# Authentication & Authorization

Anuj Gupta[1],
[1]M.Tech Scholar,
Department of C.F.I.S, G.I.T.A.M,
Kablana, Jhajjar

Ashish Kumar Sharma[2]
[2]Assistant Professor,
Department of C.F.I.S & C.S.E, G.I.T.A.M,
Kablana, Jhajjar

**This paper has taken a view about the basic security needs and there available solutions that are in used. Authentication and its other factors as well as authorization and its requirements are discussed. Both of these basic security requirments are needed to be taken into account in industry, corporates, defense and individual as well.**

***Index terms— Authentication, Authorization, Identity, E-Token.***

## I. INTRODUCTION

Authentication is used by a server when the server needs to know exactly who is accessing their information or site.

Authentication is used by a client when the client needs to know that the server is system it claims to be. In authentication, the user or computer has to prove its identity to the server or client. Usually, authentication by a server entails the use of a user name and password. Other ways to authenticate can be through cards, retina scans, voice recognition, and fingerprints. Authentication by a client usually involves the server giving a certificate to the client in which a trusted third party such as Verisign or Thawte states that the server belongs to the entity (such as a bank) that the client expects it to.

Authentication does not determine what tasks the individual can do or what files the individual can see. Authentication merely identifies and verifies who the person or system is.

## II. AUTHENTICATION OVERVIEW`

Authentication is based on factors Knowledge factors (something you know, like a password or passphrase) Possession factors (something you have, like a phone or other token) Inherence factors (something you are, like a fingerprint or other biometrics) Positive verification of identity (man or machine)

Verification of a person’s claimed identity:

- Who are you? Prove it.
- 3 Categories: What you know?

What you have? Who you are?

- What you know:
  - Password
  - Passphrase
  - PIN
- What you have
  - Digital authentication
  - Physical devices to aid authentication
  - Common examples:
  - e-Token
  - Smart cards
  - RFID

Multiple Form Factors and Authentication Methods

Single Authentication Management Server
USB Authenticators
Smart Cards
Hybrid (OTP/SC/Flash) Authenticators
OTP Authenticators
Software / Mobile Authenticators

Offering the broadest range of authenticators, from smart cards and tokens to mobile phone auth—all managed from a single platform

Authentication:

e-Token: May store credentials such as passwords, digital signatures and certificates, and private keys

Can offer on-board authentication and digital signing

Smart cards

Size of a credit card Usually an embedded microprocessor with computational and storage capabilities

Programmable platforms:

C/C++

Visual Basic

Java

.Net (beta)

Smart Cards cont’d

Contact vs. contactless

Memory vs. microprocessor

RFID

RFID - Radio Frequency Identification

Integrated circuit(s) with an antenna that can respond to an RF signal with identity information

No power supply necessary—IC uses the RF signal to power itself

Susceptible to replay attacks and theft

Examples:
Smart Tag, EZPass, Garage parking permits 13.56Mhz read/write support. May communicate with a variety of transponders (ISO15693, ISO14443 Type A & B, TagIt, Icode, etc.) Reader is controlled via PCMCIA interface using an ASCII protocol.

## III. AUTHENTICATION GLOSSARY

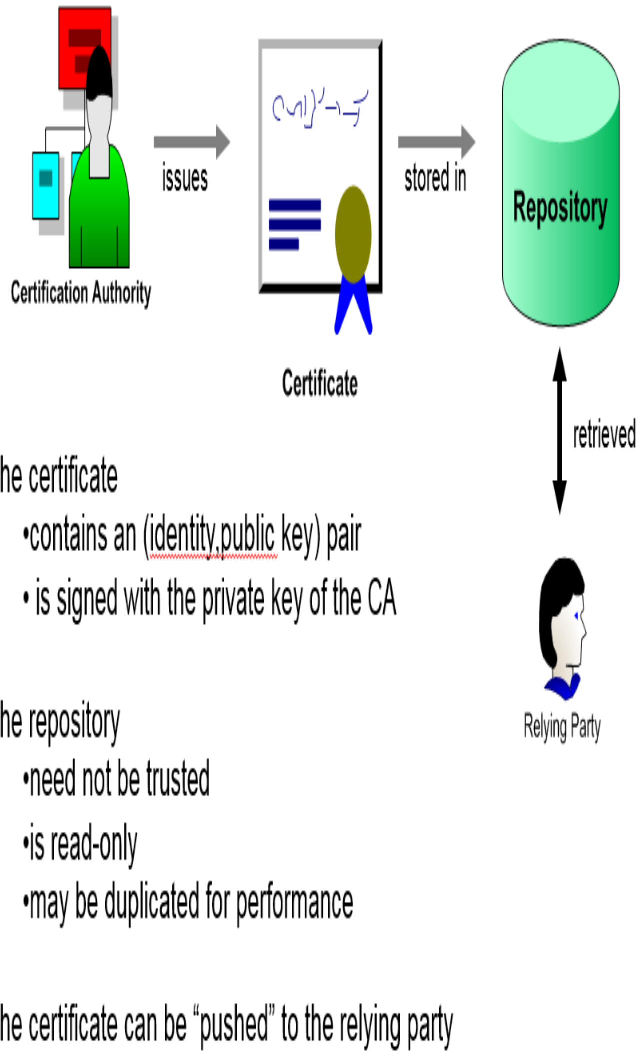
2-Way Authentication
•Authentication often needed in both directions
•Server trusting user is not only concern
–User must trust server
–Ex. User accessing online bank account
•Variety of "solutions" in user applications
Password-based Authentication
•Proof by sharing
•Doesn't prevent insider attacks (system admin)
•What is an appropriate password?
–length? snoopy, snoopy1, snoopy12
–reusable? snoppy1, snoopy2…. snoopy10, snoopy1
–timeframe?
•How to do initial password distribution? lastname123, employee#
•Simple approach, works with humans
… until user has too many to remember
–reuse across systems
–Variations of something common: dog's name
–post-it on monitor
–inconvenient to update, varying rules on what is appropriately complex, how often to change
snoopy1, Snoopy1, snoopy-1

## IV. AUTHENTICATION TOKEN FORMATS

X.509 Certificates
•Use of digital certificates issued by a trusted Certificate Authority (e.g. VeriSign)
•A Digital Certificate contains information to assert an identity claim
–Name
-Serial number
–Expiration dates
–Certificate holder's public key (used for encrypting/decrypting messages and digital signatures)
–Digital signature of Certificate Authority (so recipient knows that the certificate is valid)
•The recipient may confirm the identity of the sender with the Certificate Authority

A CA consists of a set of tools for generating and managing certificates and a database that contains all of the generated certificates.
When setting up a system, it is important to choose a suitable CA that is sufficiently secure for your requirements.
There are two types of CA you can use:
Commercial CAs are companies that sign certificates for many systems. Private CAs are trusted nodes that you set up and use to sign certificates for your system only.
Distributed set of servers that maintains a database about users. Each certificate contains the public key of a user and is signed with the private key of a CA. Is used in S/MIME, IP Security, SSL/TLS and SET.
RSA is recommended to use.

Certificates:
Carries the identity of an entity.

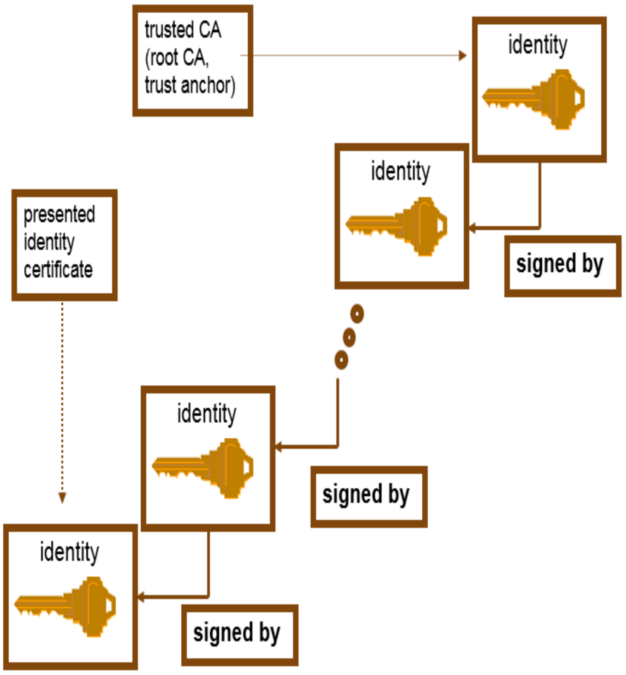


• the certificate
•contains an (identity,public key) pair
• is signed with the private key of the CA

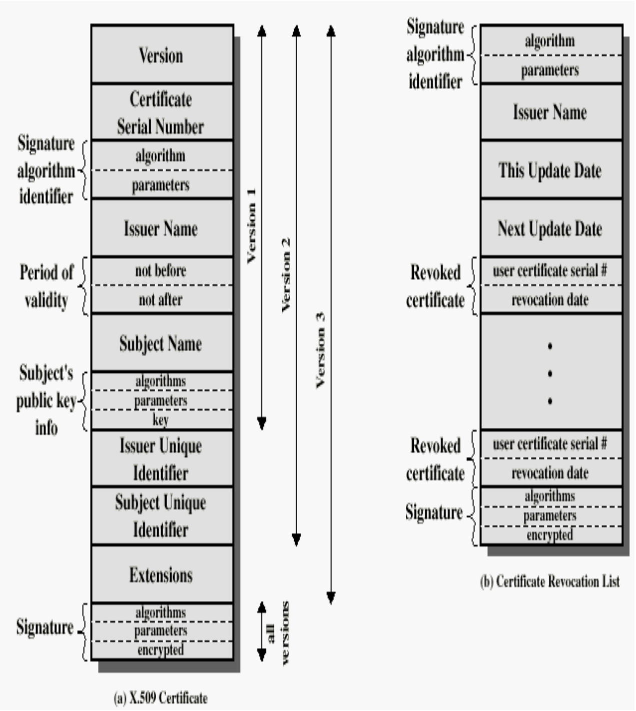
• the repository
•need not be trusted
•is read-only
•may be duplicated for performance

• the certificate can be "pushed" to the relying party

Chain of Trust :


trusted CA (root CA, trust anchor)
identity
identity
signed by
presented identity certificate
identity
signed by
identity
signed by

X.509 Certificate Format:



(a) X.509 Certificate

(b) Certificate Revocation List

## V. OBTAINING A USER'S CERTIFICATE

Characteristics of certificates generated by CA:

- Any user with access to the public key of the CA can recover the user public key that was certified.

No part other than the CA can modify the certificate without this being detected.

- Integrity of the public key

Authentication of a secure application depends on the integrity of the public key value in the application's certificate. If an impostor replaces the public key with its own public key, it can impersonate the true application and gain access to secure data.

To prevent this type of attack, all certificates must be signed by a *certification authority* (CA). A CA is a trusted node that confirms the integrity of the public key value in a certificate.

- Digital Signature:

A CA signs a certificate by adding its digital signature to the certificate. A digital signature is a message encoded with the CA's private key. The CA's public key is made available to applications by distributing a certificate for the CA. Applications verify that certificates are validly signed by decoding the CA's digital signature with the CA's public key.

- Content of an X.509 certificate:

An X.509 certificate contains information about the certificate subject and the certificate issuer (the CA that issued the certificate). A certificate is encoded in Abstract Syntax Notation One (ASN.1), a standard syntax for describing messages that can be sent or received on a network.

The role of a certificate is to associate an identity with a public key value. In more detail, a certificate includes:

- A subject distinguished name (DN) that identifies the certificate owner.
- The public key associated with the subject.
- X.509 version information.
- A serial number that uniquely identifies the certificate.
- An issuer DN that identifies the CA that issued the certificate.
- The digital signature of the issuer.
- Information about the algorithm used to sign the certificate.
- Some optional X.509 v.3 extensions; for example, an extension exists that distinguishes between CA certificates and end-entity certificates.

## VI. KERBEROS

Kerberos is a protocol for authenticating service requests between trusted hosts across an untrusted network, such as the internet. Kerberos is built in to all major operating systems, including Microsoft Windows, Apple OS X, FreeBSD and Linux. Since Windows 2000, Microsoft has incorporated the Kerberos protocol as the default authentication method in Windows, and it is an integral component of the Windows Active Directory service. Broadband service providers also use Kerberos to authenticate DOCSIS cable modems and boxes accessing their networks. Kerberos was originally developed for Project Athena at the Massachusetts Institute of Technology (MIT). The name Kerberos was taken from Greek mythology; Kerberos (Cerberus) was a three-headed dog who guarded the gates of Hades. The three heads of the Kerberos protocol represent a client, a server and a Key Distribution Centre (KDC), which acts as Kerberos' trusted third-party authentication service.

Users, machines and services using Kerberos need only trust the KDC, which runs as a single process and provides two services: an authentication service and a ticket granting service. KDC "tickets" provide mutual authentication, allowing nodes to prove their identity to one another in a secure manner.

Kerberos authentication uses conventional shared secret cryptography to prevent packets traveling across the network from being read or changed and to protect messages from eavesdropping and replay attacks.

*KERBEROS PROTOCOL OVERVIEW*

A simplified description of how Kerberos works follows; the actual process is more complicated and may vary from one implementation to another. For the purposes of this discussion, the initiating client in the scenario below is a corporate laptop running Windows, and an end user is trying to log into the corporate network.

To start the Kerberos authentication process, the initiating client sends a request to an authentication server for access to a service. The initial request is sent as plaintext because no sensitive information is included in the request. The authentication server retrieves the initiating client's private key, assuming the initiating client's username is in the KDC database. If the initiating client's username cannot be found in the KDC database, the client cannot be authenticated and the authentication process stops. If the client's username can be found in the KDC database, the authentication server generates a session key and a ticket granting ticket. The ticket granting ticket is timestamped and encrypted by the authentication server with the initiating client's password.

The initiating client is then prompted for a password; if what is entered matches the password in the KDC database, the encrypted ticket granting ticket sent from the authentication server is decrypted and used to request a credential from the ticket granting server for the desired service.

The client sends the ticket granting ticket to the ticket granting server, which may be physically running on the same hardware as the authentication server, but performing a different role.

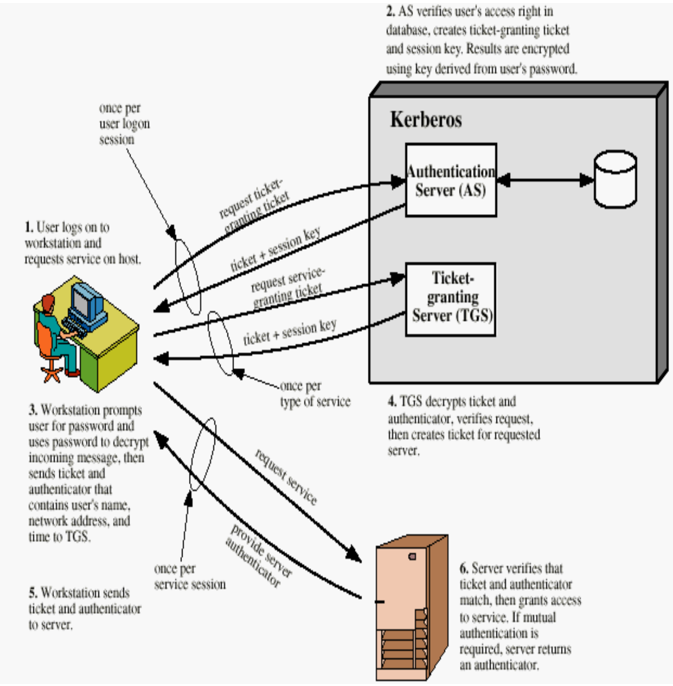
The ticket granting service carries out an authentication check similar to that performed by the authentication server, but this time sends credentials and a ticket to access the requested service. This transmission is encrypted with a session key specific to the user and service being accessed. This proof of identity can be used to access the requested "kerberized" service, which, once having validated the original request, will confirm its identity to the requesting system.

The time stamped ticket sent by the ticket granting service allows the requesting system to access the service using a single ticket for a specific time period without having to be re-authenticated.

Making the ticket valid for a limited time period makes it less likely that someone else will be able to use it later; it is also possible to set the maximum lifetime to 0, in which case service tickets will not expire.

Microsoft recommends a maximum lifetime of 600 minutes for service tickets; this is the default value in Windows Server implementations of Kerberos.

The MIT Kerberos Consortium was founded in September 2007 to further the development of Kerberos. In 2013, the consortium was expanded and renamed the MIT Kerberos and Internet Trust Consortium.


2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.
once per user logon session
Kerberos
Authentication Server (AS)
request ticket-granting ticket
ticket + session key
1. User logs on to workstation and requests service on host.
request service-granting ticket
Ticket-granting Server (TGS)
ticket + session key
once per type of service
4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.
3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.
request service
provide server authenticator
once per service session
5. Workstation sends ticket and authenticator to server.
6. Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

## VII. AUTHORIZATION

Authorization is a process by which a server determines if the client has permission to use a resource or access a file.

Authorization is usually coupled with authentication so that the server has some concept of who the client is that is requesting access.

The type of authentication required for authorization may vary; passwords may be required in some cases but not in others.

In some cases, there is no authorization; any user may be use a resource or access a file simply by asking for it. Most of the web pages on the Internet require no authentication or authorization.

## VIII. ENCRYPTION

Encryption involves the process of transforming data so that it is unreadable by anyone who does not have a decryption key.

The Secure Shell (SSH) and Socket Layer (SSL) protocols are usually used in encryption processes. The SSL drives the secure part of “https://” sites used in e-commerce sites (like E-Bay and Amazon.com.)

All data in SSL transactions is encrypted between the client (browser) and the server (web server) before the data is transferred between the two.

All data in SSH sessions is encrypted between the client and the server when communicating at the shell.

By encrypting the data exchanged between the client and server information like social security numbers, credit card numbers, and home addresses can be sent over the Internet with less risk of being intercepted during transit.

## IX. USING AUTHENTICATION, AUTHORIZATION, AND ENCRYPTION

Authentication, authorization, and encryption are used in everyday life. One example in which authorization, authentication, and encryption are all used is booking and taking an airplane flight.

Encryption is used when a person buys their ticket online at one of the many sites that advertises cheap ticket. Upon finding the perfect flight at an ideal price, a person goes to buy the ticket. Encryption is used to protect a person’s credit card and personal information when it is sent over the Internet to the airline. The company encrypts the customer’s data so that it will be safer from interception in transit.

Authentication is used when a traveller shows his or her ticket and driver’s license at the airport so he or she can check his or her bags and receive a boarding pass. Airports need to authenticate that the person is who he or she says she is and has purchased a ticket, before giving him or her a boarding pass.

Authorization is used when a person shows his or her boarding pass to the flight attendant so he or she can board the specific plane he or she is supposed to be flying on. A flight attendant must authorize a person so that person can then see the inside of the plane and use the resources the plane has to fly from one place to the next.

Here are a few examples of where encryption, authentication, and authorization are used by computers:

Encryption should be used whenever people are giving out personal information to register for something or buy a product. Doing so ensures the person’s privacy during the communication. Encryption is also often used when the data returned by the server to the client should be protected, such as a financial statement or test results.

Authentication should be used whenever you want to know exactly who is using or viewing your site. Web login is Boston University’s primary method of authentication. Other commercial websites such as Amazon.com require people to login before buying products so they know exactly who their purchasers are.

Authorization should be used whenever you want to control viewer access of certain pages. For example, Boston University students are not authorized to view certain web pages dedicated to professors and administration. The authorization requirements for a site are typically defined in a website’s “.htaccess” file.

Authentication and Authorization are often used together. For example, students at Boston University are required to authenticate before accessing the Student Link. The authentication they provide determines what data they are authorized to see. The authorization step prevents students from seeing data of other students.

## REFERENCES

[1] L. O’Gorman, “Comparing Passwords, Tokens, and Biometrics for User Authentication,” Proc. IEEE, Vol. 91, No. 12, Dec. 2003, pp. 2019-2040.

[2] http://www.computerworld.com/computerworld/records/images/pdf/kerberos_chart.pdf

[3] https://www.bu.edu/tech/about/security-resources/bestpractice/auth/

[4] http://www.computerworld.com/computerworld/records/images/pdf/kerberos_chart.pdf