Identification & Authentication

part 2

Stallings: *Chapter 3, 22*
Lidinsky: *Much Supplementary Material*

A Practical Password Scheme

Scheme Pros & Cons

Pros

- *Easy to remember*
  - Allows you to write down partial passwords
- *Allows you to have different passwords to access different target systems*
- *Reasonably secure*
- *Allows changing passwords without too much trouble*

Cons

- *Not super secure*

The Components

Passpart

Possibly easy to associate with you or the system that you want to access

Possibly easy to guess

Personal Code

Secret

Algorithm for Applying Your Personal Code

Secret

Example
Passpart + Personal Code

Write down list of passparts related to with whom you're dealing

e.g., `Bank1` `amex` `blackboard`

Remember personal code (secret)

e.g., `!0P?x` `[cG}+` `^9m@/`

Remember how to apply it (secret)

e.g., At beginning & after every 3rd letter of the passpart

Passwords

`!Ban0k1P?x` `[ameVxG}+` `^Bla9ckbmoar@d/`

Store & Remember

- Store ***passparts*** for your reference
  - *e.g., Keep them in a secure password locker in Windows*
    - Encrypt using Windows standard encryption and restrict access
    - Maybe not too good but better than nothing
  - *e.g., Keep them on your smart phone or PDA but don't leave it lying around*
    - Encrypt and restrict access to them in the PDA software on your PC
- Memorize
  - **Personal Code**
  - **Algorithm** *for applying the Personal Code*

S/Key

S/Key

S/Key is a one-time pad for passwords

RFC1760

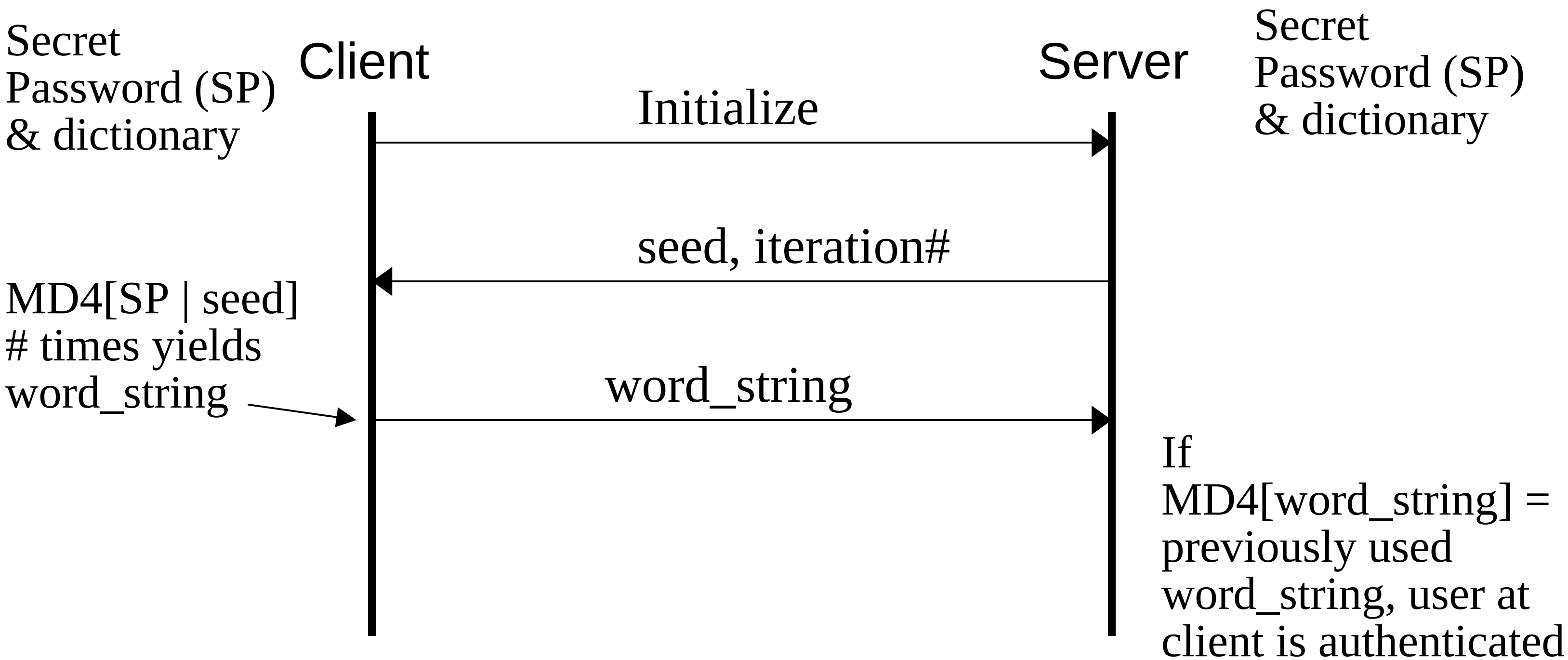
Bellcore defined it

Uses MD4 or MD5 hashing

Purpose

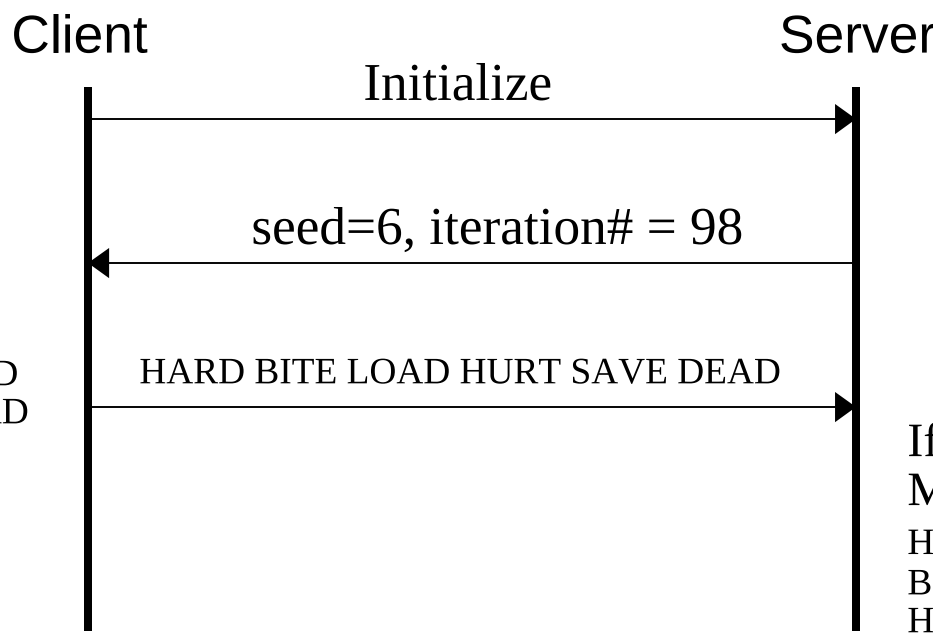
- *To make login secure*
  - Thus preventing hacker from eavesdropping, getting the login and password, and then logging in later as the legitimate user
- *Does not encrypt passwords*
  - Sends them as plaintext; “in the clear”

S/Key Operation



S/Key Operation Example

Secret
Password (SP)
& dictionary



Secret
Password (SP)
& dictionary &
previous string:
BOAT HIDE LOVE
HOME HELP WHAT

MD4[SP | 6]
98 times yields
HARD BITE LOAD
HURT SAVE DEAD

If
MD4[HARD BITE LOAD
HURT SAVE DEAD] =
BOAT HIDE LOVE
HOME HELP WHAT
user at
client is authenticated

Each *6-word string* is used only once

Thus S/Key is a one-time pad for authentication

- *Guarantees that each 6-word string is unique*
- *Iteration number <u>decreases</u> by <u>one</u> each time a client uses S/Key*

S/Key Manual Operation
Code Book

Often users carry a small code book with perhaps 100 to 200 word strings

Prepared in advance by the S/Key server

Each time the user logs into the server, she uses the next word string from the code book

e.g., HARD BITE LOAD HURT SAVE DEAD

The server does the hash and if the sequence is correct, authenticates the user

Partial Code Book Example

...OPEN JOBS STOP KILL MEAN RATS
BOAT HIDE LOVE HOME HELP WHAT
HARD BITE LOAD HURT SAVE DEAD
LEER DENT QUIT FREE ZOOS ALSO
...

S/Key Manual Operation Example

User

Any Client

Server

Secret password (SP) & dictionary & previous string:
BOAT HIDE LOVE HOME HELP WHAT

Secret Code Book

...~~OPEN JOBS STOP KILL MEAN RATS~~
~~BOAT HIDE LOVE HOME HELP WHAT~~
HARD BITE LOAD HURT SAVE DEAD
LEER DENT QUIT FREE ZOOS ALSO
...

HARD BITE LOAD HURT SAVE DEAD

If MD4[HARD BITE LOAD HURT SAVE DEAD] = BOAT HIDE LOVE HOME HELP WHAT user at client is authenticated

S/Key Comments

S/Key is a one-time pad for authentication

Dictionary is a list of short words arranged as word strings with each word coded into a single binary number

e.g., 2048 words each coded as an 11-bit binary number

Once the word_string is used, it cannot be used again

When the iteration# reaches zero or the code book is exhausted, S/Key is inoperative until

The client and server are refreshed, or

The user gets a new Code Book

Good password security achieved because once user logs off, the password (word_string) is no longer valid

PPP Authentication Protocols

Overview

There are a number of user authentication protocols

PAP	*MS-CHAPv1*
SPAP	*MS-CHAPv2*
CHAP	***EAP***

These protocols authenticate but don't encrypt

Here we will consider three of these protocols that are used with the PPP (Point-to-Point Protocol)

PAP	*Password Authentication Protocol*
CHAP	*Challenge-Handshake Authentication Protocol*
EAP	*Extensible Authentication Protocol*

Overview

PPP is used for communication between a client and server over a single link such as dial-up, ISDN, or DSL

- *Exactly two endpoints*

For all these protocols (e.g., PAP, CHAP) the **client** computer (not the user) is authenticated

Windows 2K and later supports all of these protocols

These protocols can be implemented in

- *A network access server (NAS) or*
- *A separate authentication system such as RADIUS*

PPP

We need to understand PPP before we consider user authentication protocols that use PPP

Point-to-Point Protocol

(Briefly)

PPP Overview

PPP consists of a link layer (LCP - layer 2) and a network layer (NCP - layer 3)

NCPs (Network Control Protocols)

Negotiates various configuration parameters

e.g., data compression, IP address negotiation

LCP (Link Control Protocol)

e.g., data encapsulation, network layer protocol muxing, error detection

After PPP link is set up, PPP provides an optional authentication phase before the NCP phase

PPP Frame Format

PPP uses the same **frame** format as HDLC, SDLC, LAPB and ADDCP.

Flag 01111110	Address 11111111	Control 00000011	Protocol	Data	FCS	Flag 01111110
1	1	1	2	0 - 1500	2	1

PPP Frame Format

Flag

Delimits the beginning and end of the frame.

Address

Broadcast. (point-to-point protocols have no need for addresses.)

Control

The bit pattern indicates that the frames are "unsequenced".

PPP Frame Format

Flag 01111110	Address 11111111	Control 00000011	Protocol	Data	FCS	Flag 01111110
1	1	1	2	0 - 1500	2 or 4	1

Protocol

Indicates the protocol that is being transmitted in the Data field

Similar function (but not the same as) the number that is in the Type field of the Ethernet header or the Type field in the SNAP protocol related to 802.3

For PPP, IPv4 = 0x0021; IPv6 = 0x0057...

Data

The protocol and its payload. Bit-oriented.

FCS (Frame Check Sequence)

A CRC over the entire frame sans Flag fields

Used to detect errors

Default is 2 bytes. Can be negotiated to be 4 bytes.

PPP Frame Format*
Lab

Flag 01111110	Address 11111111	Control 00000011	Protocol	Data	FCS	Flag 01111110
1	1	1	2	0 - 1500	2 or 4	1

The data field is bit-oriented

Suppose a string of 8 bits in the data field contained the string “01111110”. What would happen?

The receiver would erroneously decide this is the flag field indicating the end of the frame.

Questions:

How might the receiver detect the above error?

How can data containing a bit sub string of “01111110” be successfully received?

PAP

Password Authentication Protocol

PAP

Password Authentication Protocol

Two-way handshake just after link is established

Authenticates Peer system; <u>not</u> <u>the</u> <u>user</u>.

The Peer system is not necessarily the user's host

- *Sometimes a local modem for phone, DSL, or cable*
- *Sometimes a gateway router*

PAP
Password Authentication Protocol

1	1	1	2	0 - 1500	2 or 4	1
Flag 01111110	Address 11111111	Control 00000011	Protocol **PAP=0xC023**	Data	FCS	Flag 01111110

Type	ID	Length	PAP Data
1	1	2	>= 0

<u>Type</u>

Request

1: Authentication Request

Replies

2: Authenticate ACK

3: Authenticate NAK

<u>ID</u>

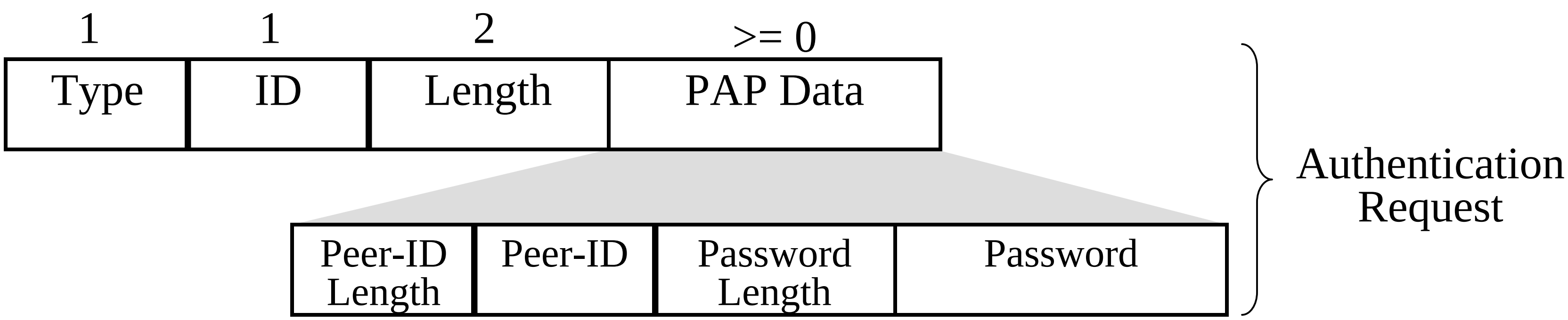
Set in request. Returned in reply. Matches request to reply.

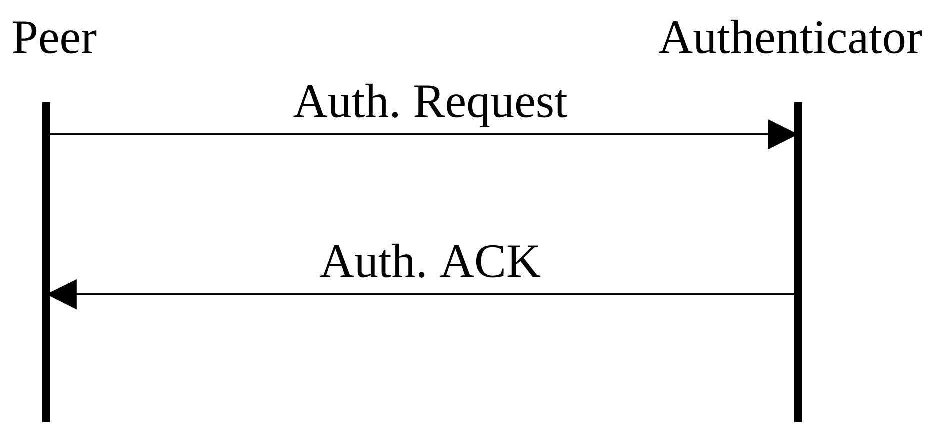
<u>Length</u>

Length of entire PAP protocol

Type, ID, Length, and PAP Data fields included

PAP
Authentication





PAP
Comments

Name and password sent as plaintext

Exhaustive attempts can be used

- *There is no limit on number of attempts*
  - The Peer decides when and how often to try

Authenticates Peer system; not user

Used primarily when server requires a plaintext password

CHAP

Challenge Handshake Authentication Protocol

CHAP
Challenge-Handshake Authentication Protocol

Three-way handshake

Used at time of link establishment

Can also be used repeatedly any time after link establishment

CHAP

Challenge-Handshake Authentication Protocol

1	1	1	2	0 - 1500	2 or 4	1
Flag 01111110	Address 11111111	Control 00000011	Protocol CHAP=0xC223	Data	FCS	Flag 01111110

Type	ID	Length	CHAP Data
1	1	2	>= 0

Type

1: Challenge

2: Response

3: Success

4: Failure

ID

Set in Challenge.

Returned in reply. Matches request to reply.

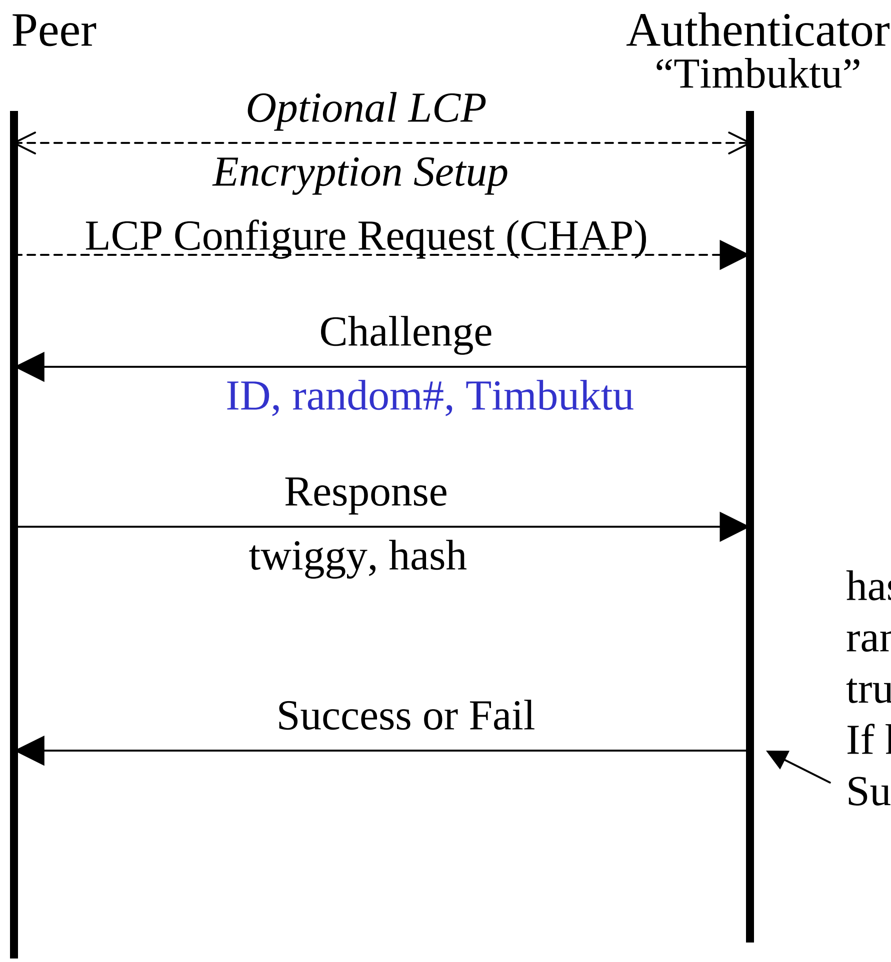
Length

Length of entire CHAP protocol

Type, ID, Length, and CHAP Data fields

CHAP
Challenge-Handshake Authentication Protocol

Peer	Passwd
twiggy	trustme



hash = MD5[ID, random#, Timbuktu, trustme]. Send

Database of Peers

Peers	Passwds
twiggy	trustme
Bill	foo!3%
...	...

hash' = MD5[ID, random#,Timbuktu, trustme]
If hash = hash’, send Success. Else send Fail.

CHAP
Comments

CHAP is a commonly used protocol for authentication

Uses hashed passwords for greater security than PAP

However, the passwords must be stored in server

- *Ether as plaintext, or*
- *As reversibly encrypted passwords*

Variations on CHAP

MS-CHAPv1

- *Proprietary*
- *Similar to CHAP*
- *Passwords stored on server in encrypted format*
  - Can use available irreversibly encrypted password databases

MS-CHAPv2

- *Proprietary*
- *Similar to MS-CHAP Version 1 but:*
  - Requires mutual authentication
  - Different encryption keys when sending and receiving
  - Thus more secure than MS-CHAP v. 1

EAP

Extensible Authentication Protocol

EAP
Extensible Authentication Protocol

- Designed as an extension to PPP to be able to use newer authentication methods
  - *Such as one-time passwords, smart cards, or biometric techniques*
- EAP postpones the authentication phase
  - *Allows authenticator to request additional information before deciding on the authentication mechanism to use*
- Separates authentication from the PPP protocol
  - *Permits the use of a "back end" authentication server*
    - The PPP server acts as a conduit between the client and the authentication server

EAP
Extensible Authentication Protocol

There are two different types of EAP, and both the server and client must be using the same type

EAP-MD5 CHAP

Used primarily for password-based security

EAP-TLS

Used primarily for certificate-based security

EAP

Extensible Authentication Protocol

1	1	1	2	0 - 1500	2 or 4	1
Flag 01111110	Address 11111111	Control 00000011	Protocol **EAP=0xC227**	Data	FCS	Flag 01111110

Request or Response:

Code 1 or 2	ID	Length	Type	Type_Data
1	1	2	1	>= 0

Success or Failure:

Code 3 or 4	ID	Length = 4
1	1	2

Code

1: Request *2: Response*

3: Success *4: Failure*

ID

Set in Request.

Returned in Response. Matches request to Response.

Length

Length of entire EAP protocol

Code, ID, Length, Type, Type_Data

Type

Type of Request or Response (next slide)

Type_Data

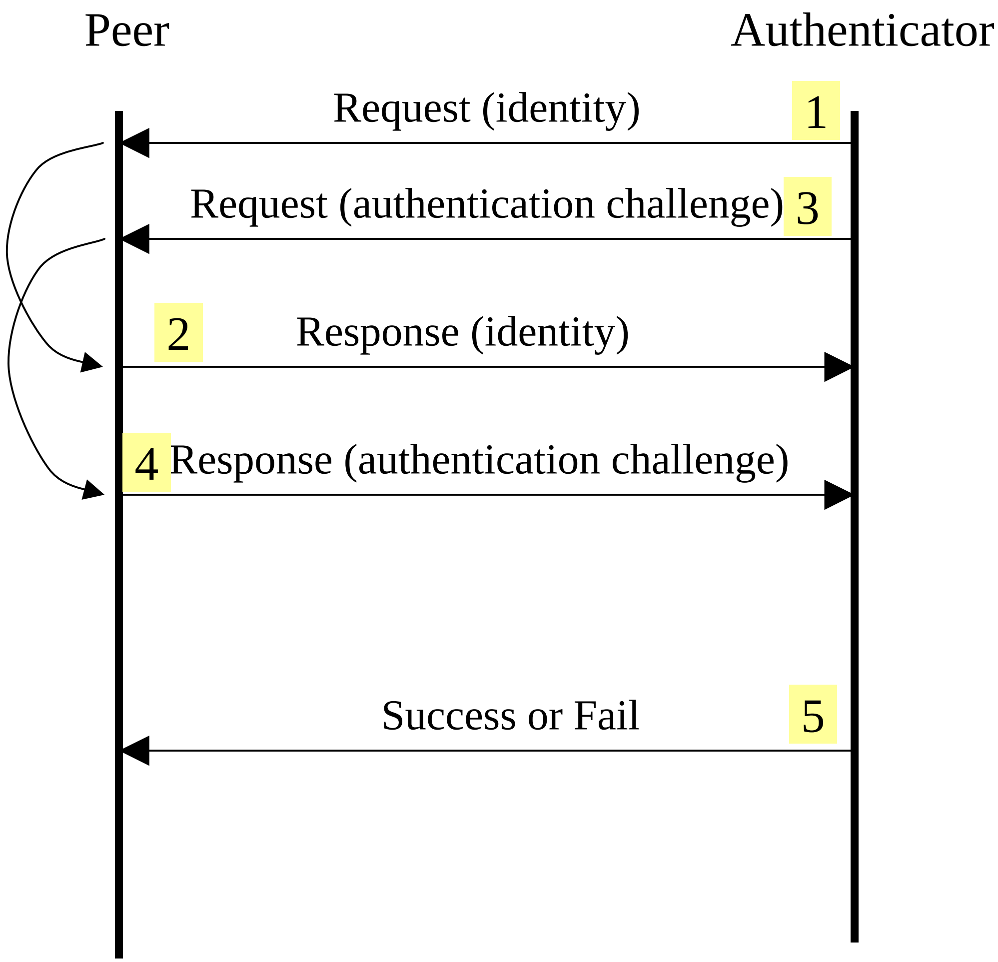
Varies with the Type Field

EAP
Extensible Authentication Protocol

Type of Request or Response

- *1: Identity*
- *2: Notification*
  - Used to send a displayable message to the peer
- *3: Nak (Response only)*
- *4: MD5-Challenge*
  - Similar to CHAP
- *5: One-Time Password (OTP) (RFC 1938)*
  - Similar to S/Key
- *6: Generic Token Card*
  - Similar to Challenge-Response

EAP
Extensible Authentication Protocol



Details of Request, Challenge, and Response are based upon authentication scheme used.

EAP
Advantages

- Supports multiple authentication mechanisms without having to pre-negotiate a particular one during LCP Phase.
- Devices such as a NAS (Network Access Server) need not understand each request type
  - *Can simply act as a passthrough agent for a "back-end" access server on a host.*
  - *The NAS only need look for the success/failure code from the access server to terminate the authentication phase.*

EAP
Disadvantages

EAP does require the addition of a new authentication type to LCP

Thus legacy PPP implementations will need to be modified to use it.

EAP strays from the previous PPP authentication model of negotiating a specific authentication mechanism during the LCP phase.

Third Party Authentication Systems

Overview

Third Party Authentication Systems

TACACS+

RADIUS

Kerberos

TACACS+ and RADIUS

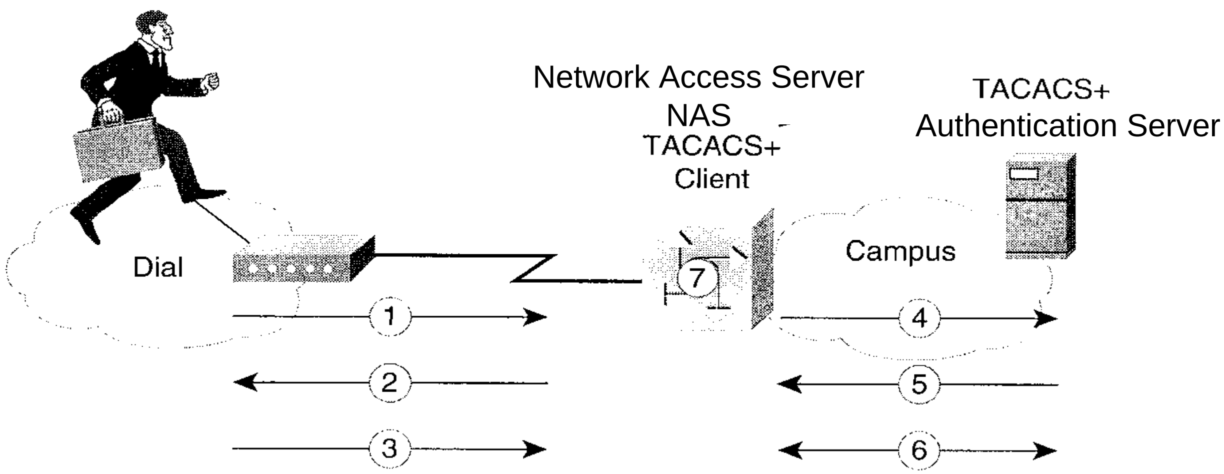
TACACS+

Used where there Network Access Server (NAS) is separated from the Authentication Server

Thus there are three participants

- *User*
- *NAS (Network Access Server)*
  - Considered the TACACS+ client
- *TACACS+ Server (Authentication Server)*
  - Passive open on 49/tcp

TACACS+ Scenario



1) User initiates PPP authentication on NAS.

2) NAS prompts user for username/password (PAP) or challenge (CHAP).

3) User replies.

4) TACACS+ client sends encrypted packet to TACACS+ server.

5) TACACS+ server responds with authentication result.

6) TACACS+ client and server exchange authorization requests and replies.

7) TACACS+ client acts upon authorization exchange.

RADIUS

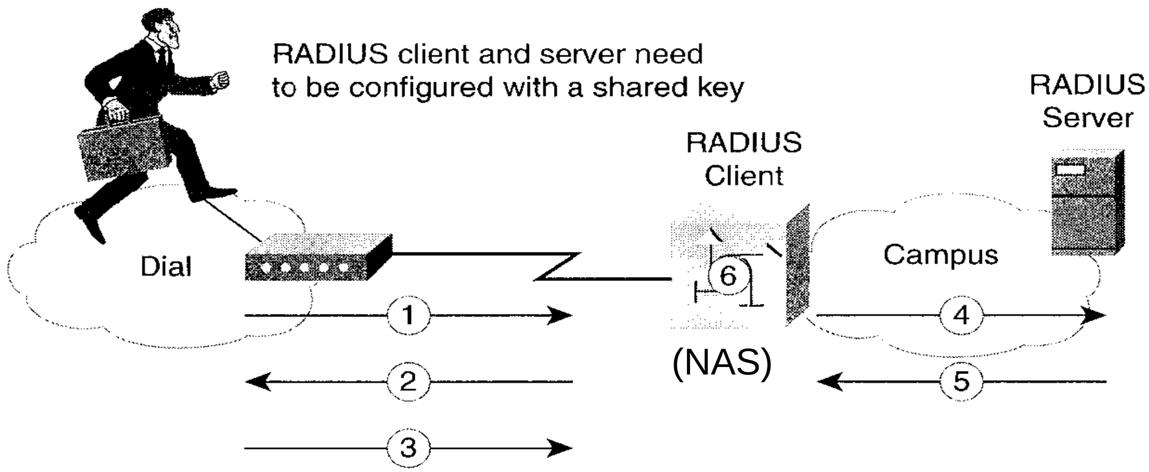
Similar in concept to TACACS+

- *Used where there NAS is separated from the authentication server*

Again there are three participants

- *User*
- *NAS*
    - Considered the RADIUS client
- *RADIUS Server*
    - Uses udp

RADIUS Scenario



1) User initiates PPP authentication to NAS.

2) NAS prompts user for username/password (PAP) or challenge (CHAP).

3) User replies.

4) RADIUS client sends username and encrypted password to RADIUS server.

5) RADIUS server responds with Accept, Reject, or Challenge.

6) RADIUS client acts upon services and service parameters bundled with Accept or Reject

Kerberos

Kerberos
History and Overview

Cross-platform

All versions of Windows from Win98 onward

All versions of Macintosh OS 8 onward

Linux	Solaris	HPUX
Irix	AIX	…

Developed by MIT

The name (Greek mythology)

Kerberos was the dog that guarded gates of Hell

Three heads	*Serpent's tail*
Mane of snakes	*Lion's claws*

Kerberos
History and Overview

In this lecture we'll try to do what Hercules is doing

Get a grip on Kerberos!

(In Latin it is Cerberus)

Kerberos
History and Overview

1987: Kerberos v4 designed, deployed at MIT in Project Athena

1990: Kerberos v5 design more or less complete

1991: Kerb v5 adopted by OSF/DCE

1992: Large-scale Kerb (4) deployments at universities

1993: RFC 1510, official v5 spec, published;
also RFC 1509 GSS-API v1 spec

1996: Kerb v5 1.0 implementation published by MIT

1997: Microsoft announces use of Kerberos for NT5;
also RFC 2222 SASL spec

1999: Windows 2000 ships with Kerberos support

Available on all Linux and Windows OSs from year 2000 onward

Kerberos
Concepts

Why do some organizations want to see your drivers license?

- *It serves as a verification from a trusted 3rd party that you are who you say you are.*
- *The 3rd party is the state that issued the license.*

This is exactly what Kerberos does

- *The trusted 3rd party is the KDC*

Kerberos
Concepts

Kerberos will verify that a user is who they claim to be

Kerberos is a centralized authentication system

Kerberos does **not** provide authorization

There exists a trusted 3rd party

- *Key Distribution Center (KDC)*

Short lifetime “tickets”

- *Tickets contain short lifetime keys + other info*

Names and passwords familiar to users

Never cache long term keys on clients

Kerberos
Concepts

Authentication Service

- *User logs in to Authentication Server*
- *The user is issued a ticket, which can be used to obtain tickets for services on other servers*

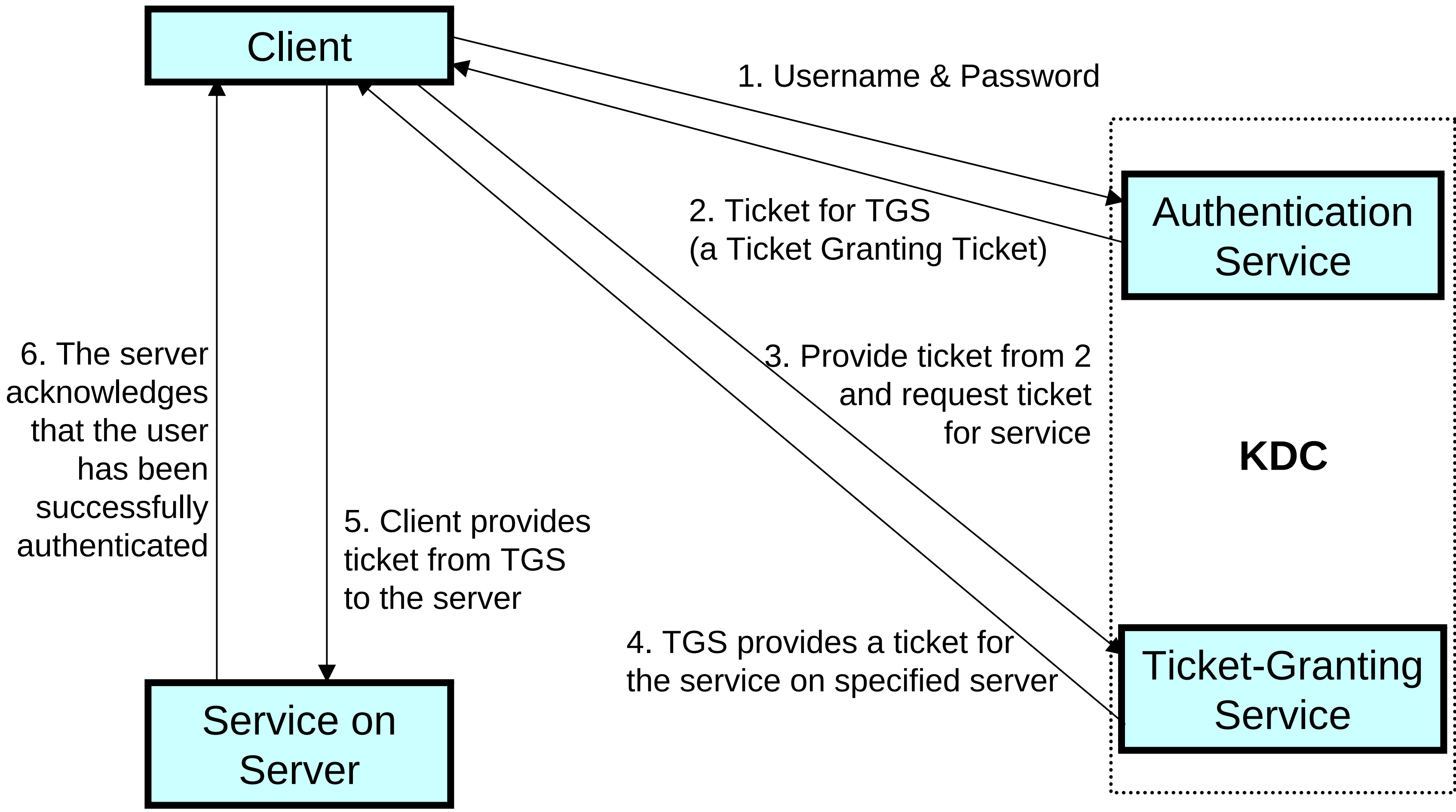
Ticket-Granting Service

- *This service provides tickets for services on other servers to users who are* ***already authenticated***

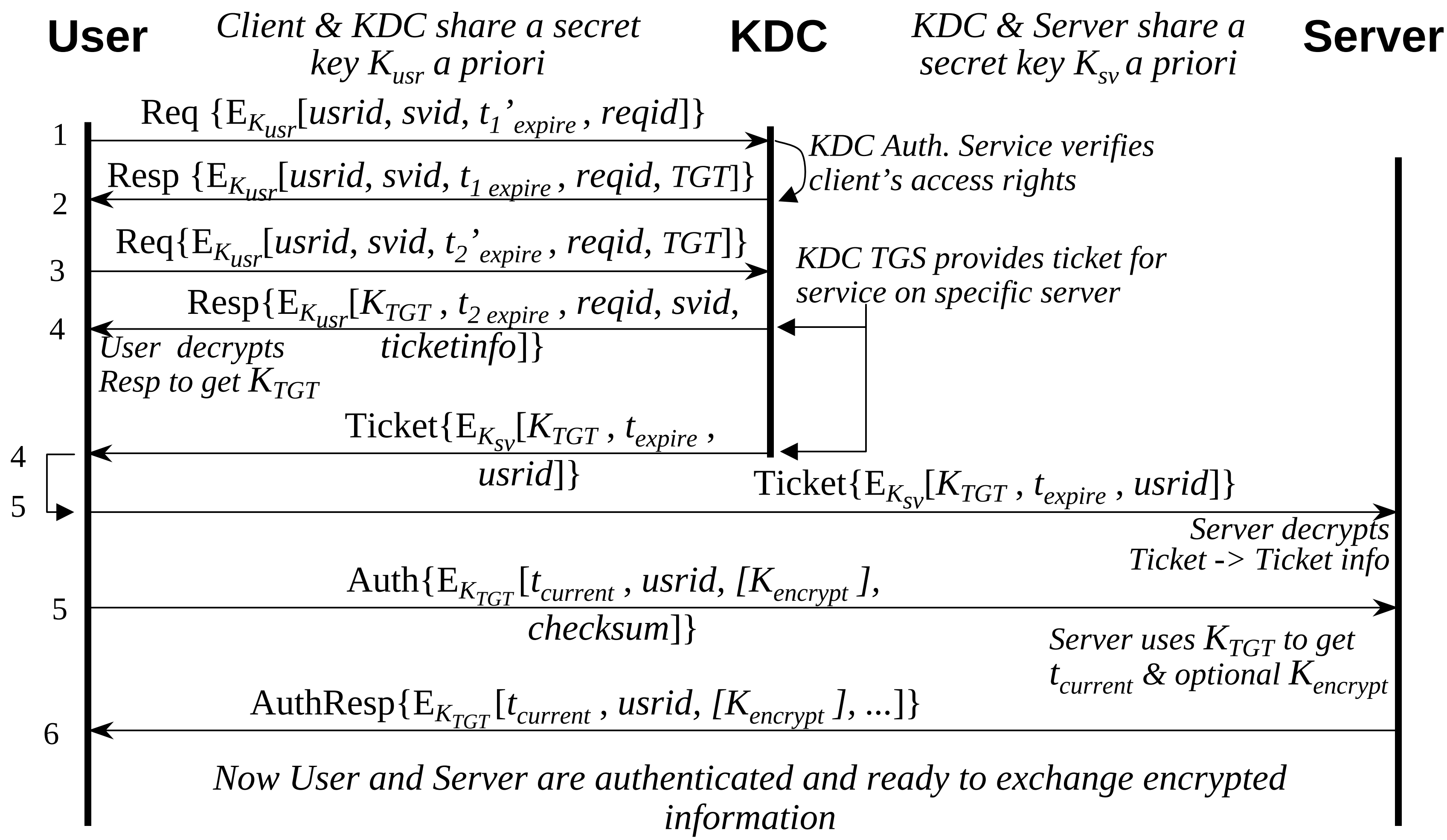
KDC is comprised of

- *Authentication Service*
- *Ticket-Granting Service*

Kerberos
Process



Kerberos
Server Authentication Process



Kerberos
Authentication

The checksum is used to initially authenticate the User to the Server

- *K_{TGT} from Ticket is used to decrypt the Auth message*
- *Server then calculates its own checksum' from the decrypted information in the Auth message*
- *If checksum' = checksum, the the User is authenticated to the Server*

But not quite!

Kerberos
Authentication

A malevolent hacker could intercept the Auth message and later replay it to the Server to impersonate the User

What can be done?

This is one of the reasons for the parameter $t_{current}$ *in the Auth message*

If the Auth message arrives within a narrow window (configurable and usually a couple of minutes) of the time $t_{current}$ *then the Auth is accepted*

If Auth arrives outside this window, it's rejected

Note that AuthResp authenticates the Server to the User

Kerberos
Single Sign-On

A problem with Kerberos was that a user had to establish a session with each server

- *The user's key or password (K_{usr}) had to be presented for each server that the user wanted to use*
- *This was cumbersome & presented a vulnerability*

It was desired that a user should be able to sign on (e.g., log in) once; not multiple times

But caching K_{usr} presents a vulnerability since the key K_{usr} is then available long term on the user's host

A key Kerberos concept is to only cache keys that work for short time periods

Kerberos
Single Sign-On

Solution: Use short term stand-in for K_{usr}

Upon initial user login

- *Client Sends a Req*
- *A Resp is returned containing:*
  - $K_{client\text{-}session}$ or K_{tgt}

The key K_{tgt} is used in all subsequent Kerberos negotiations (between User and KDC)

K_{tgt} usually is set to have a lifetime of 4-8 hours

K_{usr} is never cached

Kerberos
Realms

- Kerberos contains the concept of “realms”
  - *Windows domains are similar*
- KDCs are realm-specific
- $User_A$ in $Realm_A$ wishes to access $Server_B$ in $Realm_B$
  - *$User_A$ sends Req message to KDC_B*
  - *KDC_B authenticates $User_A$ in a query/response with KDC_A*
  - *KDC_B then completes the authentication with $User_A$*
  - *$User_A$ the interacts with $Server_B$*
- Cannot chain realms

Kerberos Weaknesses

Must keep password secret

Kerberos doesn't protect against password cracking

Kerberos is useless against DoS attacks. Why?

If a User system ceases to be authorized, it must be removed from the KDC

As long as the User is in the KDC as authorized, others can get tickets in the name of the User

The clocks of authenticating devices must be loosely synchronized

Assign13a

Read

Stallings: Chapter 3, 22

Problem Set: Stallings

Questions 3.1 – 3.9

This assignment is not to be submitted. But you WILL be responsible for it as regards the final exam.