Integrity and Authentication

MAC, Message Digest, HMAC
Authentication, Passwords, Digital Signature

Agenda

- **MAC, Hash and Message Digest**
- **Authentication**
- **Digital Signature**
- **Summary**

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

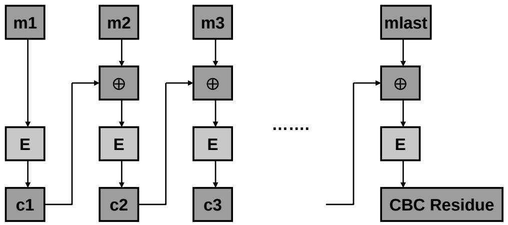
Message Authentication Code 1

- **Integrity instead of privacy is purpose of using the encryption system**
  - security aspect is integrity protection of message
- **Secret-key encryption system can be used**
  - to generate a cryptographic checksum in order to protect against undetected modifications
  - so called Message Authentication Code (MAC)
    - sometimes MIC (Message Integrity Code) is used which better describes what is achieved

Message Authentication Code 2

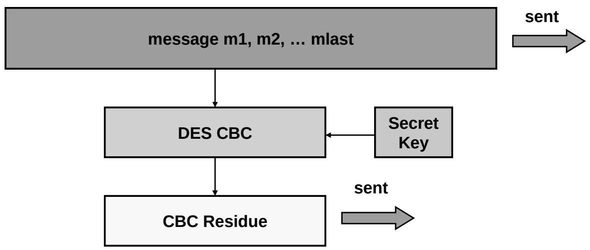
- **One possible way**
  - perform a kind of CBC on the plaintext in block mode using senders secret key
  - this will create a ciphertext called CBC residue (64-bit ciphertext) for the last plaintext block
  - send plaintext message along with CBC residue
  - receiver knowing the secret key can perform same CBC residue based on the received plaintext
  - compare computed CBC residue against received CBC
  - if an intruder had modified the plaintext message the CBC will not be the correct value
    - because generation of a correct CBC for the modified message needs the secret key

MAC CBC Residue 3



Encryption at sender to create CBC residue
Same done at the receiver in order to check

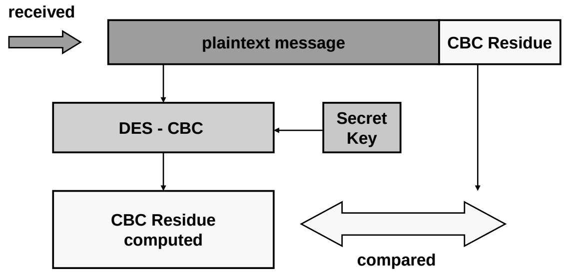
MAC CBC Residue Sender 4



Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

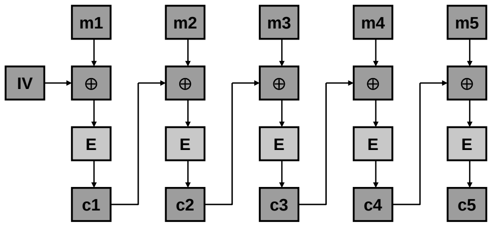
MAC CBC Residue Receiver 5



Integrity and Privacy in One Pass

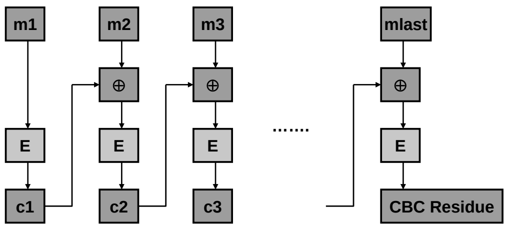
- **How to combine privacy with integrity?**
- **One simple idea would be:**
  - perform DES CBC encryption on the plaintext to get a ciphertext
    - remember that manipulating of a block by an active intruder could influence the next block (e.g. salary) in a certain way and will lead to garbage in the block (e.g. job function) which was manipulated
    - but what is about automatic detection of garbage: if the message is an English text then a human can easily detect it, but will a computer e.g. check the string for job function?
    - One holy grail of cryptographic protocol design was finding a method, which allows only the usage of a single cryptographic pass over the data which protects both privacy and integrity
  - perform CBC residue on the plaintext to get a cryptographic checksum

DES Mode – CBC for Privacy



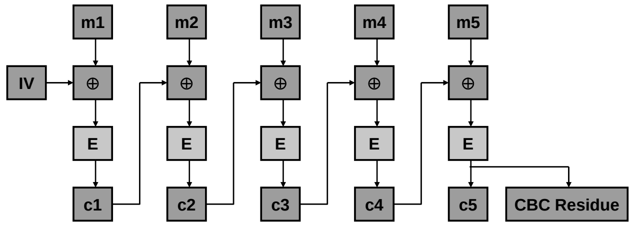
Encryption with CBC

CBC Residue for Integrity (MAC)



Encryption at sender to create CBC residue
Same done at the receiver in order to check

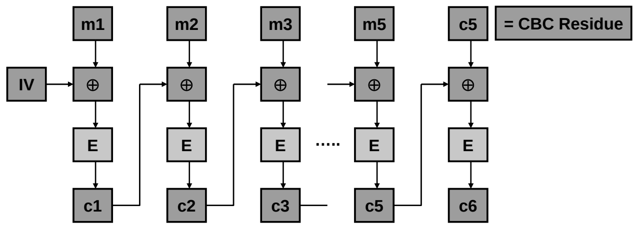
Combining Privacy and Integrity by One Cryptographic Function?



CBC Residue would just repeating the final block (c5 in this case)

!!! This will not enhance security !!!

Combining Privacy and Integrity by One Cryptographic Function?



Build CBC Residue, add it at the end of the plaintext message and encrypt it
But c5 EXOR with c5 will result in zero !!!

!!! This does not work !!!

Integrity and Privacy in Two Passes

- **One possible secure way to combine privacy and integrity**
  - perform DES CBC encryption with one secret key
  - perform CBC residue with a different secret key
  - send encrypted message plus CBC residue
- **Unfortunately this will need twice the cryptographic power**

Hash Function

- **Hash is a one-way function**
  - which takes an input (message) and produces an output
- **One-way function because of its mathematical nature**
  - h = H(M) of length k
    - given M, it is easy to compute h
    - given h, it is hard or infeasible to compute M
    - given M, it is hard to find another M', such that H(M)=H(M')
- **Other name for such a function performed on the bits of a message is**

<u>Message Digest</u>

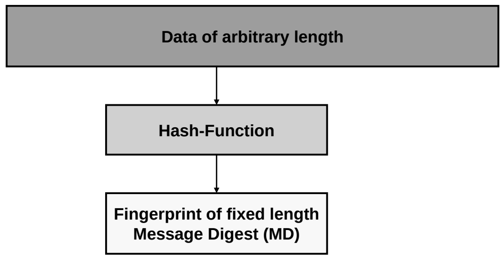
Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Message Digest (MD)

- **Message Digest is like a digital fingerprint**
  - small pieces of data that can serve to identify much larger digital objects
- **Message Digest**
  - must fulfill a sort of randomness and must be cryptographically strong
    - output should look random as with secret-key encryption
    - it is possible to create a MD based on a given message and the well-know function, but it should not be possible to predict any portion of the output
    - the only way to create the same MD for two different messages is to try out all possibilities (so take two random messages and create MDs for them and compare result)

Message Digest in Action



Message Digest

- **Because MD length k is smaller then number of message bits**
  - many messages will yield the same MD
    - e.g. for k=128 and a 1000-bit message there are on the average 2^{872} messages that map to any one particular MD
  - finding two messages which map to a given MD in example above
    - approximately 2^{64} messages must be checked based on the generic result of the birthday problem (n > square root (k))
      - $k = 2^{128}$ (amount of MDs)
      - $n > 2^{64}$ (amount of messages with same MD)
- **k should be at least 128**
  - because searching 2^{64} is not computationally feasible given the current state of computer technology

Birthday Problem

- **Example in probability courses:**
  - How many students do you need in a class before the probability of having two people with the same birthday exceed 50%?
  - Students assume more than 100 but probability theory says: it is just 23
  - birthday is like a hash (unpredictable function) for people (input n, messages) to one of 365 values (output k, message digest) and we are looking for two with the same birthday
    - we can build n*(n-1)/2 pairs
    - for each pair there is a probability of 1/k
    - we need k/2 pairs to in order for the probability to be about 50%
    - [n*(n-1)/2] > k/2 gives approximately $n^2 > k$ or n > square root (k)

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Message Digest

- **Symmetric block cipher can be used as one-way hash function (see MAC)**
  - hashing large messages
    - CBC fixed key, last ciphertext block is the hash
  - in real life more complex schemes
- **Public-key encryption can produce a hash in block chaining mode**
  - too slow for practical applications
- **Computing MD**
  - is therefore often done by other functions to achieve higher performance
  - MD2, MD5, SHA-1, etc…

MD Variants 1

- **MD2, MD5 are used in PEM (Privacy enhanced Mail)**
- **all designed by Ron Rivest**
- **MD2 (RFC 1319)**
  - 128-bit hash
  - dependent on random permutation of bytes
  - padding, append checksum, create 48-byte block, apply compression function, shuffle, chain
  - no weaknesses found
  - relatively slow

MD Variants 2

- **MD4 (RFC 1320)**
  - 128-bit hash, direct security, speed, simplicity and compactness, optimized for little-endian (Intel) architectures
  - 3 passes
  - successful partial attacks, obsoleted by MD5
- **MD5 (RFC 1321)**
  - 128-bit hash
  - padding, length inclusion, 4 chaining variables, 4 rounds (each 16 steps) in main loop
  - there is a weakness in the compression function, so collision resistance is violated in some cases

MD Variants 3

- **Secure Hash Standard**
  - Secure Hash Algorithm (SHA-1) ensures the security of Digital Signature Algorithm (DSA)
  - message less then 2^{64} bits, creates 160-bit hash
  - based on the ideas in MD4
  - padding, five chaining variables, 4 rounds of 20 operations each with non-linear functions
  - SHA is similar to MD5 with the addition of an expand transformation
  - more resistant to brute force attacks
  - no know weaknesses

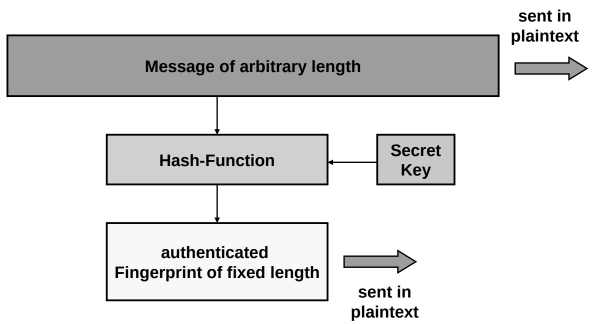
Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

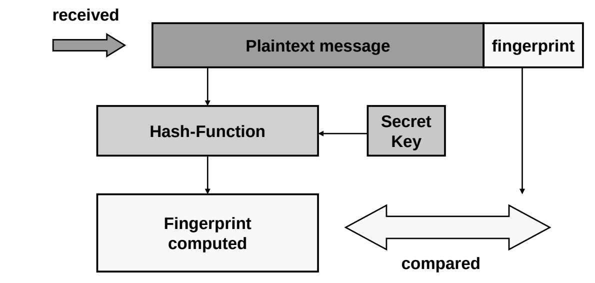
Hash Message Authentication Code (HMAC)

- **Message Digest does not provide security to transmission**
  - vulnerable to man-in-the-middle attacks
  - an attacker could intercept the message, change it, recalculate the MD based on the well-known algorithm and append it to the message
- **Hash Message Authentication Code (HMAC)**
  - use an additional secret-key as input to the hash function
  - secret-key is known to sender and receiver
  - authentication and integrity assurance
  - based on existing functions
  - e.g. keyed MD5, keyed SHA-1

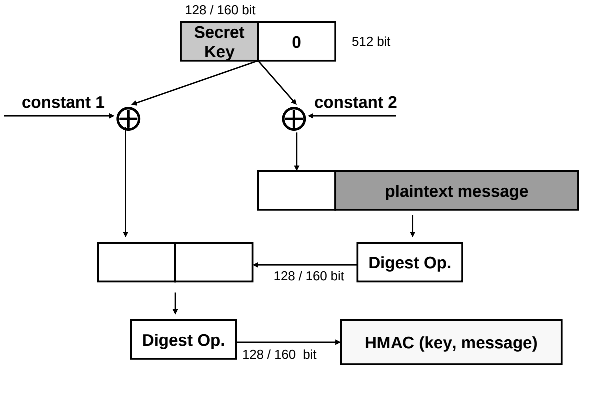
HMAC in Action (Sender) 1



HMAC in Action (Receiver) 2



HMAC Generation Details 3



© 2009, D.I. Manfred Lindner

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Agenda

- **MAC, Hash and Message Digest**
- **Authentication**
- **Digital Signature**
- **Summary**

General Aspects 1

- **Authentication**
  - is the process of proving someone's (person) or something's (computer) identity
  - fundamental component for any access control technique
  - involves challenging a person / computer to prove
    - that she/he has the knowledge of something (one-factor authentication) – "you know"
    - and additionally she/he has physical possession of something (two-factor authentication) – "you have"
    - or instead of "you have" and "you know" sometimes "you are" is challenged (in case of biometrics)
- **Basic elements for authentication:**
  - principal itself (the user, device or service requesting access)
  - credentials the principals submits as proof of identity
    - shared key (e.g. password), one-time password (OTP), digital certificate (comparable with a passport), biometrical features (fingerprint, voice, retina,…)
  - contextual information describing the transaction
    - (location, time-of-day, software state of a machine)

© 2009, D.I. Manfred Lindner

General Aspects 2

- **Examples for two-factor authentication**
  - ATM (Automatic Teller Machine – “Bankomat”)
    - Bankomat Card - I have something
    - PIN (Personal Identification Number) - I know something
  - OTP (Token card)
    - Token (credential) - I have something
    - PIN (unlocks credential)- I know something
- **Basic considerations**
  - authentication targets
    - human
    - computer programs sending messages
  - eavesdropping and impersonation

General Aspects 3

- **Token based authentication**
  - Password, dumb cards
  - Cryptographic challenge–response cards
  - Cryptographic calculator
  - Smart cards
- **Biometrical authentication**
  - Fingerprint readers
  - Handprint readers
  - Voiceprints
  - Iris scanner, Retinal scanners
  - Keystroke timing
  - Signatures

© 2009, D.I. Manfred Lindner

General Aspects 4

- **Any <u>authorization</u> is based on successful authentication performed before**
  - a person/computer is allowed to access certain areas of a computer and to perform certain actions on information stored there
    - e.g. getting money from ATM machine but only of own bank account
- **Authentication methods**
  - Password-based authentication
  - Address-based authentication
  - Cryptographically strong authentication

Password-based Authentication 1

- **Password**
  - a secret quantity that you state to prove you know it
- **Problems**
  - eavesdropping
    - e.g. Telnet / FTP with cleartext password
  - password storage
    - where and how to store on a machine
  - human factor
    - password in some convenient (memorable) form
    - one password used for multiple places

© 2009, D.I. Manfred Lindner

Password-based Authentication 2

– password guessing (on-line)
  - on-line attack could prevented by slow down and audit
    – e.g. lockout after 3 unsuccessful attempts (ATM machine)
    – auditing invalid password attempts and trace the connection

– password choosing for preventing guessing (on-line)
  - if text string is randomly chosen from (a,.. z) then 8 character password is necessary
  - 8 character randomly chosen would be sufficient but people hate them, forget them and write them down!
  - 10 character computer generated pronounceable string is about as secure
  - if user choose own passwords enforce that they choice good ones
  - best for combining remembering and difficulty is a pass-phrase
    – e.g. Mhall;Ifwwas

Password-based Authentication 3

– password guessing (off-line)
  - for an off-line attack an arbitrary amount of time might be used
    – e.g. stealing password database (even when encrypted or hashed)
    – the intruder can first guess the password, second perform the same function on it and compared it with the stolen quantity encrypted by the original password
    – “dictionary attack”
  - apply “salt” (random number) to password before hash to slow down

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Password-based Authentication 4

– password choosing for preventing guessing (off-line)
  - 64-bit of randomness necessary
    – it is computationally infeasible to try out 2^{64} possibilities
  - if text string is randomly chosen from (a,.. z, A,.. Z, 0,1,2,..9, plus punctuation marks) then 11 character password is necessary
  - 11 character would be sufficient but people hate them, forget them and write them down!
  - 16 character computer generated pronounceable string could be about as secure
  - 32 character user generated pronounceable string could be about as secure
  - but that is definitely to long

Address-based Authentication 1

- **Identity of source**
  – is based on the network address from which packets arrives
- **Mainly designed to avoid eavesdropping of passwords**
  – since no password is sent through the network when using proxies
- **Used by remote execution tools**
  – UNIX Berkley rtools
    - /etc/hosts.equiv. for global control, .rhosts for per-user control
- **Intruder can jump from one machine to another**

Address-based Authentication 2

- **Network address impersonation**
  - even MAC addresses can be changed from software!
  - screening filters might restrict it
  - IP source routing should be disabled to make impersonation more difficult
- **Network address translation (NAT)**
  - same address might be used for many objects
- **Only use address-based scheme as a raw first step, do not rely on it alone**

Cryptographic Authentication Protocols 1

- **The general model of all authentication protocols**
  - an initiating user (a process), Alice, wants to establish a secure connection with a second user, Bob
  - Alice starts by sending a message to either Bob or to a trusted key distribution center (KDC), which is always honest
  - several messages are exchanged in follow
  - an intruder, Trudy, may intercept, modify or replay these messages in order to trick Alice and Bob
  - after the protocol has been completed, Alice is sure she is talking to Bob and vice versa

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

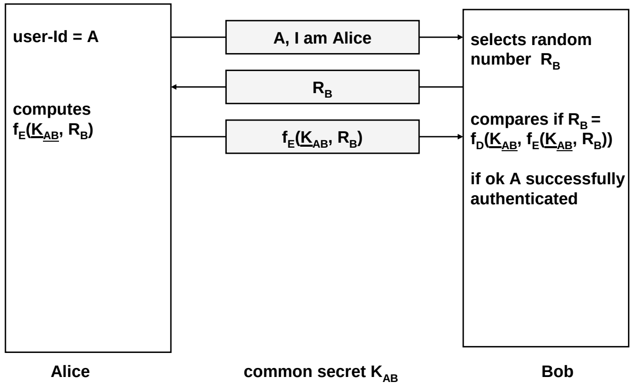
Cryptographic Authentication Protocols 2

- **The general model (cont.)**
  - in most protocols Alice and Bob will also have established a session key for use in upcoming conversation
    - privacy aspect by encryption
  - this session key is used for secret-key encryption
    - secret-key because of performance reasons
  - reason for using a new randomly chosen session key for each new connection
    - minimize amount of traffic that gets sent encrypted with that key
    - reduce amount of ciphertext an intruder can obtain
    - reduce the risk when a key falls in wrong hands
      - during the conversation only the session key should be present in a system, all other information (permanent keys, passwords) should be carefully zeroed out after session established

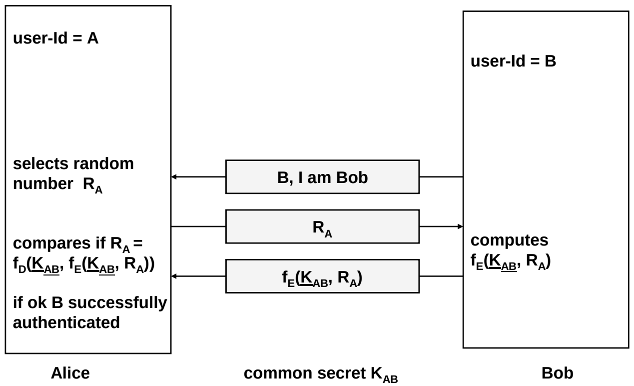
Authentication (Secret-Key) 1

- **Principle**
  - Alice and Bob share a secret-key $\mathbf{K}_{AB}$
  - one party sends a random number (so called nonces = number used only once) to the other which is then encrypted and the result is sent back
  - result is tested and compared with the sent random number, if equal then other party is authenticated
  - **Challenge - Response Technique**
  - A, B are the user-ID´s of Alice and Bob
    - used to select the right key if more than one security associations are possible
  - Random numbers are chosen from a large space
    - it is very unlikely that Trudy would have seen R_B or R_A and the corresponding responses from an earlier session (replay attack)

Authentication One-Way (Secret-Key) 2

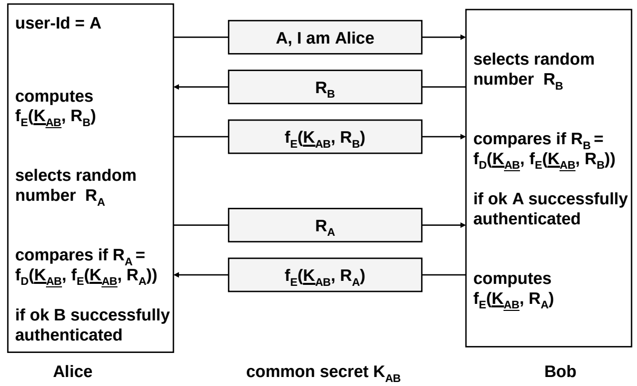


Authentication One-Way (Secret-Key) 3



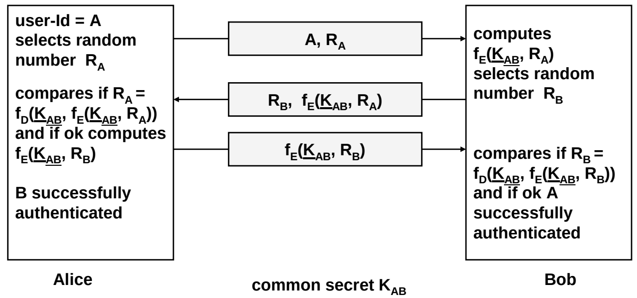
© 2009, D.I. Manfred Lindner

Authentication Mutual (Secret-Key) 4



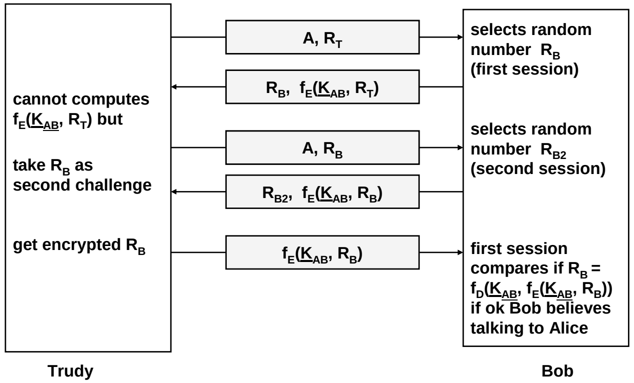
Authentication Mutual (Secret-Key) 5

Shorter (3 messages) but there is a <u>security pitfall known as reflection attack</u>



© 2009, D.I. Manfred Lindner

Reflection Attack (Secret-Key)



Morals of the Reflection Attack

- **Designing a correct authentication protocol is harder than it looks at first sight**
- **Three general rules**
  - the initiator should be the first to prove its identity
  - the initiator and the responder should use different keys for proof
    - e.g. two shared secret-keys K_{AB} and K'_{AB}
  - the initiator and responder take their challenges from different sets
    - e.g. the initiator must use even numbers and the responder must odd numbers or the own user-ID is concatenated with the random number before encryption (in the later case Trudy would need to get Bob to encrypt the user-ID Alice concatenated with some number to fool him)

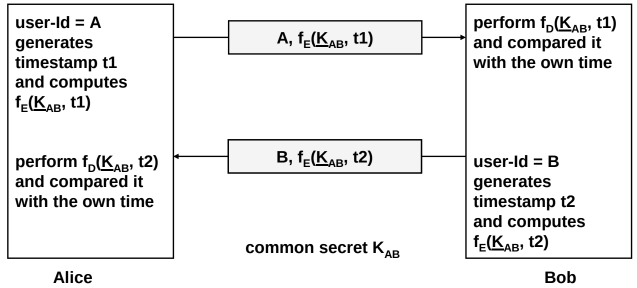
© 2009, D.I. Manfred Lindner

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Authentication (Timestamps, Secret-Key)

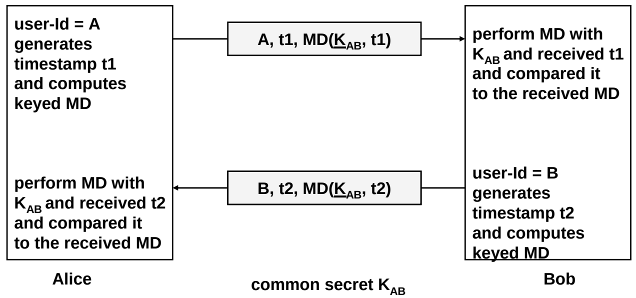
Synchronized clocks and remembering all timestamps sent by Alice (or Bob) within the acceptable clock skew needed to overcome a replay attack.



Encryption of current time

Authentication (Keyed-Message Digest)

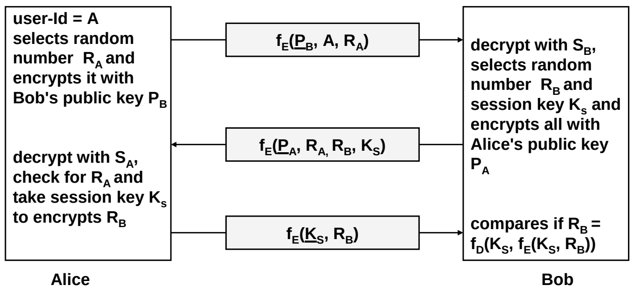
Synchronized clocks and remembering all timestamps sent by Alice (or Bob) within the acceptable clock skew needed to overcome a replay attack. Timestamps are needed in cleartext (MD is not reversible and checking would mean trying out all possible values within the acceptable clock skew).



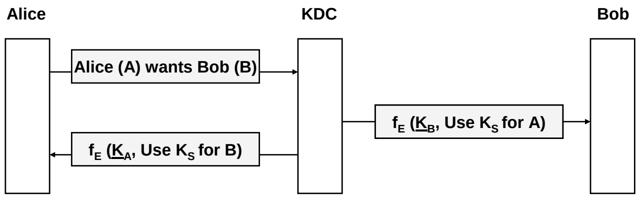
© 2009, D.I. Manfred Lindner

Authentication (Public-Key)

assumption: public keys were safely distributed



Authentication with KDC (Basic) 1



Key Distribution Centre (KDC) as a trusted third party stores a secret-key per user (K_A for Alice and K_B for Bob)

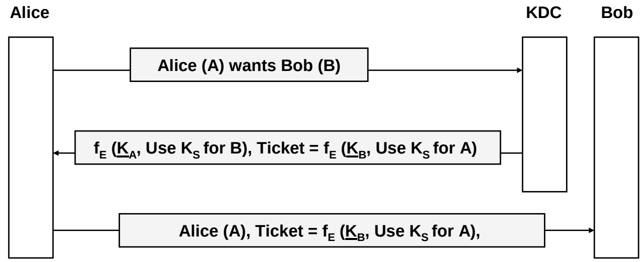
KDC selects a session key K_S and sent it encrypted with K_A to Alice

KDC forwards User-ID A and K_S encrypted with K_B to Bob

<u>Now Alice and Bob must perform mutual authentication with obtained K_S</u>

© 2009, D.I. Manfred Lindner

Authentication with KDC (Ticket) 2



KDC selects a session key K_S sent it encrypted with K_A to Alice together with a so called <u>Ticket</u>

Ticket itself contains session key K_S and user A both encrypted with K_B

<u>Now Alice and Bob must perform mutual authentication with obtained K_S</u>

Agenda

- **MAC, Hash and Message Digest**
- **Authentication**
- **<u>Digital Signature</u>**
- **Summary**

© 2009, D.I. Manfred Lindner

Digital Signature

- **The authenticity**
  - of many legal and financial documents is given by the presence or absence of an authorized handwritten signature
- **In modern e-commerce business**
  - we need something like a handwritten signature

<u>digital signature</u>

  - authentication and non-repudiation is far more important than secrecy (privacy) for e-business

Requirements of Digital Signature System

- **Authentication & Integrity**
  - the receiver can verify the claimed identity of the sender
  - the message can not be changed during transport by an intruder without recognition
- **Non-Repudiation**
  - the sender cannot later repudiate the contents of the message
  - protection of the receiver of a signed message
- **Integrity**
  - the message can not be changed by the receiver
  - protection of the sender of a signed message

© 2009, D.I. Manfred Lindner

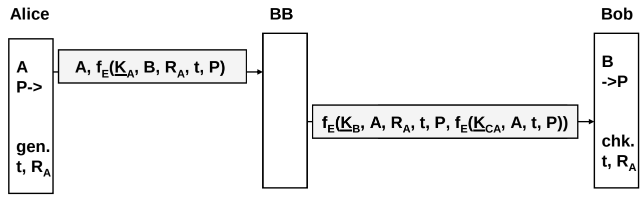
Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Secret-Key Signatures 1

- **One method**
  - central authority (BB) that knows everything and whom everyone trusts
  - each user deposits his secret-key at the central authority
  - messages from one user to the other user will pass the central authority which decrypts and encrypts accordingly based on the stored secret keys of the users
  - Plaintext message P together with timestamp t and random number R will be encrypted with the appropriate secret-key
    - A stands for user-ID Alice, B stands for user-ID Bob
    - timestamps used to prevent replay of old messages
    - random numbers used to prevent replay of fresh messages

Secret-Key Signatures 2



BB as a trusted third party stores a secret-key/user (K_A for Alice and K_B for Bob) Alice sends message P encrypted with her secret-key

BB signs the message (A, t, P) with secret-key K_{CA} which may be used later at court to prove Alice had really sent the message (actually done by decryption action of BB itself on request of the court). Note: Bob could sent such a message to himself using K_B.

BB forwards the message encrypted with Bobs secret-key

© 2009, D.I. Manfred Lindner

Secret-Key Signatures 3

- **timestamps and random numbers are used to prevent replay attack done by Trudy**
  - very old messages are rejected based on timestamp
  - Bob can check all recent messages to see if R_A is used in any of them -> if yes -> it can be discarded as replay
- **Big Brother problem of BB (can read anything)**

Public-Key Signatures 1

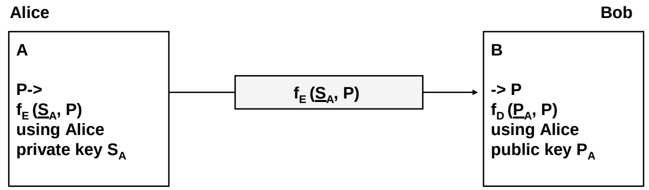
- **Big Brother problem**
  - of trusted authority can be avoided by public-key signatures
    - they have the property that $f_D(f_E(M)) = M$ and $f_E(f_D(M)) = M$
- **Method with RSA**
  - instead of
    - $M = f_D(\underline{S}_B, f_E(\underline{P}_B, M))$ for privacy
    - only Bob can decrypt message from Alice
  - we use
    - $M = f_D(\underline{P}_A, f_E(\underline{S}_A, M))$ for signature
    - only Alice could have encrypt (signed) this message, everybody knowing the public key can verify this

© 2009, D.I. Manfred Lindner

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

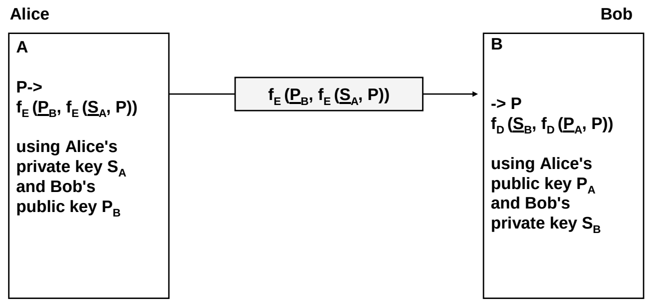
Public-Key Signatures 2



Note:
Message from Alice to Bob can be decrypted by anyone who has the public key of Alice -> our main aspect is authentication

We can additionally apply encryption with Bobs public key to combine both aspects: privacy and authentication (seen on next slide)

Public-Key Signatures 3



Note:
Only Alice will have the private key and nobody else, so at court she can not deny having sent this message

© 2009, D.I. Manfred Lindner

Public-Key Signatures 4

- **But Public-key techniques are too slow for large messages**
  - perform public-key algorithms on short message-digest (MD) of a message instead on large message itself

Public-Key Signatures 5

- **Therefore**
  - take MD and encrypt it based on <u>public-keys algorithm</u>
    - remember: MD can be computed much faster and allows a fingerprint of the message
    - the private key is used for signing
  - we call that

    “<u>Digital signature” (DS)</u>

- **Sometimes a separation of authentication and privacy is necessary or wanted**
  - authentication done based on the DS
    - also allows integrity checking, non-repudiation and if combined with timestamps prevent replay attacks
  - privacy may be done by secret-key encryption which is faster

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Public-Key Signatures 6

Alice / **Bob**

A

P-> MD(P)

encrypt MD
$f_E(\underline{S_A}$, MD(P), t) = DS
using Alice's
private key S_A

→ **P, $f_E(\underline{S_A}$, MD(P), t) = DS** →

B

-> P
compute own MD(P)

decrypt DS
$f_D(\underline{P_A}$, DS) = rcv. MD(P)
using Alice's
public key P_A

compare to prove
own MD(P) = rcv. MD(P)

Note:
DS proves authentication, because only Alice could have produced such MD.
DS allows integrity checking, too. It is impossible to change P and create a valid MD because only Alice knows the private-key. Timestamps to prevent replay attack.

Public-Key Signatures 7

- **Encrypting with private-key provides**
  - Digital Signature (DS) of the original message
  - <u>non-repudiation</u> of the message
    - Only one partner possesses the private key in contrast to secret-keys whereas all partners share the same key !!!
- **Remaining problems**
  - proof holds as long the private-key is kept secret
  - What happens when the private-key is stolen?
  - What happens if Alice changes the private key later?
  - How can we exchange public-keys in a secure way?

Public-Key Signatures 8

- **In order to use public-key signatures and to solve these problems again we need some "trusted" authority**
  - where key changes and the dates of change are recorded
  - where public-keys can be deposited and signed
  - where public-keys can be revocated
    - similar to revocation list of credit cards
- **We call such a "trusted" authority CA**

  Certification Authority
- **Modern signature systems are based on it**
  - PKI (Public Key Interchange)
  - DSS (Digital Signature Standard)

Public-Key Certificates by CA 1

- **Example with digital certificates between Alice and Bob**
  - Bob has signed his public-key P_{Bob} by CA and holds an certificate DC of his key
    - f_E (S_{Cert} , P_{Bob}) is the Digital Certificate (DC) of P_{Bob}
  - P_{Cert} = public-key of certificate authority CA must be configured manually or included in application SW in end system of the checking system (Alice)
  - To be checked system (Bob) sends its public key signed by Certificate Authority
    - P_{Bob} + f_E(S_{Cert} , P_{Bob})
  - Alice verifies
    - If f_D (P_{Cert} , DC (P_{Bob})) = received P_{Bob}

© 2009, D.I. Manfred Lindner

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Public-Key Certificates by CA 2

- Now Alice can send a shared symmetric key K_{AB} to Bob
  - By using his validated received public key P_{Bob}
  - Only Bob can decrypt the shared symmetric key K_{AB}
- Traffic between Bob and Alice can now be encrypted by using the shared key K_{AB}
- This technique is used by SSL Handshake Protocol
  - Secure Socket Layer
    - SSL invented and introduced by Netscape
    - Layer between TCP and application
  - Included in WEB browser to perform encryption HTTP session
  - RFC version of SSL is called Transport Layer Security (TLS)

Public-Key and Secret-Key Management

- **How to be sure that a key is from the right person?**
  - Man-in-the-middle attack problem
- **Methods to solve that problem:**
  - Manual outband verification
  - Trusted Third Parties
    - Usage of public-key certificates
    - Certificate Authority (CA)
    - Many public-keys can be verified by usage of only one trusted public-key of a certificate authority
  - ISAKMP (Internet Security Association and KeyMgtProt)
    - RFC 2408 for all layers of TCP/IP stack - framework
  - IKE (Internet Key Exchange Protocol)
    - RFC 2409 for IPsec

© 2009, D.I. Manfred Lindner

Agenda

- MAC, Hash and Message Digest
- Authentication
- Digital Signature
- Summary

Summary of Privacy Methods

Encryption	Decryption	Method
$f_E(\underline{K}_{AB}$, Text)	$f_D(\underline{K}_{AB}$, Text) = Text	secret-key (symmetric) Alice <-> Bob shared key K_{AB}
$f_E(\underline{P}_B$, Text)	$f_D(\underline{S}_B$, Text) = Text	public-key (asymmetric) Alice -> Bob Bob's pubic key P_B Bob's private key S_B
$f_E(\underline{P}_A$, Text)	$f_D(\underline{S}_A$, Text) = Text	public-key (asymmetric) Bob ->Alice Alice's pubic key P_A Alice's private key S_A

Institute of Computer Technology - Vienna University of Technology

L95 - Integrity and Authentication

Summary of Authentication (Identity) and Integrity Methods

Function	Verify	Name / Method
Text + $f_E(\underline{K}_{AB}$, Text) = Text + MAC	com_MAC = rcv_MAC	MAC / secret-key *)
Text + H(Text)	com_Hash = rcv_Hash	Fingerprint, no security / hash
Text + $H(\underline{K}_{AB}$, Text) = Text + HMAC	com_HMAC =rcv_HMAC	HMAC / keyed hash based on secret-key algorithm *)
$f_E(\underline{S}_B$, Text)	$f_D(\underline{P}_B$, Text) = meaningful	Identity + NR of Bob -> everybody can read/ public-key algorithm **)
Text + $f_E(\underline{S}_B$, H(Text)) = Text + DS	com_H(Text) = $f_D(\underline{P}_B$, DS)	Digital Signature of Bob Identity + Integrity + NR / keyed fingerprint based on public-key algorithm **)

Authentication *) partners trust each other **) partners don't trust each other

Summary of Certification Method

Function	Name / Method
(P_B+Bob) + $f_E(\underline{S}_{CERT}$, H(P_B + Bob)) = Certificate + DS of CA	Certificate of Bob's Public Key P_B / hash signed with Digital Signature of trusted third party -> Certification Authority (CA) S_{CERT} = private-key of CA
Verify	
hash of Certificate (= H(P_B+Bob)) = $f_D(\underline{P}_{CERT}$, DS of CA)	P_{CERT} = public-key of CA