Short Introduction to Cryptography

April 2, 2002

Christopher Boumenot

Purpose of Cryptography

- Confidentiality
- Authentication
- Integrity
- Nonrepudiation
- Access Control
- Availability

Encryption Basics

Encryption algorithms use two basic principles

- Substitution: each element of plaintext is mapped into another element
- Transposition: elements in the plaintext are rearranged

Categories of Encryption

- Symmetric: sender and receiver use the same key (aka single-key, and secret-key)
- Asymmetric: sender and receiver use different keys (aka two-key, and public-key)

Processing Encryption

- Block cipher: processes the input a block of elements at a time (typically 64-bits)
- Stream cipher: processes the input continuously producing an element at a time

Viability

- No encryption scheme is full proof!
- Two requirements are needed to make encryption viable:
  - The cost of breaking exceeds the value of the encrypted information
  - The time required to break the cipher exceeds the useful lifetime of the information

Cryptanalysis

Definition: attempting to break a cryptography algorithm

- Brute force: exhaustively searching the entire key space
- Dictionary: using well known words to guess the key

Exhaustive Key Search

- It is difficult to estimate the amount of effort required to cryptanalyze ciphertext successfully (more on this later)
- The strength of an algorithm is typically based on key size
- Usually only 50% of key space has to be searched for success

Exhaustive Key Search (cont.)

Key Size	Number of Alt. Keys	1 encryption/μs	10^6 encryptions/ μs
32	$2^{32} = 4.3 \times 10^9$	2^{31} μs = 35.8 min	2.15 ms
56	$2^{56} = 7.2 \times 10^{16}$	2^{55} μs = 1142 yrs	10.01 hrs
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} μs = 5.4×10^{24} yrs	5.4×10^{18} yrs
26 chars.	$26! = 4 \times 10^{26}$	2×10^{26} μs = 6.4×10^{12} yrs	6.4×10^6 yrs

Diffusion and Confusion

The process of thwarting cryptanalysis based upon statistical analysis

- Terms were introduced by Claude Shannon in 1945 (1949).
- Diffusion: statistical structure of the plaintext is dissipated into long-range statistics
- Confusion: relationship between the statistics of the ciphertext and the value of the encryption key is as complex as possible

Three Popular Forms of Encryption

- Hash functions
- Block ciphers
- Public Key

Hash Functions

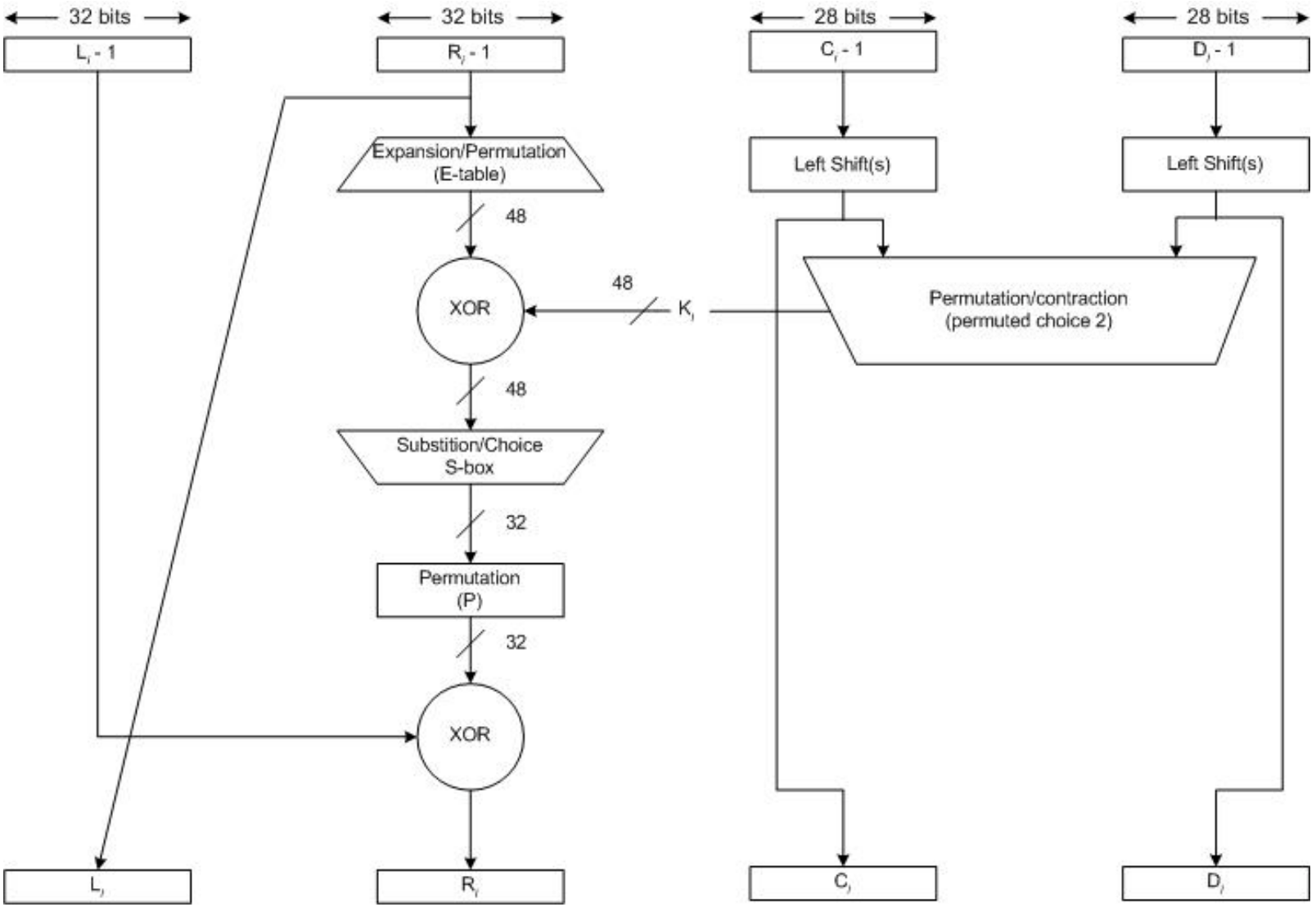
- Accepts an arbitrary sized input and produces a fixed size output
- Provides error detection
- One-way: for any give code h, it is computationally infeasible to find x such that H(x) = h
- Weak collision resistance: given a block x it is computationally infeasible to find $x \neq y$ with H(y) = H(x)
- Strong collision resistance: computationally infeasible to find any pair (x,y) such that H(x) = H(y)
- It's easy to generate a code given a message, but virtually impossible to generate a message given a code
- Examples: MD4, MD5, SHA-1, RIPEMD-160, Crypt3

Block Cipher

- Operates on a fixed number of elements at a time
- All most all block ciphers are based upon a structure created by Feistel, called the Feistel Cipher
- Feistel Cipher is composed of multiple iterations of substitutions, and permutations
- Feistel's Cipher is a practical application of Shannon's work
- Examples: DES, 3DES, AES, Blowfish, Twofish

Data Encryption Standard (DES)

- Most widely used encryption standard
- Developed by IBM in the late 1960's as part of a research project on computer cryptography
- A revised edition was developed for the NSA
- The key size of 128-bits was reduced to 56-bits


32 bits
32 bits
28 bits
28 bits
L_i - 1
R_i - 1
C_i - 1
D_i - 1
Expansion/Permutation
(E-table)
Left Shift(s)
Left Shift(s)
48
48
XOR
48
K_i
Permutation/contraction
(permuted choice 2)
48
Substition/Choice
S-box
32
Permutation
(P)
32
XOR
L_i
R_i
C_i
D_i

Public Key Encryption

- Based on mathematics as opposed to substitution and permutation
- Mostly used for key management and signature applications
- Computationally expensive compared to other encryption algorithms
- Composed of two keys: a key for encryption, and a key for decryption (doesn't matter which one)
  - Public Key: encryption
  - Private Key: decryption

RSA Algorithm

- Developed by Rivest, Shamir, and Adleman

$$C = M^{e} \bmod n$$

$$M = C^{d} \bmod n = (M^{e})^{d} \bmod n$$

Plaintext block: M, Ciphertext block: C

RSA (cont.)

- Both sender and receiver must know the value of *n*
- The sender knows the value of *e*
- Only the receiver knows the value of *d*
  - Public Key: KU = {e,n}
  - Private Key: KR = {d,n}

Key Generation

- Select p,q (both prime)
- Calculate $n = p \times q$
- Calculate $\Phi(n) = (p-1)(q-1)$
- Select integer e: $\gcd(\Phi(n),e) = 1$;
  $1 < e < \Phi(n)$
- Public Key: KU = {e,n}
- Private Key: KR = {d,n}

Numbers…Please!

- Using freely available libraries benchmark results were gathered to determine the amount of time it for various encryption algorithms to execute
- SSL handshake performance was benchmarked
- The OpenSSL and Crypto++ libraries were used to obtain the results

Crypto Benchmark

Algorithm	Bytes Processed	Time	MB/s	Crypt/s
CRC-32	134217728	0.703	182.07	N/A
MD5	134217728	0.922	138.83	N/A
SHA-1	67108864	1.078	59.369	N/A
DES	16777216	1.094	14.625	239620
Blowfish	16777216	0.750	21.333	349525
AES (128)	33554432	0.953	33.578	249823
AES (192)	33554432	1.125	28.444	233016
AES (256)	33554432	1.266	25.276	207064

Crypto Benchmark (cont.)

Operation	Iteration	Total Time	ms/op
RSA 512 Encrypt	8885	1.000	0.11
RSA 512 Decrypt	692	1.000	1.45
RSA 1024 Encrypt	3992	1.000	0.25
RSA 1024 Decrypt	137	1.000	7.30
RSA 512 Sign	689	1.000	1.45
RSA 512 Verify	9830	1.000	0.10
RSA 1024 Sign	135	1.000	7.41
RSA 1024 Verify	4263	1.000	0.23

Kerberos

Jennifer G. Steiner
Clifford Neuman
Jeffrey I. Schiller

Presented by
Christopher Boumenot

Project Athena

- Started at MIT in 1983 to integrate computers into the curriculum
- Over 6,000 computers had to be integrated
- Other projects came out Athena, including the X windowing system
- Athena – Greek Goddess of wisdom, justice, war, culture, law, and crafts

Kerberos

- Designed to securely manage all of the computers in the Athena project
- Watchdog of Hades
- Usually had three heads, a serpent's tail, a mane of snakes, and a lion's claw
- Kerberos supposed to have 3 tasks – authentication, auditing, and accounting, only one was implemented

Access Control

Three approaches to access control:

1. Nothing
2. Require the host to prove its identity but trust the host word's as to who to user is (rsh, rlogin)
3. Require the user to prove his identity for each required service, and server must prove its identity

Goals of Kerberos

- Secure
- Reliable
- Transparent
- Scalable

Kerberos

- Based upon the protocol proposed by Needham and Schroeder
- Only conventional encryption was used
- Kerberos IV makes use of DES
- Kerberos I, II, and III were internal versions

Components of Kerberos

- Administrative Server (KDBM) available in master and slaves
- Authentication Server (Kerberos server)
- Ticket Granting Server (TGS)
- Encryption Library
- Database Library
- User Programs
- Applications

What's in a Name?

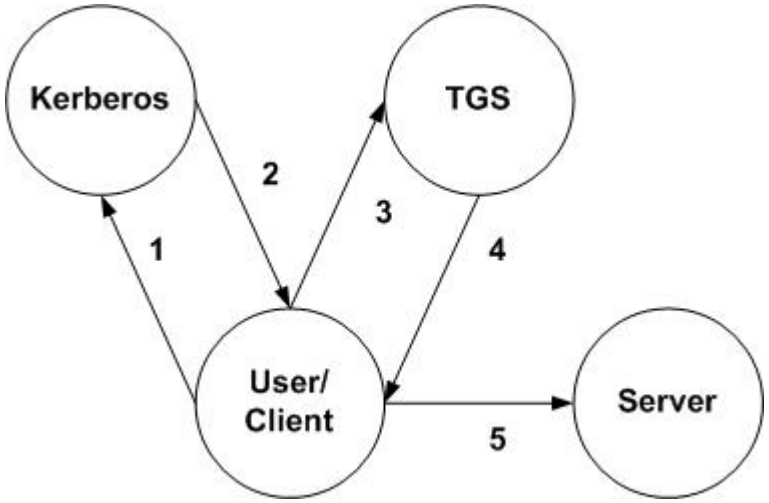
- Consists of a primary name, an instance, and a realm expressed as name.instance@realm
  - Primary name: name of user or service
  - Instance name: can be used to indicate other privileges such as root
  - Realm: name of an administrative entity that maintains authentication data

Logon Process

- User obtains credentials to be used to request access to other service
- User requests authentication for a specific service
- User presents the granted credentials to the end server

Kerberos Authentication Protocol

1. Request for TGS ticket
2. Ticket for TGS
3. Request for Server ticket
4. Ticket for Server
5. Request for service



Credentials

- Two types of credentials:
  - Ticket: securely passes the identity of the user between the authentication server and the end server
  - Authenticator: contains information that when compared against a ticket proves that the client presenting the ticket was the same one the ticket was issued too

Ticket

$$\{s,c,addr,timestamp,life,K_{s,c}\}K_s$$

- Good for a single server and service
- Ticket contains information such as name of server, IP address of client, timestamp, a lifetime, and a random session key (RSK)
- Ticket is encrypted using the key of the server it is to be used for

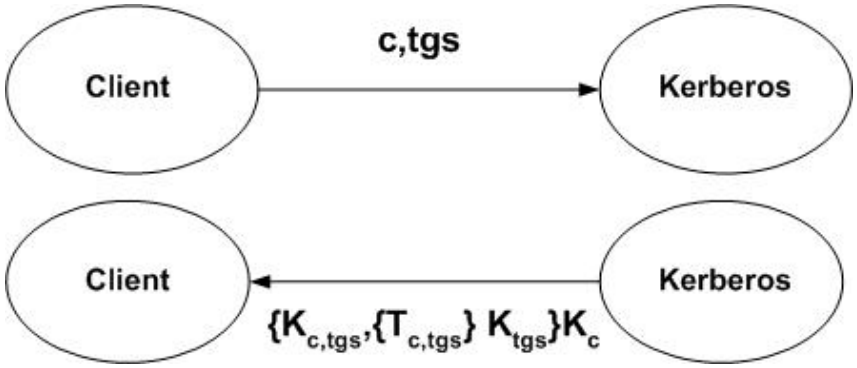
Authenticator

$$\{c, addr, timestamp\}K_{s,c}$$

- Unlike a ticket an authenticator can only be used once
- This is not a issue because the client can build all the authenticators it needs

Logging On

- Client makes request to Kerberos with user name and TGS
- Server verifies it knows the client, and generates a RSK
- Server creates a ticket for the TGS
- Ticket is encrypted in a key known only to the TGS and Kerberos server
- The client's key (derived from the user's password) is used to decrypt the message

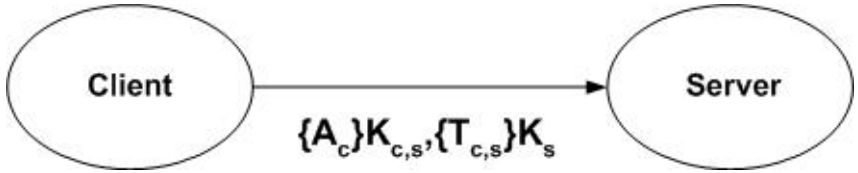


Service

- To gain access to a server, the application builds an authenticator containing the client's name, IP address, and current time
- Authenticator is encrypted using the session key that was received with the ticket for the server

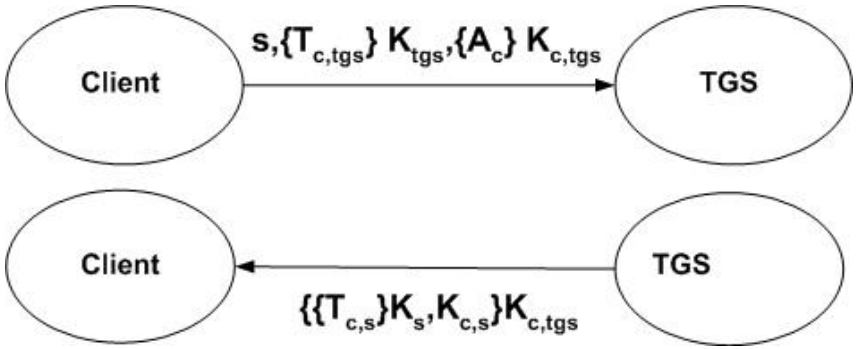
Requesting a Service

- Assume the user already has a ticket for the server
- Authenticator is built
- Client sends the authenticator with the ticket to the server
- Server decrypts ticket, then the authenticator and verifies the client's identity



My First Ticket

- Every time a program wants to make use of a service it doesn't yet have a ticket for it makes a request to the TGS
- It builds an authenticator and the service that it wants to use



My First Ticket (cont.)

- TGS builds a new RSK to be used between the client and server. It then builds a ticket for the new server containing the client's name, server's name, current time, the client's IP address, and the new session key it generated.

Kerberos Database

- Database is encrypted in master's key
- Multiple databases can be used for fault tolerance, speed, and efficiency
- Only the master database is allowed to accept changes
- Replication entails the master database dumping its contents every hour and pushing them to the slaves

KDBM Server

- KDBM only accepts requests to add principles or change the password for existing principles
- TGS will not grant tickets for the KDBM, only the authentication service can do this
  - This prevents other people from changing one principal's password if they leave a machine unattended

Inter-realm Access

- User's will want to communicate with other realms
- Realms must agree on a key to share for inter-realm access

Faults…

- DES (encryption dependence)
- Protocol dependence
- Ticket lifetime too short
- Inter-realm access is poor
- Proxy support

- Kerberos V addresses these shortcomings

Conclusions

- Kerberos is a transparent, reliable, distributed authentication system for computer networks
- Kerberos can be added to current applications for integration into the current infrastructure, and security needs