T-110.5211 Cryptosystems

Introduction to the course

11.9.2008

Sami Vaarala
Antti Nuopponen

Outline

- General course information
  - Basic course information
  - Practical arrangements
  - Course scope
  - Preliminary lecture & assignment schedule
  - Some related resources
- Introduction to cryptosystems
  - Basic concepts of safety, security, and threat modelling
  - Basics of cryptology, cryptosystems, and basic types of cryptographic primitives
  - Cryptosystem attacks and failures
  - Cryptosystem implementation

Basic course information

- T-110.5211 Cryptosystems (4 credits, not post-grad.)
- Course language = English
  - Lectures, slides, assignment, exams, web
- Course staff
  - Sami Vaarala (lecturer), Antti Nuopponen (assistant)
- Contact information
  - Web: https://noppa.tkk.fi/noppa/kurssi/t-110.5211/
  - E-mail (course staff): **t-110.5210@tml.hut.fi**
    - **NB: “5210” intended**: old course e-mail, used on this course too.
  - General news will be on course page (Noppa) => RSS, e-mail
- Passing the course
  - Assignment + exam (+ recommended lectures)

Course material

- Lecture slides
  - Exams will be based on the topics covered by slides
- Recommended books – used as reference material
  - Kaufman et al:
    Network Security, Private Communication in a Public World
    (ISBN 0130460192)
  - Stallings:
    Cryptography and Network Security, Principle and Practice
    (ISBN 0131115022)
  - Should be available from the HUT library

Course material

- Related books
  - Menezes et al: *Handbook of Applied Cryptography*
    - http://www.cacr.math.uwaterloo.ca/hac/
  - Schneier: *Applied Cryptography* / *Practical Cryptography*
  - Smart: *Cryptography: An Introduction*
  - Mao: *Modern Cryptography: Theory and Practice*
  - Stinson: *Cryptography, Theory and Practice*

Practical arrangements

- Signing up (course & exams)
  - Oodi
- Recommended lectures
  - Thursday 16:15-18:00, T1 (CS), schedule – course web page (Noppa)
  - Lectures not mandatory but recommended
  - Lecture slides will appear on course web page after each lecture
- Mandatory assignment
  - Programming in Java (more on following slides)
  - An individual assignment (each student completes on his/her own)
- Mandatory exam
  - December, January, May (see course web or Oodi)
  - Sign up through Oodi

Course scope

- Basics of cryptography
  - Security and threat modelling (background)
  - Overview of mathematics used in cryptography
  - Cryptographic primitives and cryptosystems as security tools
  - Most important algorithms (DES, AES, RSA, etc)
  - Some cryptographic protocols (especially TLS)
- Implementation issues
  - Cryptography as a component of a secure system
  - Selecting a cryptosystem with system requirements in mind
  - Support functions required by cryptosystems to work properly
  - Course assignment – some practical experience
- Practical software issues throughout SW life cycle
  - Considerations in practical cryptosystem implementation

Preliminary lecture schedule

- Preliminary schedule – follow course web page
  - 1: Intro: basic concepts, course info
  - 2: Crypto 1: math review, classical ciphers
  - 3: Crypto 2: symmetric primitives, modes of operation **A**
  - 4: Crypto 3: symmetric algorithms; DES; AES; MD5; SHA*; RC4 **A**
  - 5: Crypto 4: asymmetric; D-H, RSA, certs, sigs **A**
  - 6: TLS protocol in detail; overview of IPsec **A**
  - 7: Crypto 5: some cryptosystems **A**
  - [No lecture, exam period]
  - 8: Implementation issues (1)
  - 9: Implementation issues (2)
  - 10: Implementation issues (3)
  - 11: To be determined (probably no lecture)
  - 12: Course assignment review; summary and review of lectures

Course assignment

- Crypto programming in Java, two major components
  - Basic TLS framework with incomplete crypto library given
  - Weekly deliverables, starting after third lecture
- Component #1: simple crypto library
  - Implement generic primitives such as HMAC, CBC, etc.
  - Test vectors for each primitive given to aid in debugging
  - Each primitive is implemented separately
- Component #2: TLS specifics
  - Implement TLS specific crypto primitives
  - Modify the TLS framework to use implemented primitives
  - End result is able to negotiate a TLS connection with a HTTPS web server and use HTTP GET to fetch a protected web page
- Details may still change

Course assignment

- More info
  - Assignment pages in course web page
  - You can ask questions after each lecture
  - Or use course e-mail (please don't use our personal e-mail addresses)

Assignment deliverables (tentative)

- See course page for updates, here's a **tentative schedule**

11 Sep	-	-
18 Sep	-	-
25 Sep	8 Oct	Vigenere cipher Generalized block cipher CBC mode
2 Oct	15 Oct	3DES (given DES and CBC) HMAC (given MD5 & SHA1)
9 Oct	22 Oct	Diffie-Hellman RSA signature and padding check
16 Oct	12 Nov	TLS P-function TLS PRF-function TLS signature checking
23 Oct	12 Nov	Trusted certificate manager
EXTRAS	19 Nov	MD5 from scratch, SHA1 from scratch, DES from scratch

Grading

- Assignment
  - 5 individual deliverables + 3 extras (8 total), each pass/fail
  - Less than 3 passed => assignment fails
  - Grading

# Passed	Grade
0-2	0
3	1
4	3
5-8	5

- Exam
  - Normal 0-5 grading
- Overall grade
  - Must pass both exam and assignment
  - 50% assignment, 50% exam, round up

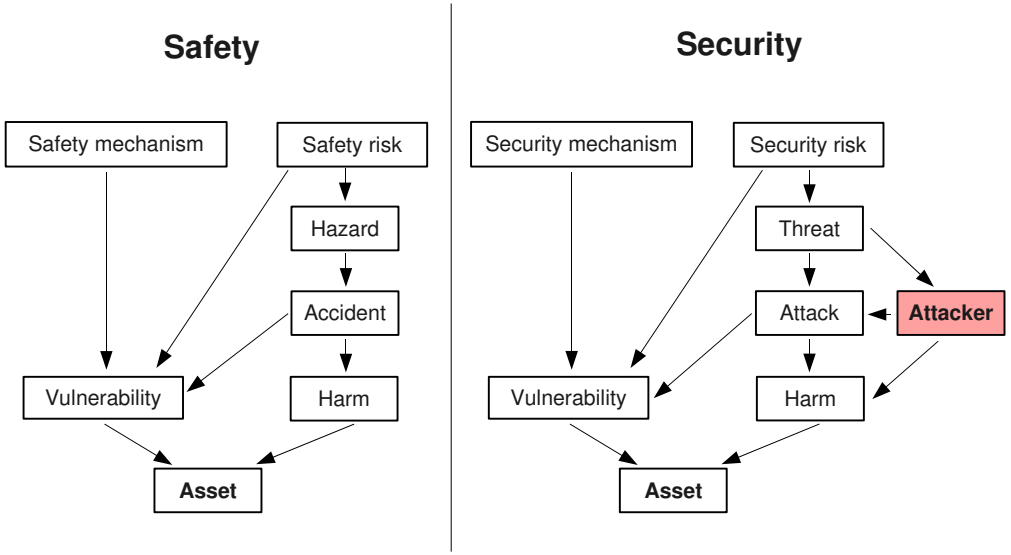
Introduction to cryptosystems

Safety and security in systems

- Safety, security, survivability (*)
  - **Safety**: “the degree to which **accidental harm** is prevented, detected, and reacted to”.
  - **Security**: “the degree to which **intentional harm** is prevented, detected, and reacted to”.
  - **Survivability**: “the degree to which essential, **mission critical services** continue to be provided in spite of **either accidental or intentional harm**”.
- Safety and security are quality factors
  - Quality factor <= Dependability <= Security
  - Quality factor <= Dependability <= Safety
  - **Dependability**: “the degree to which various kinds of users can depend on a work product”.
- Security is often in conflict with other desirable qualities

(*) Firesmith, *Common Concepts Underlying Safety, Security, and Survivability Engineering*, CMU/SEI-2003-TN-033

Safety and security compared



Assets can include people, financial assets, information, reputation, availability of resources, ...

Threat model

- Threat model describes relevant threats of a system
  - Identify assets – physical, information, people, etc
  - Identify entry points, trust boundaries, and trust levels
  - Estimate capabilities of an attacker and possible attack scenarios
  - **=> Security is always relative to expectations**
- E.g. Dolev-Yao (1981) threat model for crypto protocols
  - An attacker
    - can obtain any message passing through the network;
    - is a legitimate user of the network, and thus in particular can initiate a conversation with any other user;
    - will have the opportunity to be a receiver to any principal;
    - can send messages to any principal by impersonating any other principal.
  - In other words: the attacker carries your messages

Security and layering

- Security occurs on many layers
  - System level security
  - Application security
  - Protocol security
  - Operating system security
  - Hardware platform security
  - Physical security
  - Network security
- Relevance of cryptography
  - Provides security mechanisms for improving security at every level
  - Mechanisms range from encrypted network protocols to cryptographic tokens for physical access

Creating secure systems

- Defining security requirements
  - Threat model, trust model, analysis
- Software process
  - Design, implement, (security) testing
- Maintenance
  - Security fixes and other updates
  - New features without breaking security
- Review of **correspondence** at each step
  - Requirements are correct (match stakeholder's needs)
  - Design matches security requirements
  - Implementation matches design (security testing)
  - Etc

Background on cryptology

- Cryptography and cryptosystems are old
  - Classical example of Caesar cipher, Enigma (WW2), etc.
  - 1970s => cryptography to private sector; DES 1977
- Cryptology = cryptography + cryptanalysis
  - Cryptography: the study of mathematical techniques related to information security (e.g. encryption)
  - Cryptanalysis: the study of mathematical techniques to defeat cryptography
  - => **Adversarial process** in developing cryptographic primitives
- Computer security
  - DoD orange book: “Trusted Computer System Evaluation Criteria” (TCSEC), security modelled basically as access control
  - Modern equivalent: Common Criteria (CC)
  - Cryptography can be used e.g. as a tool in enforcing access control

Cryptographic primitives

- What is a cryptographic primitive?
  - A mathematical entity which meets defined (security) requirements
- Common division
  - Unkeyed
  - Secret-key (symmetric) – “man made”
  - Public-key (asymmetric) – “math made”
- Kerckhoffs' principle – public algorithm, secret key
- Examples
  - (Symmetric) encryption – DES, 3DES, AES, Blowfish, RC4
  - (Asymmetric) encryption – RSA, DSA
  - (Unkeyed) hashing – MD5, SHA1, SHA-{256,384,512}
  - (Symmetric) message authentication – HMAC-{MD5,SHA1}
  - (Asymmetric) digital signatures – RSA, DSA
  - (Asymmetric) key exchange – Diffie-Hellman (DH)

Cryptosystem

- Cryptosystem (or cryptographic system)
  - **Provides** information security services
  - **Through** a combination of cryptographic primitives, protocols, operational procedures, documentation, user training, etc...
  - Encompasses systems of various sizes
    - One algorithm (e.g. RSA cryptosystem)
    - Or widely: protocols, hardware, customer training, etc.
- Services provided by a cryptosystem (= CIA)
  - Confidentiality
  - Integrity (+ Authentication)
  - Availability
- In addition
  - Replay protection, traffic flow confidentiality, anonymity, non-repudiation, watermarking, etc.

Concepts of cryptographic security

- Perfect secrecy
  - It is information-theoretically impossible to decrypt a message even with infinite resources (but no defense against tampering)
  - Example: cryptographic one-time pad
- Computational security
  - Infeasible (realistic resource limits), to e.g. decrypt message
  - Brute force of 2^k keys should be best attack = k-bit effective key
  - Example: RSA encryption and signatures
- Provable security
  - A proof of security shows that a cryptosystem is no less secure than the cryptographic primitives it uses, i.e. primitives are “used properly”
  - The proof shows that an attack on the cryptosystem yields an attack on the primitive(s)
  - Of course, if primitive insecure, the cryptosystem may also be

Role of cryptosystem

- Role of cryptosystem in the whole system
  - A tool in providing security (also safety, even survivability)
  - Decrease vulnerability to data communication threats
  - Decrease vulnerability to local threats
  - May also create new threats (e.g. loss of an important key)
- Cryptosystem is part of a larger system
  - Cryptosystem provides a safe method for handling (plaintext) data, and ensures data is reasonably secure when transmitted or stored
  - Typically the larger system communicates with other systems through untrusted channels (e.g. network, hard drive)
  - A cryptosystem is implemented using software & hardware
- Cryptosystem should match requirements
  - Threat & trust model
  - Excessive crypto = extra complexity, performance loss

Potential attacks & failures of cryptosystems

- Cryptosystem environment
  - Deficiencies in deployment (e.g. faulty default configurations)
  - Organizational processes, social engineering; changes in environment
- Cryptosystem implementation (hardware or software)
  - Implementation deficiencies (e.g. buffer overflow, logical errors in security policy code)
  - Causing a hardware malfunction (e.g. smartcard attacks)
- Cryptographic protocol / cryptosystem (protocol failure)
  - Exploit some weakness in how the cryptosystem has been constructed, without a need to break any cryptographic primitives
  - E.g. exploit a replay protection weakness (duplicate message)
- Cryptographic primitive
  - Attack the underlying cryptographic primitive (e.g. brute force DES or RSA)

Software (implementation) life cycle

- Attacks on implementation **very significant**
  - Hardly anyone attacks cryptographic primitives!
  - Most typical implementation vulnerability – buffer overflow
- Security must be “built into the system”, at every step
  - Requirements, design, implementation, testing, deployment, maintenance, retirement
  - Retrofitting security is possible, but difficult
- Several life cycle models
  - Some view that waterfall model is best suited to development of security critical systems
  - Regardless of your life cycle model, security needs to be considered throughout
  - Integrating components from external sources (e.g. open source) is challenging

Snapshot of some vulnerabilities (www.cert.org, 26.8.2005)

Recent Advisories & Incident Notes

US-CERT Technical Cyber Security Alert TA05-229A
Apple Mac Products are Affected by Multiple Vulnerabilities
Buffer overflow++ => execute arbitrary code remotely

US-CERT Technical Cyber Security Alert TA05-224A
VERITAS Backup Exec Uses Hard-Coded Authentication Credentials
Hard-coded auth credentials => unauthenticated remote access

US-CERT Technical Cyber Security Alert TA05-221A
Microsoft Windows and Internet Explorer Vulnerabilities
Buffer overflow++ => execute arbitrary code remotely

US-CERT Technical Cyber Security Alert TA05-210A
Cisco IOS IPv6 Vulnerability
Undisclosed cause => execute arbitrary code from local network segment

US-CERT Technical Cyber Security Alert TA05-194A
Oracle Products Contain Multiple Vulnerabilities
Buffer overflow++ => execute arbitrary code remotely

US-CERT Technical Cyber Security Alert TA05-193A
Microsoft Windows, Internet Explorer, and Word Vulnerabilities
Buffer overflow++ => execute arbitrary code remotely

US-CERT Technical Cyber Security Alert TA05-189A
Targeted Trojan Email Attacks
General trojan attacks => e.g. execute arbitrary code

more advisories | incident notes

Most preventable using safer programming languages
All implementation related (Cisco ?)

=> How relevant is implementation security for a cryptosystem ?

Summary

- Basic course information
  - Course web page – https://noppa.tkk.fi/noppa/kurssi/t-110.5211/etusivu
  - Contact staff by e-mail – **t-110.5210@tml.hut.fi**
  - Lectures, assignment, exam
- Introduction to cryptosystems
  - Safety, security, survivability; threats and threat model
  - Basics of cryptology; cryptography, cryptanalysis, cryptosystem
  - Relationship between a cryptosystem and system design
  - Cryptosystem attacks and failures
  - Implementation issues
- Next lecture
  - Review of crypto mathematics
  - Classical ciphers