes described as script-kiddies
    ▪ Gets them into systems or with specific vulnerabilities
    ▪ Gets them account access to susceptible employees
  – The gather what they find, exfiltrate or modify, and stop there

• Strong security posture is effective
  – Sound security practices
  – Systems up to date
  – Least privilege
Opportunistic or Bottom Up Adversaries

• Looks for the weak link
  – Uses tools to scan for vulnerabilities
  – Once in, repeats the process
    ▪ This time starting with elevated access because of the system or user ID already compromised.

• Your containment architecture is critical against such adversaries.
  – You need to be aware of the paths that might be followed to reach sensitive data.
Goal Oriented Top Down Adversaries

- Learns about your organization and system
  - Goal is to compromise some component of your system or access specific data.
  - Learns precursor activities that must be achieved to meet that goal.
  - Often applies APT – Advanced Persistent Threat tactics
  - Will wait for threat vector to propagate

- Defenses require all of:
  - Strong security posture
  - Training of privileged employees
  - Containment Architecture
  - Strong defenses to subversion.
Economics of Malicious Code

• Controlled machines for sale
• “Protection” or “recovery” for sale
• Attack software for sale
• Stolen data for sale
• Intermediaries used to convert online balances to cash.
  – These are the pawns and the ones that are most easily caught
Terminology (around attacks)

Vulnerability – A weakness in a system, program, procedure, or configuration that could allow an adversary to violate the intended policies of a system.

Threat – Tools or knowledge (capabilities) that care capable of exploiting a vulnerability to violate the intended policies of a system.

Attack – An attempt to exploit a vulnerability to violate the intended policies of a system.

Compromise or intrusion – The successful actions that violate the intended polices of a system.
Incidents and Breaches

Penetration – A successful attack (intrusion) that exploits a vulnerability in the code base of a system or its configuration. The result will often be to install a subversion.

Denial of Service – An attack that prevents authorized access to a resource, by destroying a target or overwhelming it with undesired requests.

Subversion - An intentional change to the code base or configuration of a system that alters the proper enforcement of policy. This includes the installation of backdoors and other control channels in violation of the policy relevant to the system.

Subversion vectors – the methods by which subversions are introduced into a system. Often the vectors take the form of malicious code.
More Terminology

Secure – A system is secure if it correctly enforces a correctly stated policy for a system. A system can only be secure with respect to a particular set of policies and under a set of stated assumptions. There is no system that is absolutely secure.

Trusted Computing Base – That part of a system which if compromised affects the security of the entire system. One often unstated assumption made with respect to a secure system is that the TCB is correctly implemented and has not been compromised.

Attack Surface – The accumulation of all parts of a system that are exposed to an adversary against which the adversary can try to find and exploit a vulnerability that will render the system insecure (i.e. violate the security policies of the system).
What is security

• System, Network, Data
  – What do we want to protect
  – From what perspective
• How to evaluate
  – Balance cost to protect against cost of compromise
  – Balance costs to compromise with risk and benefit to attacker.
• Security vs. Risk Management
  – Prevent successful attacks vs. mitigate the consequences.
• It’s not all technical
Security and Society

• Does society set incentives for security.
  – OK for criminal aspects of security.
  – Not good in assessing responsibility for allowing attacks.
  – Privacy rules are a mess.
  – Incentives do not capture gray area
    ▪ Spam and spyware
    ▪ Tragedy of the commons
Why we aren’t secure

- Buggy code
- Protocols design failures
- Weak crypto
- Social engineering
- Insider threats
- Poor configuration
- Incorrect policy specification
- Stolen keys or identities
- Denial of service
What do we want from security

• Confidentiality
  – Prevent unauthorized disclosure
• Integrity
  – Authenticity of document
  – That it hasn’t changed
• Availability
  – That the system continues to operate
  – That the system and data is reachable and readable.
• Enforcement of policies
  – Privacy
  – Accountability and audit
  – Payment
The role of policy in security architecture

Policy – Defines what is allowed and how the system and security mechanisms should act.

Enforced By

Mechanism – Provides protection interprets/evaluates (firewalls, ID, access control, confidentiality, integrity)

Implemented as:

Software: which must be implemented correctly and according to sound software engineering principles.
Security Mechanisms

- Encryption
- Checksums
- Key management
- Authentication
- Authorization
- Accounting
- Firewalls
- Virtual Private Nets
- Intrusion detection
- Intrusion response
- Development tools
- Virus Scanners
- Policy managers
- Trusted hardware
Today’s security deployment

• Most deployment of security services today handles the easy stuff, implementing security at a single point in the network, or at a single layer in the protocol stack:
  – Firewalls, VPN’s
  – IPSec
  – SSL
  – Virus scanners
  – Intrusion detection
A more difficult problem

• Unfortunately, security isn’t that easy. It must be better integrated with the application.
  – At the level at which it must ultimately be specified, security policies pertain to application level objects, and identify application level entities (users).
Security Systems vs Systems Security

Integration of dynamic security services creates feedback path enabling effective response to attacks

- **Firewalls**
- **Web Servers**
- **Databases**
- **IPSec**
- ...
Loosely Managed Systems

• Security is made even more difficult to implement since today’s systems lack a central point of control.
  – Home machines unmanaged
  – Networks managed by different organizations.
  – A single function touches machines managed by different parties.
    ▪ Clouds
  – Who is in control?
Who is in Control

- The Intruder?
- The Government?
- Your employer?
- Those with whom you do business?
- Infrastructure (cloud) providers?
- Ultimately, it must be you who takes control, but today’s systems don’t take that view.
  - You must balance conflicting interests and control.
• End of Lecture 1

• Following slides are start of lecture 2
Announcements

• I will hold half of my office hours from 12:15 to 12:45 today in RTH512.
  – The second half will be from 4:30 to 5PM in RTH512.
  – This change is in recognition that the original office hours conflicted with our lab instructions.
Announcements

• Assignment 1 has been posted on the class webpage at:
  – http://ccss.usc.edu/csci530
  – Assignment is due on 13 September 2017
Cryptography and Security

- Cryptography underlies many fundamental security services
  - Confidentiality
  - Data integrity
  - Authentication
- It is a basic foundation of much of security.
A Brief History

• Steganography: “covered writing”
  – Demaratus and wax tablets
  – German microdots (WWII)
  – Flaw: Discovery yields knowledge
    – Confidentiality through obscurity
• Cryptography: “secret writing”
  – TASOIINRNPSTO and TVCTUJUVUJPO
Encryption used to scramble data

**PLAINTEXT**

![](image1.png)

**CIPHERTEXT**

![](image2.png)

**PLAINTEXT**

*Encryption*:

![](image3.png)

*Decryption*:

![](image4.png)
The Basics of Cryptography

- Two basic types of cryptography
  - TASONO PINSTIR
    - Message broken up into units
    - Units permuted in a seemingly random but reversible manner
    - Difficult to make it easily reversible only by intended receiver
    - Exhibits same first-order statistics
The Basics of Cryptography

• Two basic types of cryptography
  – TRANSPOSITION (TASONOPINSTIR)
    • Message broken up into units
    • Units permuted in a seemingly random but reversible manner
    • Difficult to make it easily reversible only by intended receiver
    • Exhibits same first-order statistics
The Basics (continued)

• Two basic types of cryptography (cont)
  – TVCTUJUVUJPO
    ▪ Message broken up into units
    ▪ Units mapped into ciphertext
      – Ex: Caesar cipher
    ▪ First-order statistics are isomorphic in simplest cases
    ▪ Predominant form of encryption
The Basics (continued)

• Two basic types of cryptography (cont)
  – Substitution (TVCTUJUVUJPO)
    ▪ Message broken up into units
    ▪ Units mapped into ciphertext
      – Ex: Caesar cipher
    ▪ First-order statistics are isomorphic in simplest cases
    ▪ Predominant form of encryption
How Much Security?

• Mono-alphabetic substitution cipher
  – Permutation on message units—letters
    ▪ 26! different permutations
    ▪ Each permutation considered a *key*
  – Key space contains 26! = 4x10^{26} keys
    ▪ Equals number of atoms in gallon H_2O
    ▪ Equivalent to a 88-bit key
How Much Security?

• So why not use substitution ciphers?
  – Hard to remember 26-letter keys
    ▪ But we can restrict ourselves to shorter keys
      ▪ Ex: JULISCAERBDFGHK, etc
  – Remember: first-order statistics are isomorphic
    ▪ Vulnerable to simple cryptanalysis
    ▪ Hard-to-read fonts for crypto?!
Crypto-analytic Attacks

• Classified as:
  – Cipher text only
    ▪ Adversary sees only the ciphertext
  – Known plain text
    ▪ May know some corresponding plaintext (e.g. Login:)
  – Chosen plaintext
    ▪ Can ask to have text encrypted
Substitution Ciphers

- Two basic types
  - Symmetric-key (conventional)
    ▪ Single key used for both encryption and decryption
    ▪ Keys are typically short, because key space is densely filled
  - Ex: AES, DES, 3DES, RC4, Blowfish, IDEA, etc
Substitution Ciphers

• Two basic types (cont)
  – Public-key (asymmetric)
    ▪ Two keys: one for encryption, one for decryption
    ▪ Keys are typically long, because key space is sparsely filled
    ▪ Ex: RSA, El Gamal, DSA, etc
One Time Pads

• For confidentiality, One Time Pad provably secure.
  – Generate truly random key stream size of data to be encrypted.
  – Encrypt: Xor plaintext with the keystream.
  – Decrypt: Xor again with keystream.

• Weak for integrity
  – 1 bit changed in cipher text causes corresponding bit to flip in plaintext.

• Key size makes key management difficult
  – If key reused, the cipher is broken.
  – If key pseudorandom, no longer provably secure
  – Beware of claims of small keys but as secure as one time pad – such claims are wrong.
Block vs. Stream: Block

- Block ciphers encrypt message in units called blocks
  - E.g. DES: 8-byte key (56 key bits), 8-byte block
  - AES (discussed later) is also a block cipher.
  - Larger blocks make simple cryptanalysis useless (at least for short messages)
    - Not enough samples for valid statistics
    - 8 byte blocks common
    - But can still tell if something is the same.
Key and Block Size

- Do larger keys make sense for an 8-byte block?
  - 3DES: Key is 112 or 168 bits, but block is still 8 bytes long (64 bits)
  - Key space is larger than block space
  - But how large is permutation space?
More on DES Internals

• More details on the internal operation of DES is covered in CSci531 and Inf521
• But we cover Modes of Operation in this lecture since these modes are important to apply DES, and the same modes can be used for other block ciphers.
Block vs. Stream: Stream

- Stream ciphers encrypt a bit, byte, or block at a time, but the transformation that is performed on a bit, byte, or block varies depending on position in the input stream and possibly the earlier blocks in the stream.
  - Identical plaintext block will yield a different cipher text block.
  - Makes cryptanalysis more difficult.
  - DES modes CBC, CFB, and OFB modes (discussed next) create stream ciphers from DES, which is a block cipher.
  - Similar modes available for AES.
DES Modes of Operation – Electronic Code Book (ECB)

- Each block encrypted in isolation
- Vulnerable to block replay
DES Modes of Operation – Cipher Block Chaining (CBC)

Encrypt:

- Each plaintext block XOR'd with previous ciphertext
- Easily incorporated into decryption
- What if prefix is always the same? IV!

Decrypt:
DES Modes of Operation – Cipher Feedback Mode (CFB)

Encrypt:

- For encrypting character-at-a-time (or less)
- Chains as in CBC
- Also needs an IV – Must be Unique – Why?
DES Modes of Operation – Output Feedback Mode (OFB)

Encrypt:

Decrypt:

—Like CFB, but neither ciphertext nor plaintext is fed back to the input of the block encryption.
Variants and Applications

• 3DES: Encrypt using DES 3x
  – Two and three-key types
  – Inner and outer-CBC modes
• Crypt: Unix hash function for passwords
  – Uses variable expansion permutations
• DES with key-dependent S-boxes
  – Harder to analyze
3DES Using Two Keys

- Can use K1,K2,K3, or K1,K2,K1, or K1,K1,K1

- Figure courtesy William Cheng
3DES Outer CBC

CBC on the outside:

- Figure courtesy William Cheng
3DES Inner CBC

- Inner is more efficient, but less secure
  - More efficient due to ability to pipeline implementation
  - Weaker for many kinds of attacks

Figure courtesy William Cheng
Why not Two Round

- Meet in middle attack makes it not much better than single DES.

  - Figure courtesy William Cheng
Certification of DES

• Had to be recertified every ~5 years
  – 1983: Recertified routinely
  – 1987: Recertified after NSA tried to promote secret replacement algorithms
    ▪ Withdrawal would mean lack of protection
    ▪ Lots of systems then using DES
  – 1993: Recertified after continued lack of alternative
Enter AES

• 1998: NIST finally refuses to recertify DES
  – 1997: Call for candidates for Advanced Encryption Standard (AES)
  – Fifteen candidates whittled down to five
  – Criteria: Security, but also efficiency
    ▪ Compare Rijndael with Serpent
    ▪ 9/11/13 rounds vs 32 (breakable at 7)
  – 2000: Rijndael selected as AES
Structure of Rijndael

• Unlike DES, operates on whole bytes for efficiency of software implementations
• Key sizes: 128/192/256 bits
• Variable rounds: 9/11/13 rounds
• More details on structure in the applied cryptography class.
Security of Rijndael

• Key size is enough
• Immune to linear or differential analysis
• But Rijndael is a very structured cipher
• Attack on Rijndael’s algebraic structure
  – Breaking can be modeled as equations
Impact of Attacks on Rijndael

• Currently of theoretical interest only
  – Reduces complexity of attack to about $2^{100}$
  – Also applicable to Serpent
• Still, uncomfortably close to feasibility
  – DES is already insecure against brute force
  – Schneier (somewhat arbitrarily) sets limit at $2^{80}$
• Certainly usable pending further results
Administration

- Class home page
  http://ccss.usc.edu/530
- Assignment 1 is due Wednesday
- Paper proposals due 28 September
NSA 4th Annual Codebreaker Challenge

This Fall, NSA is launching its 4th annual Codebreaker Challenge. It is a hands-on software reverse engineering challenge where students work to complete mission-focused objectives to push their school to the top of the competition leaderboard.

The theme for this year's challenge is "countering Improvised Explosive Devices (IEDs)". Students are given six tasks of increasing difficulty that culminate in developing the capability to permanently disable fictional IED software in a fictional scenario. Feedback from previous challenges indicated students learned a great deal from participating.

Here are the pertinent details:

• The challenge will be hosted at [https://codebreaker.Ltsnet.net](https://codebreaker.Ltsnet.net)
• The challenge begins 9 September 9 pm ET & ends 31 December 2016
• Students should register on the site using their .edu email addresses
• Links to software reverse engineering lectures and other educational material can be found on the site.
Public Key Cryptography

- aka asymmetric cryptography
- Based on some NP-complete problem
  - Unique factorization
  - Discrete logarithms
    - For any $b, n, y$: Find $x$ such that $b^x \mod n = y$
- Modular arithmetic produces folding
A Short Note on Primes

• Why are public keys (and private keys) so large?
• What is the probability that some large number p is prime?
  – About 1 in 1/\ln(p)
  – When p \sim 2^{512}, equals about 1 in 355
    • About 1 in 355^2 numbers \sim 2^{1024} is product of two primes (and therefore valid RSA modulo)
RSA

- Rivest, Shamir, Adleman
- Generate two primes: p, q
  - Let n = pq
  - Choose e, a small number, relatively prime to (p-1)(q-1)
  - Choose d such that \( ed = 1 \mod (p-1)(q-1) \)
- Then, \( c = m^e \mod n \) and \( m = c^d \mod n \)
An Example

• Let $p = 5$, $q = 11$, $e = 3$
  – Then $n = 55$
  – $d = 27$, since $(3)(27) \mod 40 = 1$
• If $m = 7$, then $c = 7^3 \mod 55 = 343$
  mod $55 = 13$
• Then $m$ should $= 13^{27} \mod 55$
An Example

- **Computing** $13^{27}$ mod 55
  - $13^1$ mod 55 = 13, $13^2$ mod 55 = 4,
    $13^4$ mod 55 = 16, $13^8$ mod 55 = 36,
    $13^{16}$ mod 55 = 31
  - $13^{27}$ mod 55 = $(13)(4)(36)(31)$ mod 55 = $(1872$ mod 55$(31)$ mod 55 = 62 mod 55 = 7 (check)
Other Public Cryptosystems

- ElGamal (signature, encryption)
  - Choose a prime $p$, a generator $< p$
  - Choose a random number $x < p$
  - Public key is $g$, $p$, and $y = g^x \mod p$
  - Private key is $x$; to obtain from public key requires extracting discrete log
  - Mostly used for signatures
Other Public Cryptosystems

• Elliptic curve cryptosystems
  – $y^2 = x^3 + ax^2 + bx + c$
  – Continuous elliptic curves used in FLT proof
  – Discrete elliptic curves used to implement existing public-key systems
    ▪ Allow for shorter keys and greater efficiency
Importance of ECC

- There has been rapid progress in cryptanalysis of RSA and Diffie-Hellman public key systems.

- ECC is based on different mathematics, which has been shown to be NP complete.
Current Event
The Threat of GPS Spoofing

Joshua Insel
U.S. Navy Collisions

- Two collisions involving U.S. Navy ships have recently occurred.
  - June 17 – USS Fitzgerald was hit by a cargo ship.
  - August 21 – USS John S. McCain collided with an oil tanker.
- Such incidents are rare, but were most likely accidents.
- Some cybersecurity experts suggest the possibility these were caused by either:
  - Hacking into navigational systems
  - GPS spoofing
GPS Spoofing

- Done with GPS signal simulators:
  - Used for testing GPS devices in factories.
  - Can be purchased very cheaply.
- In 2013, Mark Humphreys (University of Texas at Austin) and a team of graduate students changed the course of an $80 million yacht using such equipment.
- GPS spoofing could be used for various malicious purposes.
- No commercial GPS devices currently offer reliable protection against spoofing.
- U.S. Navy uses encrypted GPS signals, which would have to be recorded and replayed for a spoofing attack.
A Solution through Cryptography

- Europe's new global navigation satellite system (GNSS): Galileo
- University of Leuven researchers Ashur and Rijmen (designer of AES) developed the TESLA signature for this system.
- An authentication protocol which creates a cryptographic “signature”.
- This signature will be sent with location data, and used to verify authenticity by GPS receivers.
- Symmetric cryptography is used rather than public-key cryptography, due to low bandwidth.
CSci530: Security Systems
Lecture 3 – September 8, 2017
Hash Functions, then
Key Management

Dr. Clifford Neuman
University of Southern California
Information Sciences Institute
Announcements

• I will hold half of my office hours from 12:15 to 12:45 today in RTH512.
  – The second half will be from 4:30 to 5PM in RTH512.
  – This change is in recognition that the original office hours conflicted with our lab instructions.
Announcements

• Assignment 1 has been posted on the class webpage at:
  – http://ccss.usc.edu/csci530
  – Assignment is due on 13 September 2017
Hash Functions

• Given m, compute H(m)
• Should be…
  – Efficient: H() easy to compute
  – One-way: Given H(m), hard to find m’ such that H(m’) = H(m)
  – Collision-resistant: Hard to find m and m’ such that H(m’) = H(m)
Digital Signatures

• Provides data integrity
  – Can it be done with symmetric systems?
    ▪ Verification requires shared key
    ▪ Doesn’t provide non-repudiation

• Need proof of provenance
  – Hash the data, encrypt with private key
  – Verification uses public key to decrypt hash
  – Provides “non-repudiation”
    ▪ But what does non-repudiation really mean?
Digital Signatures

- RSA can be used
- DSA: Digital Signature Algorithm
  - Variant of ElGamal signature
  - Adopted as part of DSS by NIST in 1994
  - Slower than RSA (but likely unimportant)
  - NSA had a hand in its design (?!)
  - Key size ranges from 512 to 1024 bits
  - Royalty-free
Key Exchange

- Diffie-Hellman key exchange
  - Choose large prime $n$, and generator $g$
    - For any $b$ in $(1, n-1)$, there exists an $a$ such that $g^a = b$
  - Alice, Bob select secret values $x$, $y$, resp
  - Alice sends $X = g^x \mod n$
  - Bob sends $Y = g^y \mod n$
  - Both compute $g^{xy} \mod n$, a shared secret
    - Can be used as keying material
CSci530: Security Systems
Lecture 3 – September 8, 2017
Key Management

Dr. Clifford Neuman
University of Southern California
Information Sciences Institute
Cryptography in Use

• Provides foundation for security services
  – Provides confidentiality
  – Validates integrity
  – Provides data origin authentication
  – If we know the key

• Where does the key come from
  – Straightforward plan
    ▪ One side generates key
    ▪ Transmits key to other side
    ▪ But how?
Key Management

• Key management is where much security weakness lies
  – Choosing keys
  – Storing keys
  – Communicating keys
What to do with keys

• Practical issues
  – How to carry them
    ▪ Passwords vs. disks vs. smartcards
  – Where do they stay, where do they go
  – How many do you have
  – How do you get them to begin with.
Relevant News:
It is Easier to do it wrong

Internet of Sins: Million more devices sharing known private keys for HTTPS, SSH admin – The Register September 7, 2016.

Millions of internet-facing devices – from home broadband routers to industrial equipment – are still sharing well-known private keys for encrypting their communications.

This is according to research from SEC Consult, which said in a follow-up to its 2015 study on security in embedded systems that the practice of reusing widely known secrets is continuing unabated.

Devices and gadgets are still sharing private keys for their built-in HTTPS and SSH servers, basically. It is not difficult to extract these keys from the gizmos and use them to eavesdrop on encrypted connections and interfere with the equipment: imagine intercepting a connection to a web-based control panel, decrypting it, and altering the configuration settings on the fly. And because so many models and products are using the same keys, it's possible to attack thousands of boxes at once.
Bootstrapping Security

• Exchange the key in person
  – Can exchange key before it is needed.
  – Could be a password.
• Hide the key in something else
  – Steganography, fairly weak
• Armored courier
  – If all else fails
• Send key over the net encrypted
  – But, using what key (bootstrap)
Diffie-Hellman Key Exchange (1)

- Choose large prime n, and generator g
  - For any b in (1, n-1), there exists an a such that \( g^a = b \). This means that every number mod p can be written as a power of g (mod p).
    - To find such a g, pick the p such that \( p = 2q + 1 \) where q is also prime.
    - For such choices of p, half the numbers will be generators, and you can test if a candidate g is a generator by testing whether \( g^q \pmod{n} \) is equal to n-1.
Diffie-Hellman Key Exchange (2)

- Alice, Bob select secret values $x, y$
- Alice sends $X = g^x \mod n$
- Bob sends $Y = g^y \mod n$
- Both compute $g^{xy} \mod n$, a shared secret
  - Can be used as keying material
Man in the middle of DH

- DH provides key exchange, but not authentication
  - You don’t really know you have a secure channel
- Man in the middle
  - You exchange a key with eavesdropper, who exchanges key with the person you think you are talking to.
  - Eavesdropper relays all messages, but observes or changes them in transit.
- Solutions:
  - Published public values
  - Authenticated DH (Sign or encrypt DH value)
  - Encrypt the DH exchange
  - Subsequently send hash of DH value, with secret
Two Cases so Far

• Can exchange a key with anyone, but you don’t know who you are talking with.
• Can exchange keys with known parties in advance, but are limited to communication with just those parties.
Peer-to-Peer Key Distribution

- Technically easy
  - Distribute keys in person
- But it doesn’t scale
  - Hundreds of servers…
  - Times thousands of users…
  - Yields ~ million keys
Incremental Key Distribution

• Build toward Needham-Schroeder and Kerberos mechanisms
• Key-distribution tied to authentication.
  – If you know who you share a key with, authentication is easy.
  – You want to know who has the key, not just that anyone has it.
But first a look forward – Encryption Based Authentication

• Proving knowledge of encryption key
  – Nonce = Non repeating value

{Nonce or timestamp}\text{K}_{\text{CS}}

But where does \text{K}_{\text{CS}} come from?
That is the subject of Key Distribution/Management
KDC Based Key Distribution

As used in both Needham Schroeder and Kerberos we will use Kerberos terminology

• User sends request to KDC: \{s\}
• KDC generates a random key: \(K_{c,s}\)
  – Encrypted twice: \{\{K_{c,s}\}K_c, K_{c,s}\}K_s\)
  – \{K_{c,s}\}K_s called ticket
  – Ticket plus \(K_{c,s}\) called credentials
  – Ticket is opaque and forwarded with application request
• No keys ever traverse net in the clear
Kerberos or Needham Schroeder

Third-party authentication service
- Distributes session keys for authentication, confidentiality, and integrity

Simplified

1. $s,n$
2. $\{K_{c,s} S, n\} K_c, \{K_{c,s} C\} K_s$
3-5. $\{\text{Nonce or } T\} K_{cs}$
Problem

• User now trusts credentials
• But can server trust user?
• How can server tell this isn’t a replay?
• Legitimate user makes electronic payment to attacker; e.g. attacker replays message to get paid multiple times
  – Requires no knowledge of session key
Solution

• Add challenge-response
  – Server generates second random nonce
  – Sends to client, encrypted in session key
  – Client must decrypt, decrement, encrypt

• Effective, but adds second round of messages
  – Can use timestamps as nonces
    ▪ But must remember what seen
Problem

• What happens if attacker does get session key?
  – Answer: Can reuse old session key to answer challenge-response, generate new requests, etc
  – Think of this like finding a cookie
Solution

• Replace (or supplement) nonce in request/reply with timestamp [Denning, Sacco]
  – \{K_{c,s}, s, n, t\}K_c and \{K_{c,s}, c, t\}K_s, resp
  – Also send \{t\}K_{c,s} as authenticator
    • Prevents replay without employing second round of messages as in challenge-response
    • Lifetime of ticket
Problem

• How to reduce vulnerability of client’s primary key: Kc
Solution

• Introduce Ticket Granting Server (TGS)
  – Daily ticket plus session keys
• TGS+AS = KDC
  – This is modified Needham-Schroeder
  – Basis for Kerberos
• Pre-authentication
• Note: not a full solution
  – Makes it slightly harder for adversary.
Kerberos

Third-party authentication service
- Distributes session keys for authentication, confidentiality, and integrity

1. Req
2. T+{Reply}Kc
3. TgsReq
4. Ts+{Reply}Kt
5. Ts + {ts}Kcs
Key Distribution linked to Authentication

• It's all about knowing who has the keys.
• Authentication is really a topic for next lecture, but the tight linkage with key management is the reason that we covered the Kerberos authentication system in the past few slides.
Public Key Distribution

• Public key can be public!
  – How does either side know who and what the key is for? Private agreement? (Not scalable.)
• Does this solve key distribution problem?
  – No – while confidentiality is not required, integrity is.
• Still need trusted third party
Key Management

- Key management is where much security weakness lies
  - Choosing keys
  - Storing keys
  - Communicating keys
Certification Infrastructures

- Public keys represented by certificates
- Certificates signed by other certificates
  - User delegates trust to trusted certificates
  - Certificate chains transfer trust up several links

Do you trust a certificate signed by Amazon?
Examples

- **PGP**
  - “Web of Trust”
  - Can model as connected digraph of signers

- **X.500**
  - Hierarchical model: tree (or DAG?)
  - (But X.509 certificates use ASN.1!)
Examples

• SSH
  – User keys out of band exchange.
  – Weak assurance of server keys.
    ▪ Was the same host you spoke with last time.
  – Discussion of benefits

• SET
  – Hierarchical
  – Multiple roots
  – Key splitting
Key Distribution

• Conventional cryptography
  – Single key shared by both parties
• Public Key cryptography
  – Public key published to the world
  – Private key known only by owner
• Third party certifies or distributes keys
  – Certification infrastructure
  – Authentication
Practical use of keys

- Email (PEM or S/MIME or PGP)
  - Hashes and message keys to be distributed and signed.
- Conferencing
  - Group key management (discussed later)
- Authentication (next lecture)
- SSL
  - And other “real time” protocols
  - Key establishment
Recovery from exposed keys

- Revocation lists (CRL’s)
  - Long lists
  - Hard to propagate
- Lifetime / Expiration
  - Short life allows assurance of validity at time of issue.
- Realtime validation
  - Online Certificate Status Protocol (OCSP)
- What about existing messages?
Key Management Overview

- Key size vs. data size
  - Affects security and usability
- Reuse of keys
  - Multiple users, multiple messages
- Initial exchange
  - The bootstrap/registration problem
  - Confidentiality vs. authentication
Key Management Review

• KDC’s
  – Generate and distribute keys
  – Bind names to shared keys
CSci530: Security Systems
Lectures 4 & 5 – September 15 & 22, 2017
Key Management Continued
then Authentication

Dr. Clifford Neuman
University of Southern California
Information Sciences Institute
Key Management Overview

• Who needs strong secrets anyway
  – Users?
  – Servers?
  – The Security System?
  – Software?
  – End Systems?

• Secret vs. Public
Group Key Management

- Group key vs. Individual key
  - Identifies member of groups vs. which member of group
  - PK slower but allows multiple verification of individuals
Group Key Management Issues

• Revoking access
  – Change messages, keys, redistribute

• Joining and leaving groups
  – Does one see old message on join
  – How to revoke access

• Performance issues
  – Hierarchy to reduce number of envelopes for very large systems
  – Hot research topic
Group Key Management Approaches

- Centralized
  - Single entity issues keys
  - Optimization to reduce traffic for large groups
  - May utilize application specific knowledges
- Decentralized
  - Employs sub managers
- Distributed
  - Members do key generation
  - May involve group contributions
Look Forward

Security Architectures

• DSSA
  – Delegation is the important issue
    ▪ Workstation can act as user
    ▪ Software can act as workstation
      – if given key
    ▪ Software can act as developer
      – if checksum validated
  – Complete chain needed to assume authority
  – Roles provide limits on authority – new sub-principal
CSci530: Security Systems  
Lectures 4&5 – September 15&22, 2017  
Authentication

Dr. Clifford Neuman  
University of Southern California  
Information Sciences Institute
Identification vs. Authentication

Identification

Associating an identity with an individual, process, or request

Authentication

– Verifying a claimed identity
Basis for Authentication

Ideally

Who you are

Practically

Something you know

Something you have

Something about you

(Sometimes mistakenly called things you are)
Something you know

Password or Algorithm
  e.g. encryption key derived from password

Issues
  Someone else may learn it
  Find it, sniff it, trick you into providing it
  Other party must know how to check
  You must remember it
  How stored and checked by verifier
Examples of Password Systems

Verifier knows password

Encrypted Password

One way encryption

Third Party Validation
Attacks on Passwords

Brute force
Dictionary
Pre-computed Dictionary
Guessing
Finding elsewhere
General Problems with Password

Space from which passwords Chosen
  Too many passwords
    And what it leads to
  Too few passwords
    i.e. password re-use
  That you need to present the password to use it
    Compromise of verifier affects password.
What makes for a good password

How some systems define good passwords:
MickeyMinniePlutoHueyLouieDewey
DonaldGoofyWashington

Other attacks on passwords
Social Engineering attacks
Including Phishing
Recent News

Phishing is now (and has been) an automated process.

Discussion:
Why we need to move away from passwords.
What are the effective alternatives.
Something you Have

Cards
  Mag stripe (= password)
  Smart card, USB key
  Time varying password

Issues
  How to validate
  How to read (i.e. infrastructure)
Case Study – RSA SecureID

Claimed - Something You Have
Reduced to something they know

How it works:
  Seed
  Synchronization

Compromises:
  RSA Break-in
  Or man in the middle
Something about you

Biometrics

Measures some physical attribute

- Iris scan
- Fingerprint
- Picture
- Voice

Issues

How to prevent spoofing

Suited when biometric device is trusted, not suited otherwise
Other forms of authentication

IP Address
Caller ID (or call back)
   Now “phone factor” (probably tm)
Past transaction information
   (second example of something you know)
“Enrollment”

How to initially exchange the secret.

In person enrollment
Information known in advance
Third party verification
Mail or email verification
SAN FRANCISCO — Cybersecurity professionals who track down bugs discovered, created a fix for, and told the industry about the vulnerability that allowed attackers into the Equifax network two months before the company was hit by hackers.

"The Equifax data compromise was due to (Equifax') failure to install the security updates provided in a timely manner," The Apache Foundation, which oversees the widely-used open source software, said in a statement Thursday.

Equifax told USA TODAY late Wednesday that the criminals who potentially gained access to the personal data of up to 143 million Americans had exploited a website application vulnerability known as Apache Struts CVE-2017-5638.
Multi-factor authentication

Require at least two of the classes above.

- e.g. Smart card plus PIN
- RSA SecurID plus password (AOL)
- Biometric and password

Issues

- Better than one factor
- Be careful about how the second factor is validated. E.g. on card, or on remote system.
Single Sign On

“Users should log in once
And have access to everything”
Many systems store password lists
Which are easily stolen
Better is encryption based credentials
Usable with multiple verifiers
Interoperability is complicating factor.
Encryption Based Authentication

• Proving knowledge of encryption key
  – Nonce = Non repeating value

\[\{\text{Nonce or timestamp}\}K_{cs}\]

\[\text{C} \rightarrow \text{S}\]
Authentication w/ Conventional Crypto

- Kerberos or Needham Schroeder

Diagram:
- 1: \text{KDC} \rightarrow \text{C}
- 2: \text{KDC} \rightarrow \text{S}
- 3, 4, 5: \text{C} \leftrightarrow \text{S}
Authentication w/ PK Crypto

- Based on public key certificates
Public Key Cryptography (revisited)

• Key Distribution
  – Confidentiality not needed for public key
  – Solves $n^2$ problem

• Performance
  – Slower than conventional cryptography
  – Implementations use for key distribution, then use conventional crypto for data encryption

• Trusted third party still needed
  – To certify public key
  – To manage revocation
  – In some cases, third party may be off-line
Certificate-Based Authentication

Certification authorities issue signed certificates

- Banks, companies, & organizations like Verisign act as CA’s
- Certificates bind a public key to the name of a user
- Public key of CA certified by higher-level CA’s
- Root CA public keys configured in browsers & other software
- Certificates provide key distribution
Certificate-Based Authentication (2)

Authentication steps

- Verifier provides nonce, or a timestamp is used instead.
- Principal selects session key and sends it to verifier with nonce, encrypted with principal’s private key and verifier’s public key, and possibly with principal’s certificate.
- Verifier checks signature on nonce, and validates certificate.
Secure Sockets Layer (and TLS)

Encryption support provided between Browser and web server - below HTTP layer

Client checks server certificate
   Works as long as client starts with the correct URL

Key distribution supported through cert steps
Authentication provided by verify steps
Trust models for certification

• X.509 Hierarchical
  – Single root (original plan)
  – Multi-root (better accepted)
  – SET has banks as CA’s and common SET root
• PGP Model
  – “Friends and Family approach” - S. Kent
• Other representations for certifications
• No certificates at all
  – Out of band key distribution
  – SSH
Federated Identity
Passport v Liberty Alliance

• Two versions of Passport
  – Current deployed version has lots of weaknesses and is centralized
  – Version under development is “federated” and based on Kerberos
Liberty Alliance
  – Loosely federated with framework to describe authentication provided by others.
Passport v1

• Goal is single sign on
• Implemented via redirections

Assigned reading: http://avirubin.com/passport.html
Federated Passport

- Announced September 2001
- Multiple registrars
  - E.g. ISPs register own users
- Kerberos credentials
  - Embedded authorization data to pass other info to merchants.
- Federated Passport is predominantly vaporware today, but .net authentication may be where their federated model went.
Liberty Alliance

- Answer to MS federated Passport
- Design criteria was most of the issues addressed by Federated Passport, i.e. no central authority.
- Got off to slow start, but to date has produced more than passport has.
- Use SAML (Security Association Markup Language) to describe trust across authorities, and what assertions means from particular authorities.
- These are hard problems, and comes to the core of what has kept PKI from being as dominant as orginally envisioned.
- Phased approach: Single sign on, Web service, Federated Services Infrastructure.
Federated Identity - Shibboleth

- Internet 2 Project
  - Federated Administration
  - Attribute Based Access Control
  - Active Management of Privacy
  - Based on Open SAML
  - Framework for Federation
Shibboleth - Architecture

• Service Provider
  – Browser goes to Resource Manager who uses WAYF, and users Attribute Requester, and decides whether to grant access.

• Where are you from service
  – Redirects to correct servers

• Federation
The Shibboleth Protocol

1. User requests resource

2. I don’t know you, or where you are from

3. Where are you from?

4. Redirect to IdP for your org

5. I don’t know you. Authenticate using your org’s web login

6. I know you now. Redirect to SP, with a handle for user

7. I don’t know your attributes. Ask the IdP (peer to peer)

8. Based on attribute values, allow access to resource

Source: Kathryn Huxtable khuxtable@ku.edu 10 June 2005
Generic Security Services API
Moving up the Stack

Standard interface for choosing among authentication methods

Once an application uses GSS-API, it can be changed to use a different authentication method easily.

Calls

- Acquire and release cred
- Manage security context
  - Init, accept, and process tokens
- Wrap and unwrap
Authentication in Applications

Unix login
Telnet
RSH
SSH
HTTP (Web browsing)
FTP
Windows login
SMTP (Email)
NFS
Network Access
Unix Login

One way encryption of password
Salted as defense against pre-computed dictionary attacks
To validate, encrypt and compare with stored encrypted password
May use shadow password file
Telnet

A remote login application
Normally just an unencrypted channel over which plaintext password sent.
Supports encryption option and authentication options using protocols like Kerberos.
RSH (Remote Shell/Remote Login)

Usually IP address and asserted account name.

Privileged port means accept asserted identity.

If not trusted, request unix password in clear.

Kerberos based options available

Kerberos based authentication and optional encryption
Secure Shell (SSH)

Encrypted channel with Unix login

Establish encrypted channel, using public key presented by server

Send password of user over channel

Unix login to validate password.

Public key stored on target machine

User generates Public Private key pair, and uploads the public key to directory on target host.

Target host validates that corresponding private key is known.
Web Browsing (HTTP)

Connect in the clear, Unix Password
Connect through SSL, Unix password
Digest authentication (RFC 2617)
    Server sends nonce
    Response is MD5 checksum of Username, password, nonce URI
User certificate, strong authentication
File Transfer Protocol

Password based authentication or GSS-API based authentication
Including use of Kerberos
Authentication occurs and then stream is encrypted
Windows Network Login

In Win2K and later uses Kerberos

In Win NT

Challenge response

Server generates 8 byte nonce

Prompts for password and hashes it

Uses hash to DES encrypt nonce 3 times
Email

SMTP – To send mail
Usually network address based
Can use password
Can be SSL protected
SMTP after POP
Email

Post Office Protocol
Plaintext Password
Can be SSL protected
Eudora supports Kerberos authent

IMAP
Password authentication
Can also support Kerberos
Email – Message Authentication

PGP and S/MIME

Digital Signature on messages
Message encrypted in session key
Optional
Hash of message encrypted in private key
Validation using sender’s public key
Email – Message Authentication

SPF and SenderID

- Authenticate domain of sender
- SPF record for domain in DNS
  - Specifies what hosts (i.e. mail server host) can send mail originating from that address.
  - Receivers may validate authorized sender based on record
  - Can falsely reject for forwarded messages
Email – Message Authentication

Domain Keys

– Public key associated with domain in DNS
– Originators MTA attaches signature
  ▪ Authenticates senders domain
  ▪ Not individual sender
  ▪ Signature covers specific header fields and possibly part of message.
– Messages may be forwarded
File System Authentication

Sun’s Network File System
Typically address based
Athena Kerberized version
Maps authenticated UID’s to addresses
NFS built on ONC RPC
ONC RPC has stronger Kerberos/GSSAPI support
File System Authentication

Andrew File System
Based on Andrew RPC
Uses Kerberos authentication

OSF’s DCE File System (DFS)
Based on DCE RPC
Uses Kerberos authentication
Network Access Servers

Radius

Problem: Not connected to network until connection established
Need for indirect authentication

Network access server must validate login with radius server.
Password sent to radius server encrypted using key between agent and radius server.
Delegated Authentication

Usually an authorization problem
How to allow an intermediary to perform operations on your behalf.
Pass credentials needed to authenticate yourself
Apply restrictions on what they may be used for.
Proxies

• A proxy allows a second principal to operate with the rights and privileges of the principal that issued the proxy
  – Existing authentication credentials
  – Too much privilege and too easily propagated

• Restricted Proxies
  – By placing conditions on the use of proxies, they form the basis of a flexible authorization mechanism
Restricted Proxies

- Two Kinds of proxies
  - Proxy key needed to exercise bearer proxy
  - Restrictions limit use of a delegate proxy
- Restrictions limit authorized operations
  - Individual objects
  - Additional conditions

PROXY CERTIFICATE
Conditions:
- Use between 9AM and 5PM
- Grantee is user X, Netmask is 128.9.x.x, must be able to read this fine print, can you
Next Generation Secure Computing Base (Longhorn)

- Secure booting provides known hardware and OS software base.
- Security Kernel in OS provides assurance about the application.
- Security Kernel in application manages credentials granted to application.
- Security servers enforce rules on what software they will interact with.
• End of Lecture 5

• Following slides are start of lecture 6
Announcements

Mid-term exam next Friday
9AM-10:40AM on October 6th
Followed by lecture
Open Book, Open Note
Review at end of this class
Past exams posted on web site
- If alternate location, will announce to class.
CSci530: Computer Security Systems
Lecture 6 – 30 September 2016
Authorization and Policy

Dr. Clifford Neuman
University of Southern California
Information Sciences Institute
Announcements

• Mid-term exam on October 7th
  – In class
  – Open Book
  – Open Note
  – No Electronics
Authorization

• Final goal of security
  – Determine whether to allow an operation.
• Depends upon
  ▪ Policy
  ▪ Possibly authentication
  ▪ Other characteristics
The role of policy in security architecture

Policy – Defines what is allowed and how the system and security mechanisms should act.

Enforced By

Mechanism – Provides protection interprets/evaluates (firewalls, ID, access control, confidentiality, integrity)

Implemented as:

Software: which must be implemented correctly and according to sound software engineering principles.
Policy: The Access Matrix

- Policy represented by an Access Matrix
  - Also called Access Control Matrix
  - One row per object
  - One column per subject
  - Tabulates permissions
  - But implemented by:
    - Row – Access Control List
    - Column – Capability List
Instantiations of ACMs

• Capabilities
  – For each principal, list objects and actions permitted for that principal
  – Corresponds to columns of ACM
  – Example: Kerberos restricted proxies

• The Unix file system is an example of…?
Problems

• Permissions may need to be determined dynamically
  – Time
  – System load
  – Relationship with other objects
  – Security status of host
Problems

• Distributed nature of systems may aggravate this
  – ACLs need to be replicated or centralized
  – Capabilities don’t, but they’re harder to revoke

• Approaches
  – GAA
Policy models: Bell-LaPadula

• Discretionary Policy
  – Based on Access Matrix
• Mandatory Policy
  – Top Secret, Secret, Confidential, Unclassified
  – * Property: S can write O if and only if Level S <= Level O
    ▪ Write UP, Read DOWN
  – Categories treated as levels
    ▪ Form a matrix
(more models later in the course)
Other Policy Models

• Mandatory Access Control
  – Bell-Lepadula is an example
• Discretionary Access Control
  – Many examples
• Role Based Access Control
• Integrity Policies
  – Biba Model – Like Bell-Lepadula but inverted
  – Clark Wilson
    ▪ Constrained Data, IVP and TPs
Role Based Access Control

- Similar to groups in ACLs, but more general.
- Multiple phases
  - Administration
  - Session management
  - Access Control
- Roles of a user can change
  - Restrictions may limit holding multiple roles simultaneously or within a session, or over longer periods.
  - Supports separation of roles
- Maps to Organization Structure
Integrity Policies

• Biba Model – Like BellLepadula but inverted
• Clark Wilson
  – Constrained Data, IVP and TPs
Authorization Examples

- **Access Matrix**
- **Access Control Lists**
  - `.htaccess (web servers)`
  - `Unix file access (in a limited sense)`
    - On login lookup groups
  - `SSH Authorized Keys`
- **Capabilities**
  - `Unix file descriptors`
  - Proxies mix ACLs and capabilities
Security is more than mix of point solutions

• Today’s security tools work with no coordinated policy
  – Firewalls and Virtual Private Networks
  – Authentication and Public Key Infrastructure
  – Intrusion Detection and limited response
• We need better coordination
  – Intrusion response affected at firewalls, VPN’s and Applications
  – Not just who can access what, but policy says what kind of encryption to use, when to notify ID systems.
• Tools should implement coordinated policies
  – Policies originate from multiple sources
  – Policies should adapt to dynamic threat conditions
  – Policies should adapt to dynamic policy changes triggered by activities like September 11th response.
GAA-API: Integration through Authorization

• Focus integration efforts on authorization and the management of policies used in the authorization decision.
  – Not really new - this is a reference monitor.
  – Applications shouldn’t care about authentication or identity.
    ▪ Separate policy from mechanism
  – Authorization may be easier to integrate with applications.
  – Hide the calls to individual security services
    ▪ E.g. key management, authentication, encryption, audit
Authorization and Integrated Security Services

- Firewalls
- Web Servers
- Databases
- IPSec
- Authentication

GAA API

EACL

SECURITY AUDIT RECORDS

Unauthorized Access to Firewalls

Unauthorized Access to Web Servers

Unauthorized Access to Databases

Unauthorized Access to IPSec

Unauthorized Access to Authentication

Unauthorized Access to Security Audit Records
Generic Authorization and Access-control API

 Allows applications to use the security infrastructure to implement security policies.

gaa_get_object_policy_info function called before other GAA API routines which require a handle to object EACL to identify EACLs on which to operate. Can interpret existing policy databases.

gaa_check_authorization function tells application whether requested operation is authorized, or if additional application specific checks are required.
Three Phases of Condition Evaluation

GAA-API

- gaa_get_object_policy_info()
- gaa_check_authorization() \( T/F/U \)
- gaa_execution_control() \( T/F/U \)
- gaa_post_execution_actions() \( T/F/U \)

System State

Copyright © 1995-2013 Clifford Neuman - UNIVERSITY OF SOUTHERN CALIFORNIA - INFORMATION SCIENCES INSTITUTE
GAA-API Policies originate from multiple sources

- Discretionary policies associated with objects
  - Read from existing applications or EACLs
- Local system policies merged with object policies
  - Broadening or narrowing allowed access
- Policies imported from policy/state issuers
  - ID system issues state credentials, These credentials may embed policy as well.
- Policies embedded in credentials
  - These policies attach to user/process credentials and apply to access by only specific processes.
- Policies evaluated remotely
  - Credential issuers (e.g. authentication and authorization servers) evaluate policies to decide which credentials to issue.
Communicating threat conditions

Threat Conditions and New Policies carried in signed certificates

– Added info in authentication credentials
– Threat condition credential signed by ID system

Base conditions require presentation or availability of credential

– Matching the condition brings in additional policy elements.
Integrating security services

The API calls must be made by applications.
– This is a major undertaking, but one which must be done no matter how one chooses to do authorization.

These calls are at the control points in the app
– They occur at auditable events, and this is where records should be generated for ID systems
– They occur at the places where one needs to consider dynamic network threat conditions.
– Adaptive policies use such information from ID systems.
– They occur at the right point for billable events.
Advances Needed in Policy

• Ability to merge & apply policies from many sources
  – Legislated policies
  – Organizational policies
  – Agreed upon constraints
• Integration of Policy Evaluation with Applications
  – So that policies can be uniformly enforced
• Support for Adaptive Policies is Critical
  – Allows response to attack or suspicion
• Policies must manage use of security services
  – What to encrypt, when to sign, what to audit.
  – Hide these details from the application developer.
GAA - Applications and other integration

- Web servers - apache
- Grid services - globus
- Network control – IPsec and firewalls
- Remote login applications – ssh
- Trust management
  - Can call BYU code to negotiate credentials
  - Will eventually guide the negotiation steps
What dynamic policies enable

• Dynamic policy evaluation enables response to attacks:
  – Lockdown system if attack is detected
  – Establish quarantines by changing policy to establish isolated virtual networks dynamically.
  – Allow increased access between coalition members as new coalitions are formed or membership changes to respond to unexpected events.
Demo Scenario - LockDown

- You have an isolated local area network with mixed access to web services (some clients authenticated, some not).
Demo Scenario - LockDown

- You have an isolated local area network with mixed access to web services (some clients authenticated, some not).
- You need to allow incoming authenticated SSH or IPSec connections.
Demo Scenario - LockDown

- You have an isolated local area network with mixed access to web services (some clients authenticated, some not).
- You need to allow incoming authenticated SSH or IPSec connections.
- When such connections are active, you want to lock down your servers and require stronger authentication and confidentiality protection on all accesses within the network.
Policies

• HIPAA, other legislation
• Privacy statements
• Discretionary policies
• Mandatory policies (e.g. classification)
• Business policies
Mechanisms

- Access Matrix
  - Access Control List
  - Capability list
- Unix file system
- Andrew file system
- SSH authorized key files
- Restricted proxies, extended certificates
- Group membership
- Payment
Summary

• Policies naturally originate in multiple places.
• Deployment of secure systems requires coordination of policy across countermeasures.
• Effective response requires support for dynamic policy evaluation.
• Such policies can coordinated the collection of data used as input for subsequent attack analysis.
Review for Mid-term

- Cryptography
  - Basic building blocks
  - Conventional
    - DES, AES, others
  - Public key
    - RSA
  - Hash Functions
  - Modes of operation
    - Stream vs. Block
Review for Mid-term

• Key Management
  – Pairwise key management
  – Key storage
  – Key generation
  – Group key management
  – Public key management
  – Certification
Review for Mid-term

• Authentication: Know, Have, About you
  – Unix passwords
  – Kerberos and NS
  – Public Key
  – Single Sign On
  – Applications and how they do it
  – Weaknesses
Review for Mid-term

• Authorization and Policy:
  – Access Matrix
    ▪ ACL
    ▪ Capability
  – Bell Lapadula
  – Dynamic Policy Management
  – Delegation
  – Importance of getting policy right