WEB SECURITY AND PRIVACY

WEB SECURITY

EMAIL AND WEB ATTACKS

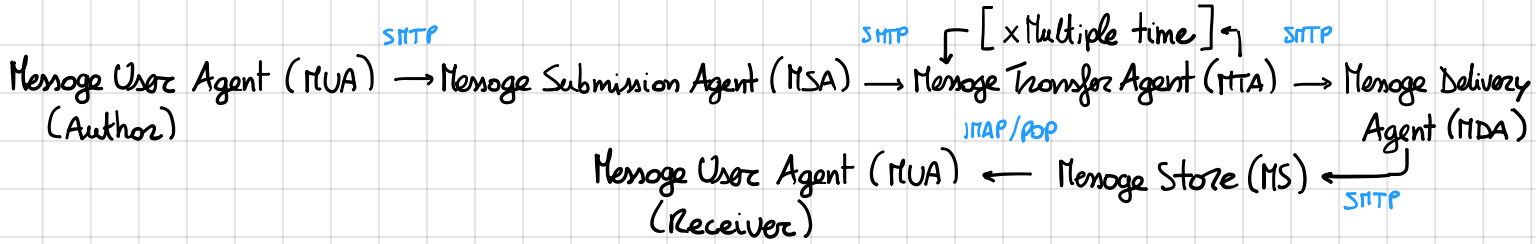
Most common attacks are made via email. No precise target, the target is the whole net.
RANSOMWARE: lock documents of victim and ask money to unlock them.
This a tipical example of attack with no precise victim → wait that someone get trapped (may be specific target sometimes)

Malwares can be sent via email as attachment (change extension to fake the victim) or by convincing the victim to click on malicious link (Phishing)

SPAM is not necessarily dangerous (eg. advertisement)

SPEAR Phishing: phishing where the victim is a specific person/organization. Content of email in this case is highly targeted.

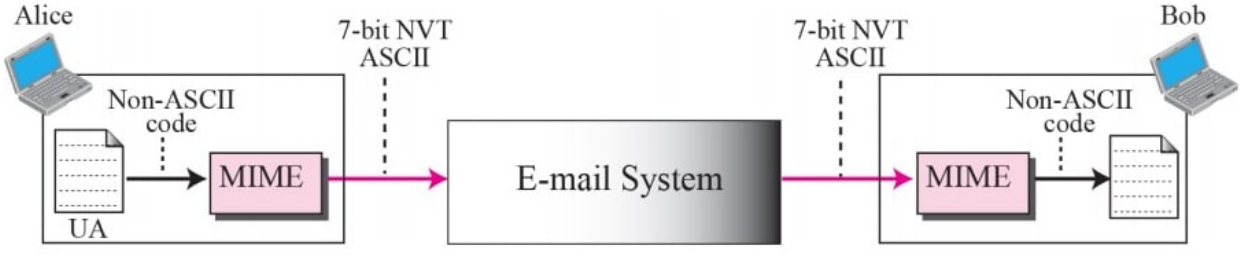
EMAIL



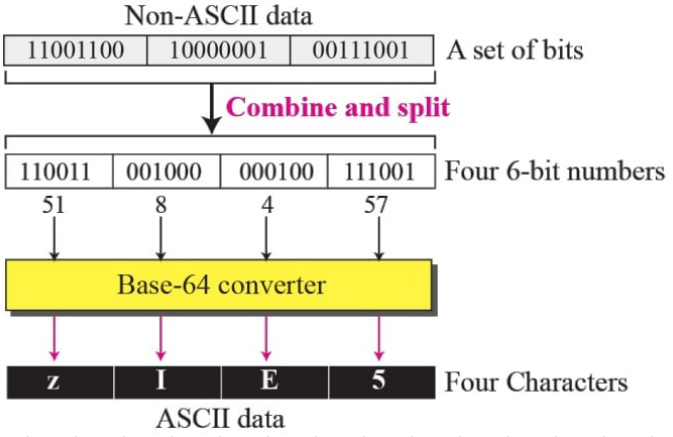
IMAP/POP act as a view on the MS sending command to it (messages are copies of the ones stored in MS).
Modern email systems use the store-and-forward model → Asynchronous model
EMAIL = message envelope = header + body
message content
Note: email messages are not secure if email encryption is not used
Header: structured in fields with <name: value> couples. "From", "Date", "Message ID", "In-reply-To" and so on.
One important field is "Content type" to define the language of the message (usually MIME).
Note: what is inside envelope may be incoerent with header e.g. From field of header ≠ from field of envelope (Forwarding on email)

SMTP defines the trace information of message and write it in the header (last to first)

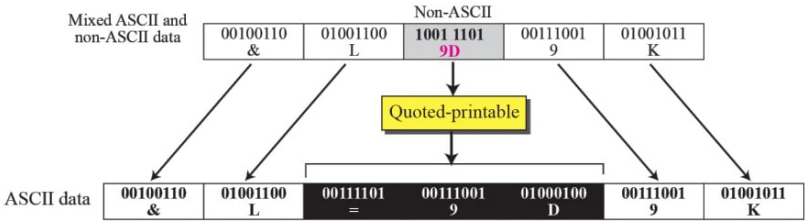
MIME Scheme: used to convert NON-ASCII code into 7-bit NVT ASCII to be send via email (SMTP)
NVT = network virtual terminal



MIME standard is defined in the header along with content-type field : type/subtype eg text/html
Content-Transfer-encoding : the way the message is encoded during transfer. An example of that is Base64 and Quoted-Printable



Value	Code	Value	Code	Value	Code	Value	Code	Value	Code	Value	Code
0	A	11	L	22	W	33	h	44	s	55	3
1	B	12	M	23	X	34	i	45	t	56	4
2	C	13	N	24	Y	35	j	46	u	57	5
3	D	14	O	25	Z	36	k	47	v	58	6
4	E	15	P	26	a	37	l	48	w	59	7
5	F	16	Q	27	b	38	m	49	x	60	8
6	G	17	R	28	c	39	n	50	y	61	9
7	H	18	S	29	d	40	o	51	z	62	+
8	I	19	T	30	e	41	p	52	0	63	/
9	J	20	U	31	f	42	q	53	1		
10	K	21	V	32	g	43	r	54	2		



- 8bit byte value may be encoded with 3 characters : "=" + two hexadecimal digits representing the byte's numeric value
- NON 8 bit bytes are ASCII chars from 33 to 126 (excluded 61 → "=")
- Special cases for SPACE and TAB

UNWANTED EMAIL

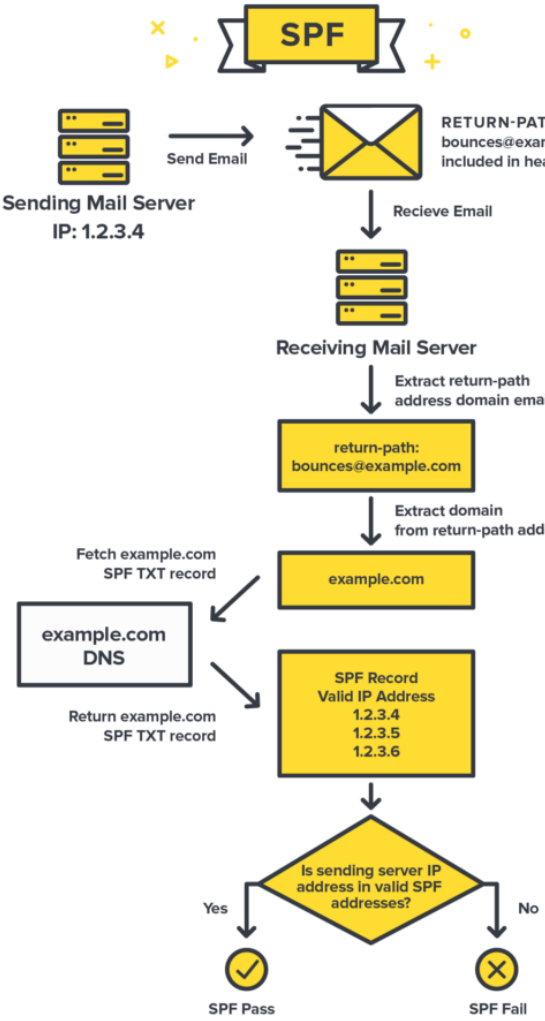
SPAM is just a little part of them. Frauds and Malware are very common too.
Main goal of spreading malware is to trap the most user possible.

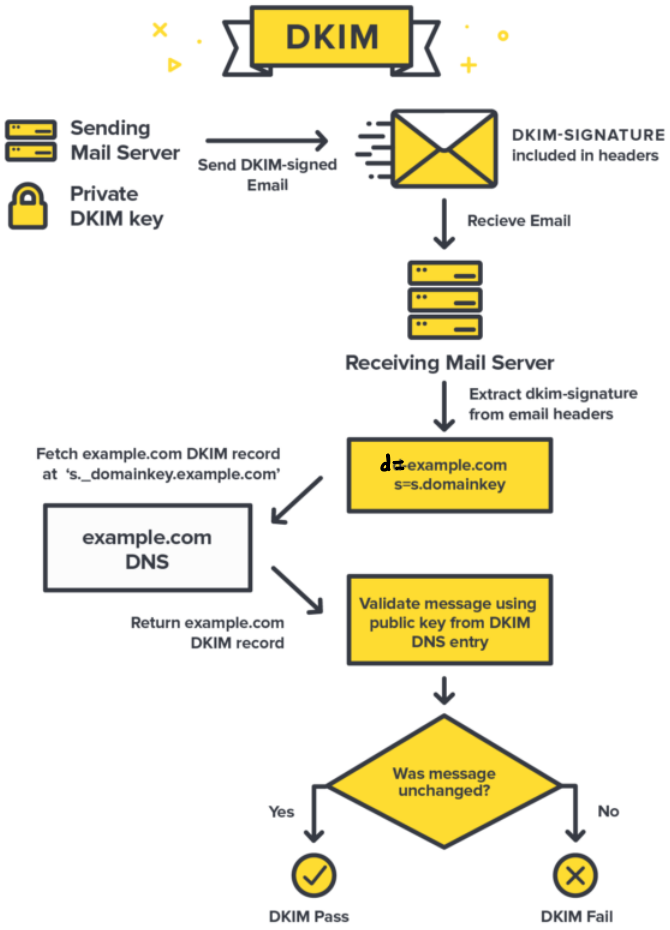
EMAIL VALIDATION SYSTEMS

① Sender Policy Framework
Prevents email spam by detecting email spoofing through verification of sender IP address
Sender address is protected in envelope by allowing the owner of the domain to specify which mail servers are authorized to send email from the domain using special DNS records.
The receiver first get only the header with return-path field to be checked using DNS record, if passes the verification, the rest of the message is downloaded.
CONS: difficult to update SPF records on every changes, SPF breaks when message is forwarded.

② DomainKey Identified Mail (DKIM)
Specify how the email messages can be cryptographically signed by the sender and checked on receive. CONS Message can be modified by mail server, invalidating the signature.

③ Domain-based Message Authentication, Reporting and Conformance (DMARC)
Technical standard that help protect email senders and recipients from spam, spoofing and phishing i.e allow an organization to publish a policy that defines its email authentication practices and instructions.
e.g DMARC policy, listed on DNS records, defines email authentication and how to handle violations
On receipt of message, check :
- DKIM signature valid?
- SPF authorization?
- Header shows "domain alignment"? For SPF, message's From domain and its Return-Path domain must match, For DKIM, From domain and its DKIM "d=" domain must match (the signing domain)





CHECK OUTCOMES

When receiving an email, we can check in the header the results of previous controls (SPF and DKIM)

④ Vouch by Reference (VBR)

Sender certification by third-party entities. A user of VBR email signs email with DKIM and includes in header a VBR-Info field in header containing the certified domain and one or more vouching services, domain names of services that vouch for the sender for a specific content type

⑤ Authenticated Received Chain Overview

Domains with strict DMARC policy may reject even good messages (eg. SPF fails just forwarding a message)
ARC instead allows multiple intermediaries and ARC header can be verified at each hop.
Authentication-Results = content intact from first intermediary forward.
At reception, the receiver check DMARC as always and if fails it can decide (according to intermediary's reputation) if override and check ARC.
ARC chains gives us DKIM, SPF, DMARC results as seen by first hop and allow intermediaries to alter message with attribution.

BASIC EMAIL ANALYSIS

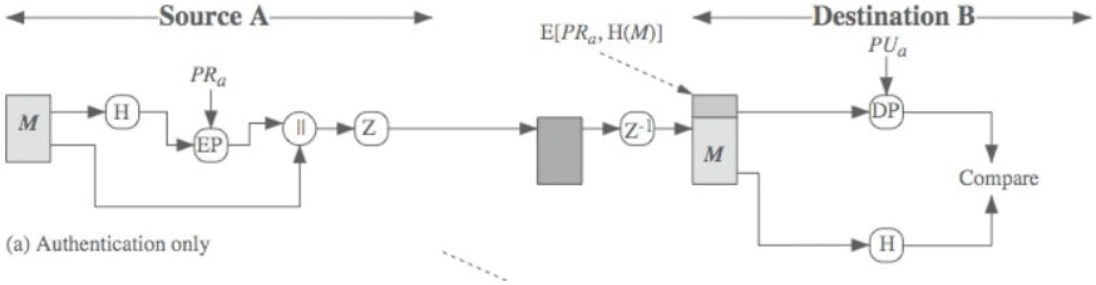
EMAIL SPOOFING : altering multiple fields of the header (usually the sender). Very easy without authentication.
To check for spoofing analyze full header + body and check for consistency (From, Return-Path, Reply-To, Received ...)

EMAIL NEEDS
- Confidentiality : protection from disclosure
- Authentication : of message's sender
- Message integrity : protection from modification
- Non-Repudiation : protection from denial by sender

⇒ Pretty Good Privacy (PGP) : standard for email privacy.

PGP Authentication

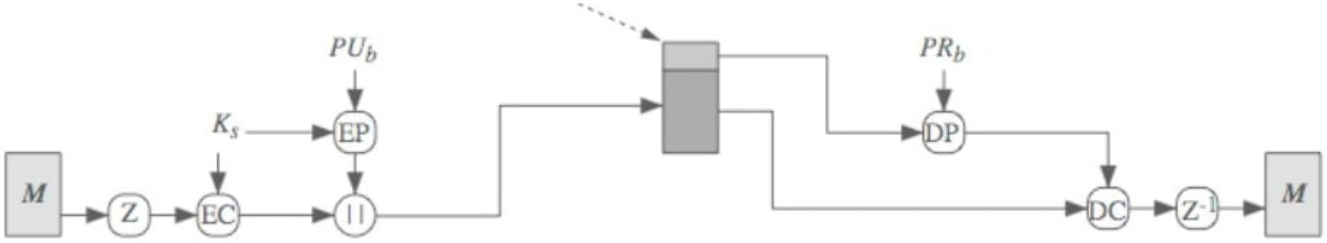
1. sender creates message
2. make SHA-1 160-bit hash of message
3. attached RSA signed hash to message
4. receiver decrypts & recovers hash code
5. receiver verifies received message hash



(a) Authentication only

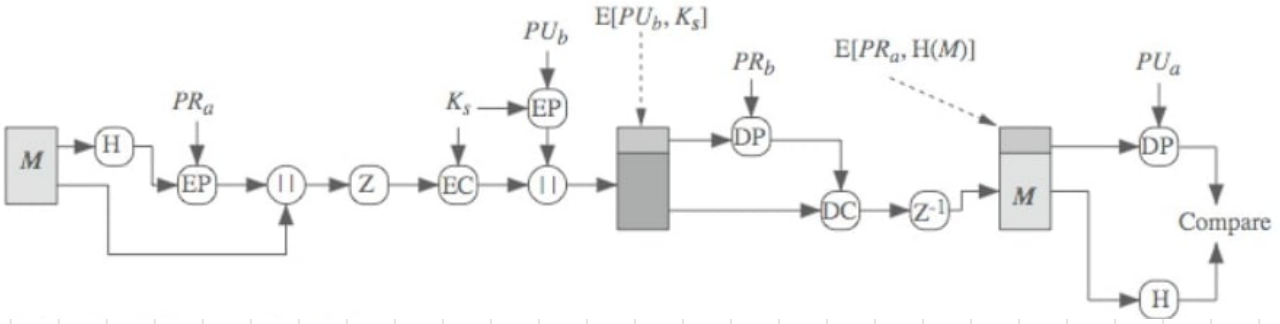
PGP Confidentiality

- sender forms 128-bit random session key
- encrypts message with session key
- attaches session key encrypted with RSA
- receiver decrypts & recovers session key
- session key is used to decrypt message



(b) Confidentiality only

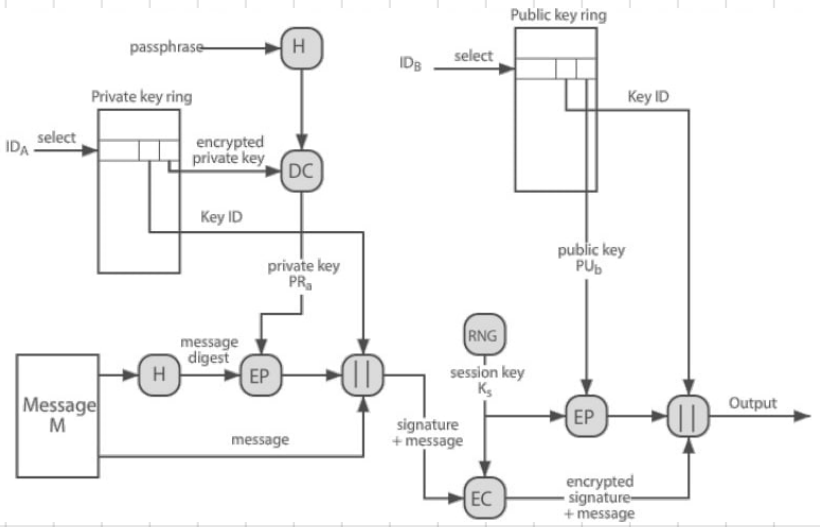
- can use both services on same message
  - create signature & attach to message
  - encrypt both message & signature
  - attach RSA/ElGamal encrypted session key



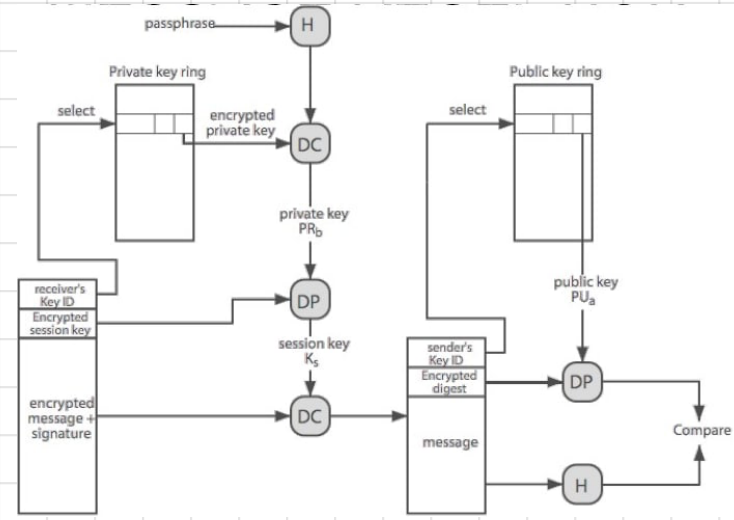
(Z) = ZIP = compression algorithm

PGP uses session keys for each message. Since there can be lot of pub/priv keys pairs (for each user) PGP adds key ID to target the correct pair.

Each user has a pair of keyrings:
- Public KeyRing contains all pubk. of users known by this user, indexed by ID
- Private KeyRing contains all pubk/priv keys pairs of this user, indexed by ID and encrypted using a passphrase.



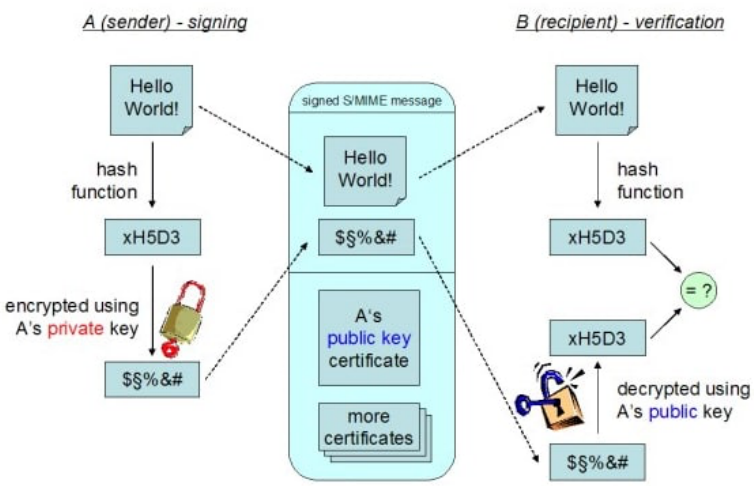
Generation

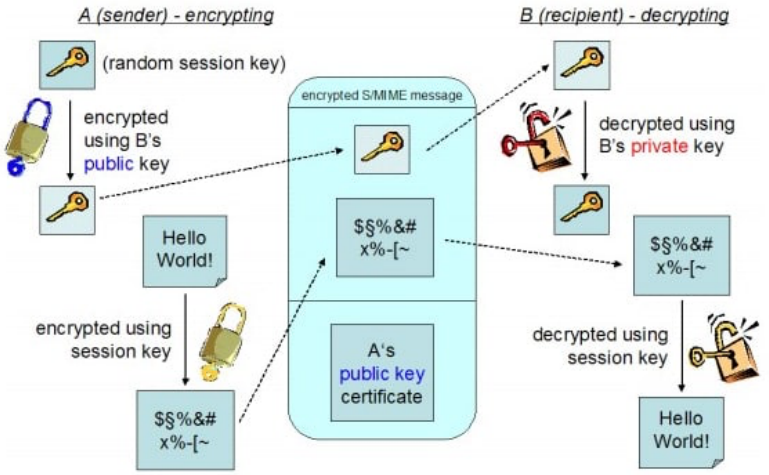


Reception

S/MIME

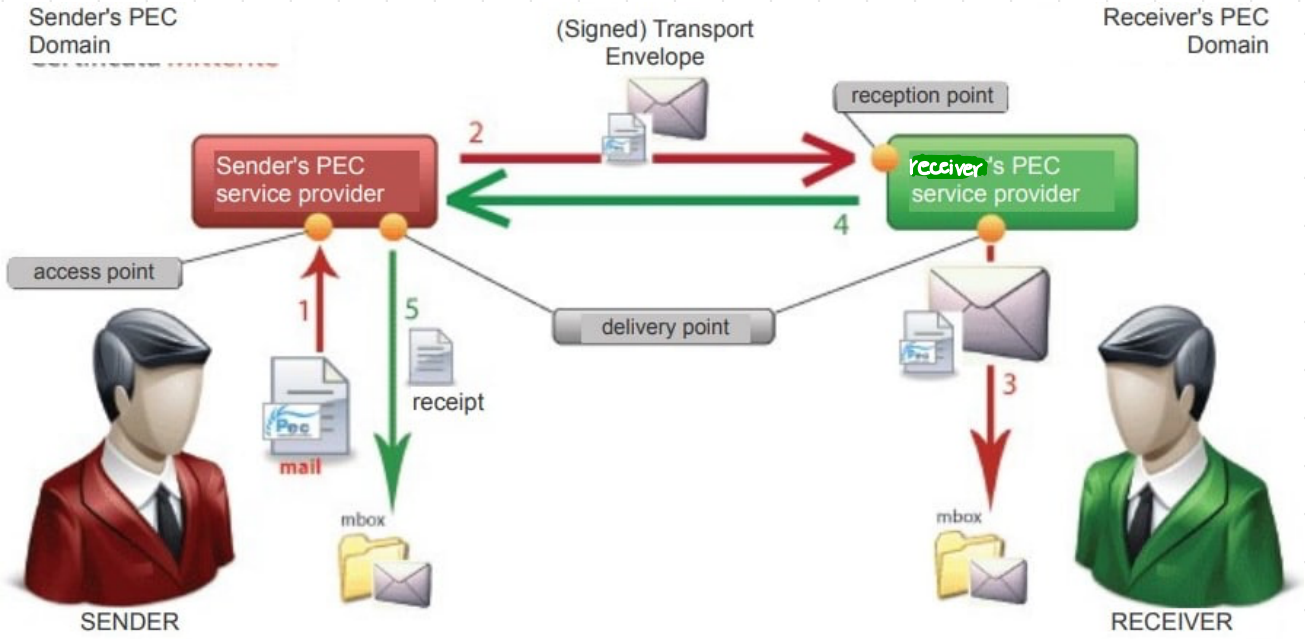
Email communication is insecure. Email can be read and modified as they are sent as clear-text. S/MIME is an attempt to standardize a protocol used to encrypt and digitally sign email correspondence. It works more or less like PGP: in addition the message contain various certificates (eg sender's public key certificate)





None of previous approaches give the sender any information about the message (is it arrived? receiver has read it?) → We need also tracing functionalities!

POSTA ELETTRONICA CERTIFICATA (PEC)



WEB SECURITY

Why? Because we are in war, kiddo! Attack increased il last 8 years. We must defend ourselves from communists!

→ Open Web Application Security Project (OWASP)

- Apply defence in depth: multiple levels (vertical and horizontal), not only border-based defences.
- Use positive security model: prevents by definition new types of attacks (whitelist)
- Fail securely: handle errors in security
- Run with least privilege: no more than needed (root not always needed)
- Avoid security by obscurity: secrecy is not always good
- Keep security simple
- Detect intrusions: log events, respond to intrusions properly
- Dont trust infrastructure: "trust nobody, not even yourself"
- Dont trust services: they could refer to external resources
- Establish secure defaults: default settings to improve security

RISKS

Most common are server side, usually with the goal of stealing data or remote command execution.
From client side they are mainly related to browser focusing on use and abuse of private information.
Last we have network side eg eavesdropping and man-in-the-middle.

About server side risks, web servers play a crucial role in web security. It is characterized by server root (logs, scripts, configurations) and document root (folder for web pages).
Server root may be used by attacker to run its own script and in the same way document root should be reachable by everyone but with huge limitations by the user (eg read only)

Server Root Permissions:
- rwx everywhere on "www" folder
- rx everywhere on "wwwgroup" but w in contents only
- rx on contents in "other"

Document Root Permission:
- o+rx to run webserver as nobody user
- ug+rwx to let authors to insert contents

Note that many web servers allow selective access to portions of document root based on IP/auth. but with o+rx permission, local user are allowed to access everything → server process runs with id ≠ nobody, with minimal privileges and belonging to wwwgroup → g+rx and o-rwx for "restricted" sections
If we make server running with other user id make sure that log directory is not writeable for it: attacker could exploit it.

Symbolic Links: extending the document tree with symlinks could be useful but dangerous since they can be exploited to reach not allowed folders.
Server Side Includes (SSI): possibility to dinamically include file inside an other served by the web server. Ofc. this can be used as exploit since the contained file can be executed
User-Maintained dirs: users have personal folder in root, what could go wrong?
Shared Areas WWW/FTP: good as long users cant upload files that can be executed by server daemon.

A common way to secure the intranet (LAN) is to put web server outside firewall and make it sacrificable installing "www" and firewall in different folders.

DUAL HOMED HOST: host has two network adapters dividing LAN and internet, check on packets before sending on LAN

SCREENED HOST (BASTION): a physical proxy. A screening router forwards all inbound traffic to bastion host and only it can send packets outside.

How to recognize a compromised server? Usually scanning logs but also them could be compromised. A better solution is to run a intrusion detection system that control and analyze the host 24/7

About client side: only way to install trojan or spyware.

"SAME ORIGIN" AND COOKIES

Compared URL	Outcome	Reason
http://www.example.com/dir/page.html	Success	Same protocol and host
http://www.example.com/dir2/other.html	Success	Same protocol and host
http://www.example.com:**81**/dir/other.html	Failure	Same protocol and host but different port
https://www.example.com/dir/other.html	Failure	Different protocol
http://**en.example.com**/dir/other.html	Failure	Different host
http://**example.com**/dir/other.html	Failure	Different host (exact match required)
http://**v2.www.example.com**/dir/other.html	Failure	Different host (exact match required)

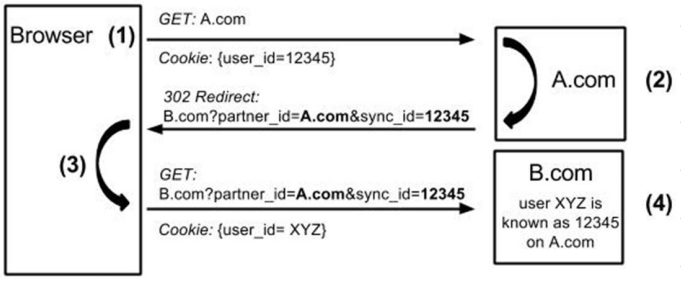
Two resources are considered from same origin if and only if domain name, application layer protocol and (in most browsers) port number are the same.

Cookies are small pieces of information stored on client side to allow HTTP to be "stateful". They are readable only to the website that has created them. They can be:
- session cookies: removed when browser quits
- persistent cookies: stay on client until defined expiration date.
- first party: received from a directly visited website
- third party: received from contents (coming from other website) hosted on visited webpage. eg advertisement.

About cookies, user profiling is considered as a potential threat to user privacy expecially from third party cookies. Law defines that cookies can be stored on user's pc only if:
- user has been informed ("accept cookies?")
- user can block the operation

Some cookies can be mandatory anyway.

Cookies syncing



THE ONION ROUTER (TOR)

As we said, all our actions online are tracked in some way, for example via cookies, for many reasons eg advertisement.
Internet is designed as public network and routing information is public but encryption does not hide identities (hides payload, not routing info)

ANONYMITY

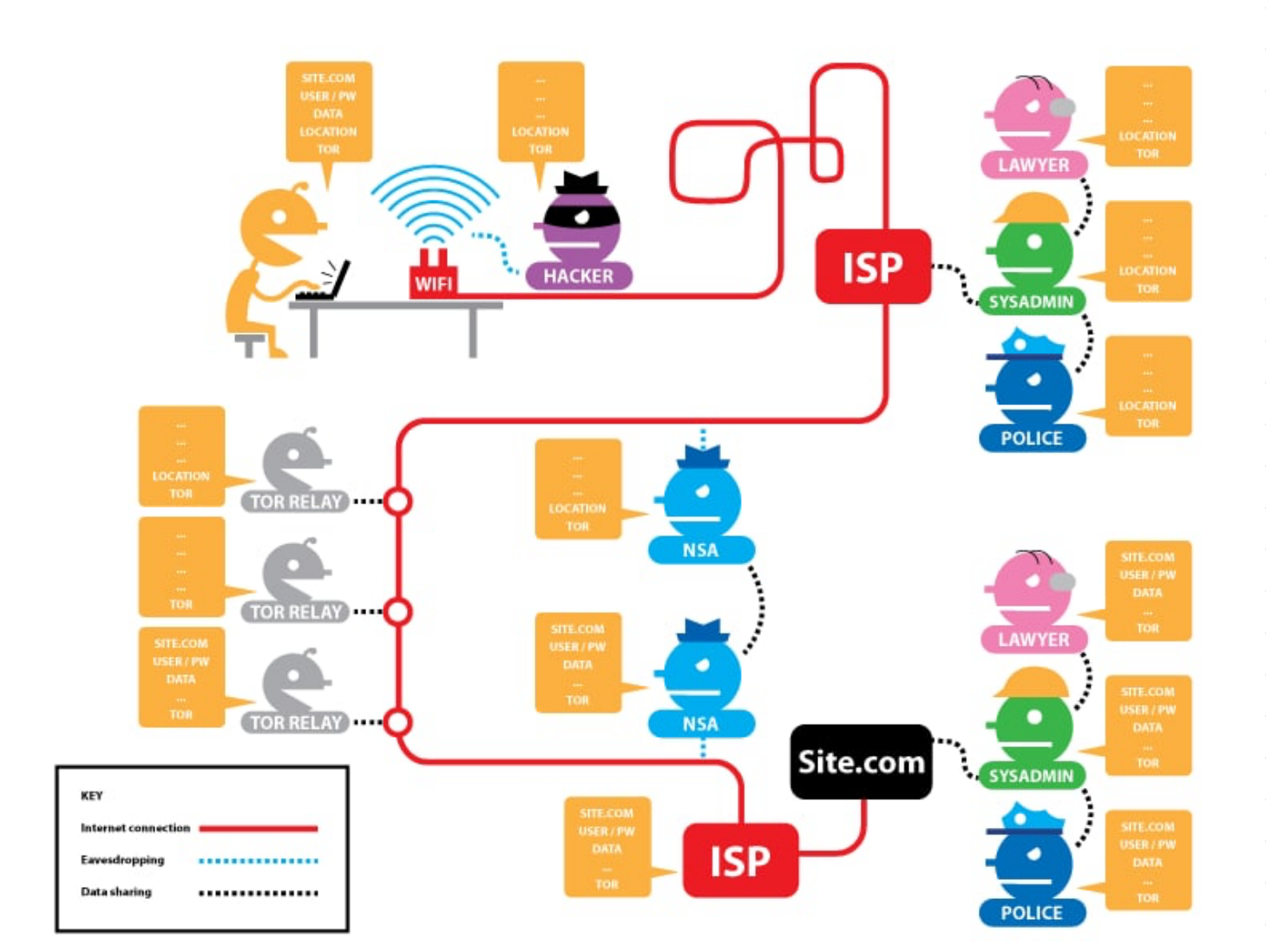
A person is not identifiable with a set of subjects. You cant be anonymous by yourself.

- Unlinkability of action and identity: eg. sender and his email no more correlated.
- Unobservability: hard to achieve, adversary can't even tell if we are using a system/protocol or not.

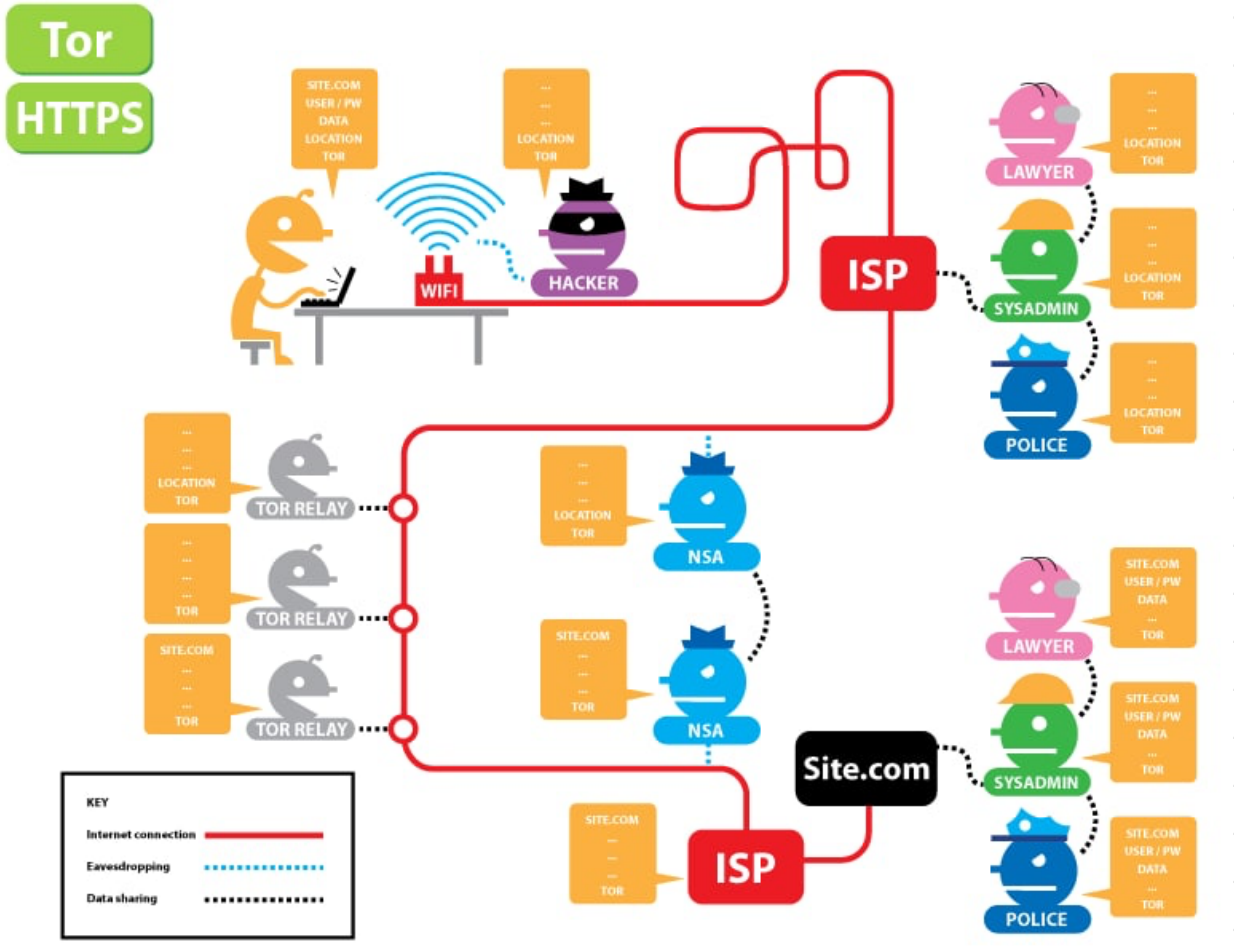
Possible attacks on anonymicity:

- Passive traffic analysis: scan pocket and find source and destination
- Active " : create a pocket signed and check where it goes
- Compromise nodes/router

TOR creates an overlay network that allow users to improve privacy and security. (+ many other features). Dont require ROOT privileges.

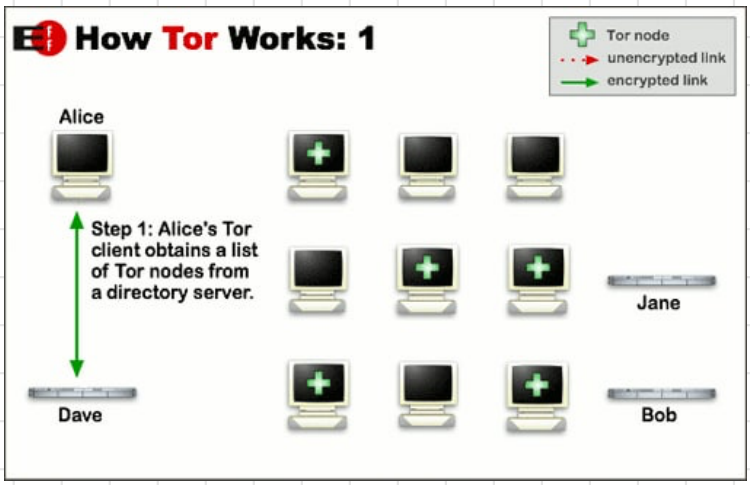


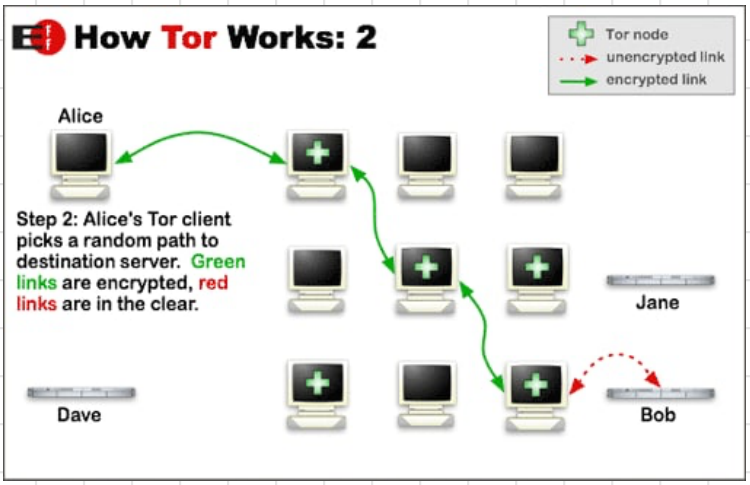
Even safier with HTTPS

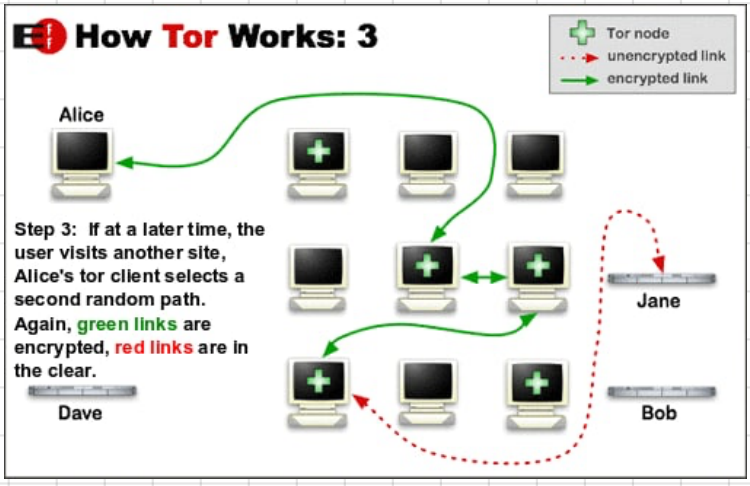


Tor network is made by onion router.

How it work:







Encryption: encrypt message as many times as the router the message will visite. Alice will have a shared key (AES) with each router. Hops number are configurable but fixed. Each router will remove a single encryption layer.

Every node knows only its predecessor and successor.

ONION ROUTER (OR): route traffic in the circuit

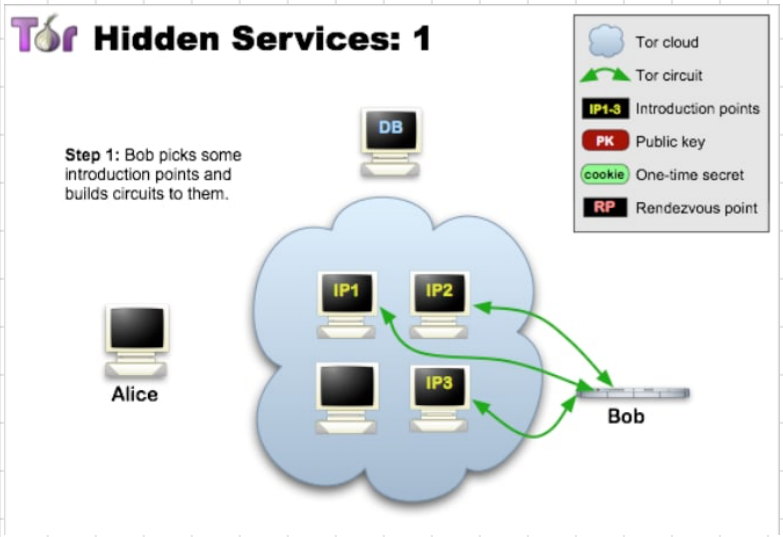
ONION PROXY (OP): is the tool that let the user interact with the virtual circuits on the network eg tor browser

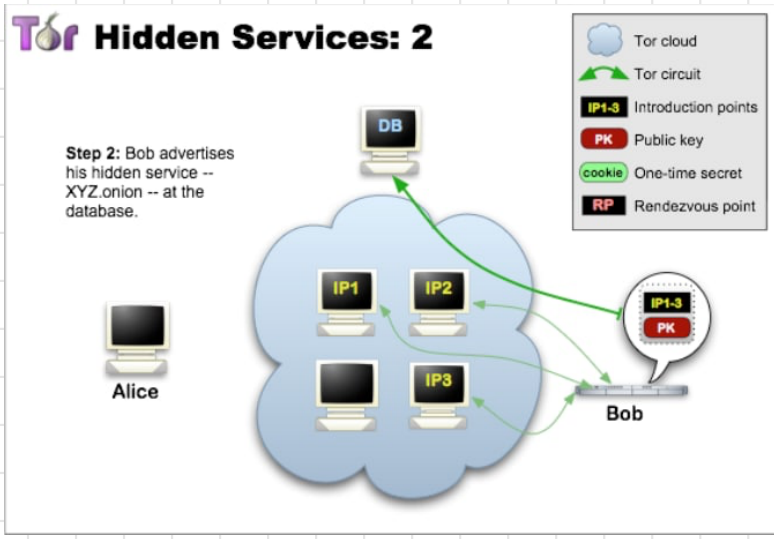
Both uses TCP with TLS and all data is sent in fixed size (bytes) cells.

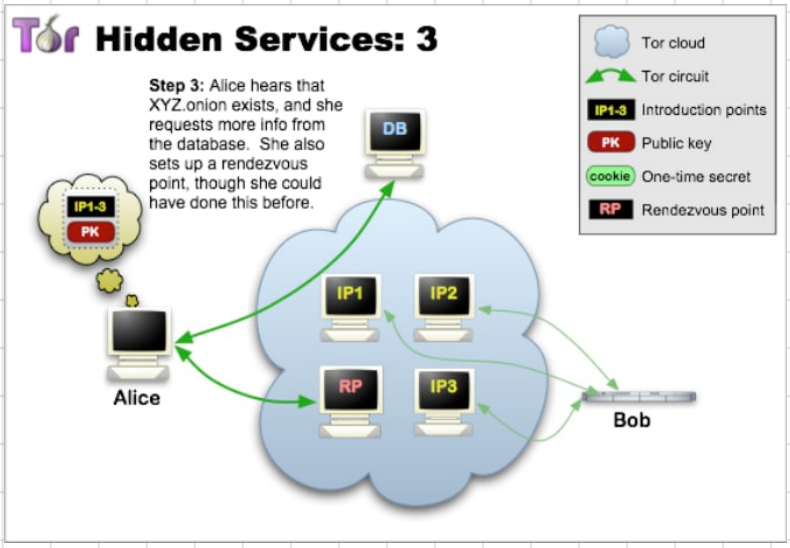
Directory Servers: they mantain lists of active relay nodes, their locations, current public keys etc. They are contacted during virtual circuit creation.

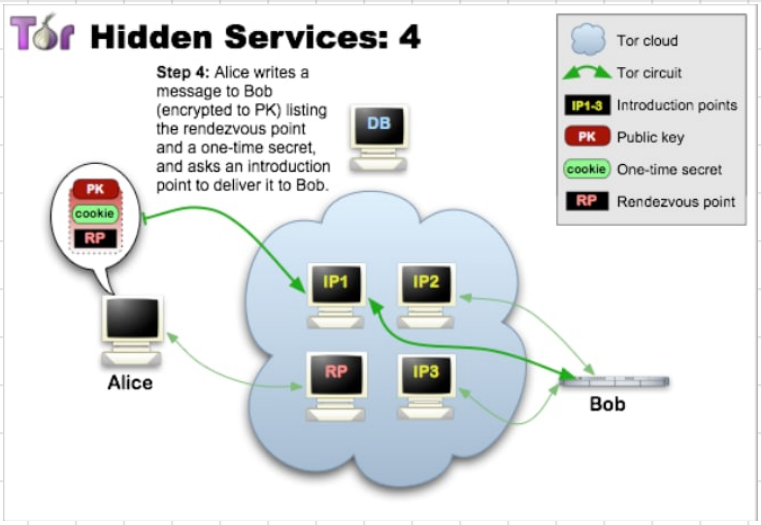
HIDDEN SERVICES

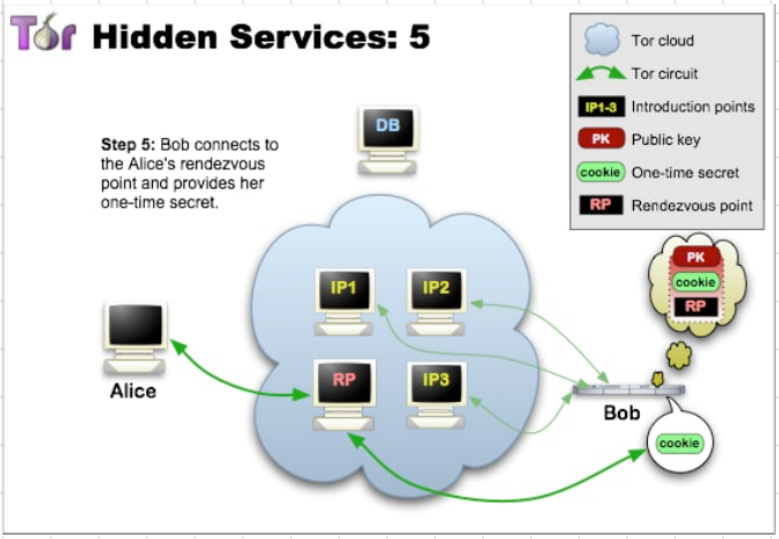
TOR makes possible to deploy a server on the internet that anyone can connect to without knowing where it is or who runs it. An example is SILKROAD website, only accessible via TOR

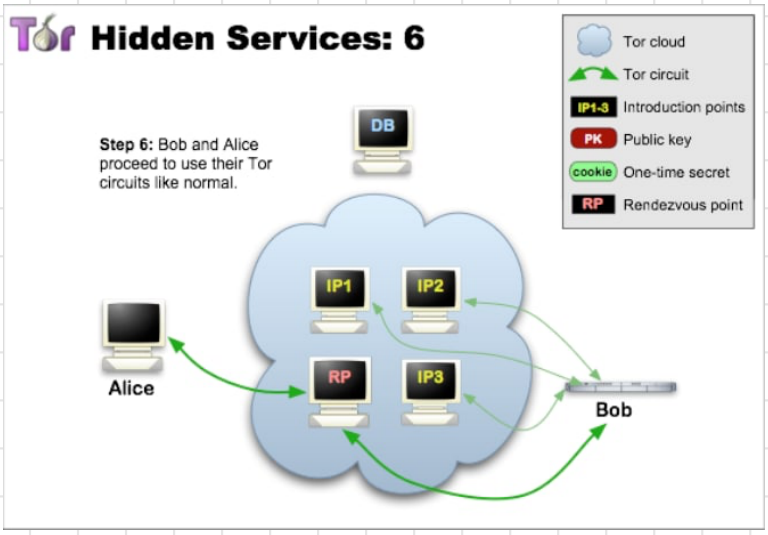












- Bob generates a long-term public key pair to identify his service
- Bob chooses some introduction points, and advertises them on the lookup service, also providing his public key
- Bob builds a circuit to each of his introduction points, and tells them to wait for requests.
- Alice learns about Bob's service out of band, retrieves the details of Bob's service from the lookup service
- Alice chooses an OR as the rendezvous point (RP) for her connection to Bob's service. She builds a circuit to the RP, and gives it a randomly chosen "rendezvous cookie" to recognize Bob.
- Alice opens an anonymous stream to one of Bob's introduction points, and gives it a message (encrypted with Bob's public key) telling it about herself, her RP and rendezvous cookie, and the start of a DH handshake. The introduction point sends the message to Bob.
- If Bob wants to talk to Alice, he builds a circuit to Alice's RP and sends the rendezvous cookie, the second half of the DH handshake, and a hash of the session key they now share.
- The RP connects the two circuits. Note that RP can't recognize Alice, Bob, or data they transmit.
- Alice sends a *relay begin* cell along the circuit. It arrives at Bob's OP, which connects to Bob's webserver.
- An anonymous stream has been established, and Alice and Bob communicate as normal.

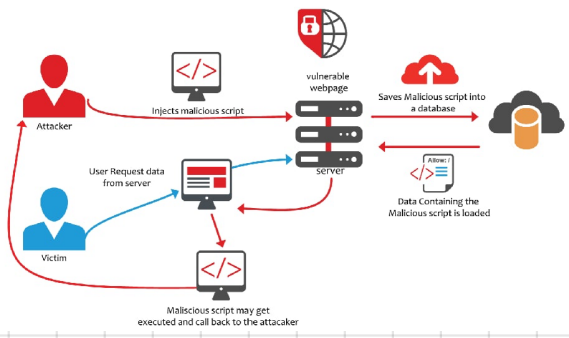
(HTTP) AUTHENTICATION

In few words: process of verification that an individual/entity/website is whom it claims to be. Usually makes use of username and password.
Old methods: send username/password without encryption or hash the password (digest)

FORM-BASED: browser requests protected data, server returns a page with an html form and performs verification and validation on data filled in by the user
Since http is stateless, a succesful authentication is immediately forgotten → solved using session cookie
Session cookie contains session id, is stored both server and client side and is used to bypass authentication.
Problem: session hijacking or cookie hijacking

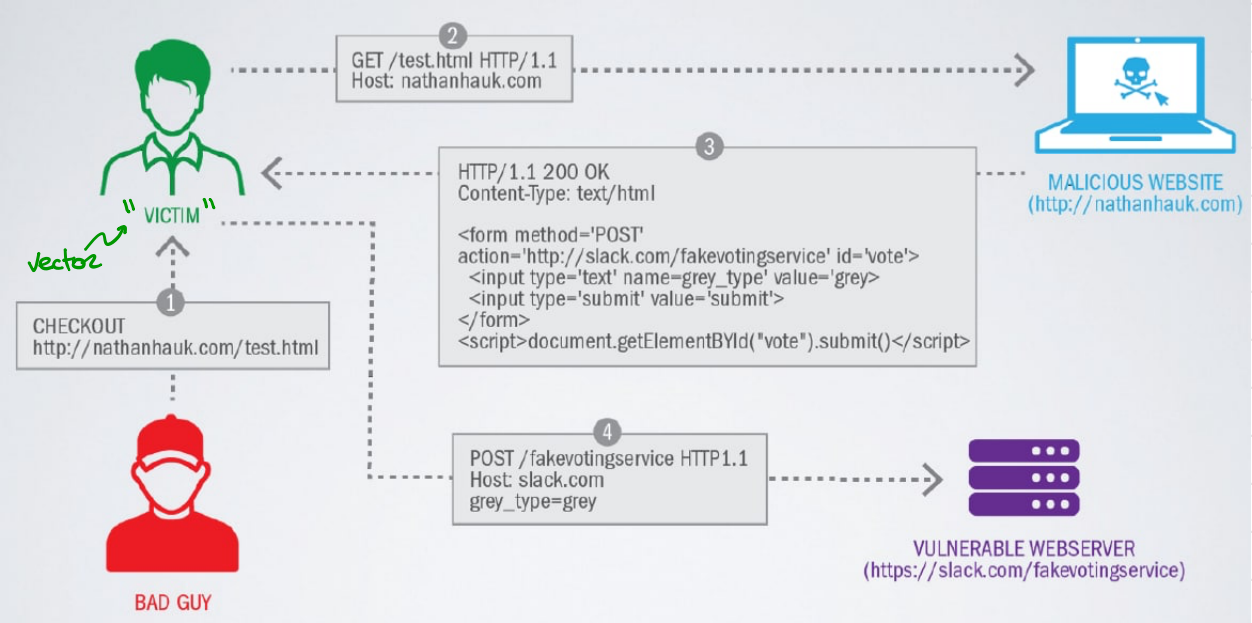
UNVALIDATED INPUT

General problem that possibly affect all software, including web applications. Attacker may tamper with any part of an http request. Usually caused by input validation only at client side
Input validation means to ensure that software receives only data it can use, rejecting non feasible ones.
How to implement it? → POSITIVE VALIDATION: explicit definition of type of data that can be send
eg non negative age, min/max length and so on.
Possible consequences are buffer overflow (heartbleed attack) or cross site scripting (XSS)
Typical XSS scenario: the "secure" webapplication used by the user acquires data from non-secure source and send them to the user → compromised data may be code that will be executed on user's machine
eg. Javascript or flash. During XSS attacks, personal information of the user can be eavesdropped.



In the case in picture the XSS attack is in the form of stored XSS: source code of scripting is stored on vulnerable server and sent to user when he requests data. Most dangerous.
Reflected XSS uses other ways such as links via emails or redirect of webpages to send source code of script.
DOM-based: alter DOM environment in victim's browser to let original script is executed in some unexpected way

An attack similar to XSS is CSRF, cross site request forgery (c-surf). In this case the victim is the web application and the user session is the attack vector. User authenticated session required.
Attacker builds a fake request that mimics a valid user request and lures an authenticated user to send the request to the application.



Solution: make sure forms require some sort of random token on every submission

INJECTION

An other type of attack is Injection Flaws. If the web application takes user input into a back-end database, shell command or operating system call an attacker may append additional "functionalities" allowing him to create, read, update or delete any data. An example is SQL-injection: in this case the attacker "uses" the user to add/delete unwanted data inside DB (for which the user is allowed).
Impacts can be very serious (data breaches, data corruption etc).
HOW IT WORK: client injects SQL code into input data of an application. Typical scenario: application dynamically creates SQL query using altered data without good validation.

Even if no output is provided to injector, generic error can be used to probe the DB for its content (ask DB true/false questions). This is called BLIND injection

Basic prevention system:
① input validation
② parameterized queries: based on predefined query strings whose structure is already defined
③ stored procedures: validation at server side (increasing its load).

#1 in OWASP top 10 critical risks.

BROKEN AUTHENTICATION

#2 in OWASP top 10 critical security risks, mostly due to password-based auth. or session token stealing.
Typical faults: poor password management, offline dictionary attack, predictable/weak passwords (or mix of all of them).
Password stealing countermeasure: 2-factor authentication.

What is a poor password management?
- password stored as plaintext
- storing of encrypted password
- storing of secrets deriving from passwords

SESSION HIJACKING

① By XSS: HTML allows user to post unfiltered data → attacker edit website to let user send him cookies
② SESSION FIXATION: website accepts ANY session identifier → attacker may trick the user to let him click a link to the unsafe website already with a (known to attacker) identifier. User will authenticate → attacker can now use the identifier to login.
Solution: session must be stored somewhere and should be created only by the application itself.
③ SESSION SIDEJACKING: some websites use TLS encryption only for authentication → session cookies can be sniffed
Solution: send cookies only on HTTPS.
④ based on MALWARE: may altere browser and its configuration or grab files/folders containing session ids.

CONTENT SECURITY POLICY (CSP)

An added layer of security that helps to detect and mitigate certain types of attack (XSS and injection). It is an HTTP header that provides tighter control over script loaded. CSP reduce XSS vectors by specifying the domain that that the browser should consider to be valid source. HTTP header response or meta field inside page.
A policy is set by specifying a set of rule: for each content-type, from where can we load it?

In addition:
1. a nonce can be generated and some value must be provided in the tag that loads the script. If doesnt mach → no execution
2. CSP can specify hash of the content of the script trusted. If the hash of the actual script it wont be executed.

ACCESS CONTROL ATTACKS

Access control is the critical mechanism within application because responsible for making the decision of whether it should permit a request or not. If defective, an attacker may take control of whole system.
VERTICAL ACCESS CONTROL: allow different types of users to access different parts of the application. Used to enforce business policies such as separation of duty and least privile
HORIZONTAL ACCESS CONTROL: allow users to access a certain subset of a wider range of resources of the same type eg. web store user can access only its own orders.

Possible attacks: Vertical / Horizontal privilege escalation
Weaknesses:
COMPLETELY UNPROTECTED FUNCTIONALITY
In many broken systems, sensitive functionality can be accessed by knowing the relevant URL. Link can be guessed or bruteforced, may appear in history / bookmarks etc.
IDENTIFIER-BASED FUNCTIONS
When a function is used to gain access to a specific resource its very common to see an identifier for the request → Attacker just need to know the name of application page and identifier of document he wishes to view.
MULTI STAGE FUNCTIONS
Some webpage use multistage functions eg step 1 - step 2 etc.
Problem is that on further stages, past one may be considered checked already
→ attacker may access step 2 avoiding step 1 and website will consider him authorized.
STATIC FILES
Some files are accessed just by their URL → just knowing URL you can access the resource.

How to implement a good access controls?
1) Explicitly evaluate and document the access control requirements for every unit of app. functionality
Plan before development to avoid vulnerabilities. Design for: Least Privilege and Separation of Duties

2) Drive all access control decisions from the user's session
3) Use central application component to check access controls.
4) Process every single client request via this component to validate that the user making the request is permitted to access the functionality and resource being requested
5) Use programmatic techniques to ensure that there are no exceptions to previous point.
6) Increase security checks for particularly sensitive functionality
7) Two method to protect static contents:
   - files can by accessed by name on a server-side page that implements relevant access control logic
   - HTTP authentication
8) Server should trust only identifier (data in general) created at server side
9) Per-transaction re-authentication and dual authorization for security-critical application functions
10) Log every event
11) Layered Access Control: each layer has its own access control

PROTECT DATA IN TRANSIT

When using ordinary HTTP connection, users are always in danger (most common risk is Man-in-the-Middle attack). Users has no control on internet environment security and should consider every channel as a potentially dangerous link. Or at least use TLS (HTTPS) channels. HTTPS guarantees basic security functions: confidentiality, data integrity and authentication (if needed). Generally TLS is used only on critical connections but this may leak data such as session id from different resources (eg. Https connection to website but HTTP download of content)
→ always use TLS (also for cookies).

Possible attack:
SSL STRIP ATTACK: downgrade HTTPS connection to HTTP with MitM attack. Attacker setup a fake router between victim and server and proxies all connections. Victims will connect to the fake router via HTTP while the attacker setup HTTPS connections with real server: protected contents from server are forwarded on the unprotected channel.
Solution: HTTP Strict Transport Security (HSTS), after an insecure request, server wont accept new ones from browser without HTTPS.

PRIVACY

PRIVACY

Defining privacy is difficult cause there is not a common definition (literally: deprived of something)
What is considered private differs among individuals / entities

NOTHING TO HIDE argument: if you have nothing to hide, then what do you have to fear?
→ This assumes that people want to hide only bad informations → Not always true!
Private informations, legally ok, may still be used against you.
"NTH" claims that limited surveillance of lawful activity will not limit individuals but:
① aggregating data: combining small informations innocuous alone may say many things
② exclusion: people dont know how their data are used and cant change wrong ones
③ secondary use: data collected for one purpose can be used for a second one

Privacy Policies come with some problem:
- hard to find
- hard to understand (legal language)
- expensive to read (if every user read GDPR, for every site, they would spend time that they would have spent on the website → companies lose money)

More: our privacy is in danger also just browsing the web because cookies we "create" can be sold by companies for target advertising. Which page we visited, how many time, when and so on.

BITCOIN

Most payments are expensive online (cause of fee) and not anonymous.
Online payments are made over SSL, simple but seller have info of clients.
Mastercard propose SET, a set of protocols and formats for anonymous transactions but too much complicated → never used.

Bitcoin has some properties of common money plus:
- Decentralized
- Immune to censorship / confiscation by authorities

It still has transaction fee (most of the times)
Bitcoin works over blockchain → google it if you dont know it!

Properties:
① Cryptographic hash function: to ensure collision resistance etc
② Digital Signatures
③ Public key

Note: "anonimicity" is ensured because to create a wallet you dont need any ID document!

Every computer connected to blockchain is called node.
FULL node store all blockchain copy, LIGHT node just mines new block and uses Simplified Payment Verification (SPV) to verify transactions only using block headers (hash)

DISTRIBUTED CONSENSUS:
① Form Block: collect transaction from the world and put them together to form a block
② Check Block: check if block is correct checking transactions backword (ledger)
③ Solve Math (Mining): to make "costly" and prevent fake transactions, only first one is rewarded.
④ Announce Block: send new block to the world
⑤ Users verify block correctness

Cold Storage: keep your crypto wallet stored offline instead of keeping it connected to internet (hot).

Back to anonymity: there are some way to overcome it. For example the first node who notify the others of a transaction is usually the one who generated it.
Anonymity comes with the dilemma that it can help also fraudolent systems (eg Silk Road).
When paying real goods with bitcoin (eg a car) you are asked to provide your ID and so it would be posible to link your ID to your BTC address ⇒ Mixing Services: instead of sending directly to seller, move BTC in other's addresses to make it hard to trace you.
Principles:
① Use a series of mixes. Make them easy with standard API
② Uniform transactions: all transactions should have same value
③ Client side automated.
This model was proposed as COINJOIN, a standard for decentralized mixing that should grant anonymity (FAILED)

Back to cryptocoins: two approaches, proof of work (PoW) and proof of stake (PoS)

PROOF OF WORK
To add a block in the blockchain, a node must first solve a cryptographic puzzle (to avoid double spending etc) and is rewarded with digital values. This reward is needed to keep up the work of nodes (mining). Nowadays this puzzle is very complicated due to the huge input of the algorithm.

PROOF OF STAKE
The "validator" is choosen according to its "stake" i.e how much (crypto) money it has. The more it has the higher will be the probability to be elected as validator. This is called forging.
Making proportional to user's shares the validation system aim to avoid fraudolent behaviours cause people with high amount are not interested in bad actions that could lower their money value.

ETHEREUM
Ethereum is a decentralized platform that runs "smart contracts": self-executing programs that run as programmed without any posibility of downtime (i.e blockchain always running), censorship or attack.
Is more complicated than BTC since smart contracts can do whatever action they are programmed to do → Money is just to incentivise the transaction validation or as a fee (fuel)

The blockchain challenges are:
- be tamper-proof → use one-way hash function
- distributed consensus → different on each blockchain.

} To solve them an agreement must be reached

Also, BLOCKCHAIN TRILEMMA: security, scalability, decentralization (eg. BTC suffer of scalability and permissioned blockchains are not decentralized)

⇒ ALGORAND

Key Ideas:

① Weighted users as PoS i.e. according to their money.
Consensus algorithm should be Byzantine Consensus Alg. (BA)

② Instead of having all nodes run BA, let a subset of them run BA*. This group represent all nodes, is called committe and rotates at each block.

As long as an attacker controls less than 1/3 of monetary value of the system he cannot amplify his power with pseudonyms. Probability for forks is negligible.

This is called PURE PoS.

BLOCK Creation:

- Phase 1: single token is randomly selected and its owner is the one who will propose the next block.
- Phase 2: X tokens are selected among all (committe) to approve phase 1 block

How to choose the committe members? Each node play a cryptographic lottery (everybody the same) that result in an evidence of win if succed. After the win, the nodes notify all other about their partecipation in the committe (verifiable) and their "opinion" on new block.

SCALABILITY: lottery is calculated locally and committe members are fixed → fixed messages
SECURITY: committe members are secret before their first message and after that they cant be corrupted since the message is already in the network

SPLIT COINS

HARD FORK: change in block validation rules such that all nodes using old software will see block validated from new nodes as invalid. If a group of node continue to use old software a permanent split may occur
SOFT FORK: changes in blockchain backward compatible that let old nodes validate new transaction if they follow new rules.

Why we talk about this? Because, more or less, bitcoin blockchain verify ~7 transactions per second and if we have lot of transactions in queue why a miner should pick ours? → transaction fees! User can add a "reward" for the miner who validate its transaction: higher the fee, higher the probability to be picked
⇒ Problem: fees are too high now!
→ Solution 1: rise block size from 1MB (hard)
⇒ Solution 2: block of any size (harder)

Applying one of these solution a permanent split may occur if no consensus.

K-ANONYMITY AND OTHER CLUSTER-BASED METHODS

Large amount of person-specific data has been collected in recent years: health-care datasets, genetic datasets etc.
This bring a problem: Inference control, protect this kind of data (private) while publishing useful information (eg. statistics)
Access and disclosure control must be granted.

PRIVACY

How? Remove "personally identifying information" (PII) such as name, phone number etc.
Not enough: attacker could infer the identity combining multiple data set.
Microdata: series of record. Each record contains info about a person / institution etc.
Macrodata: computed data eg statistics
Masked Microdata: microdata without PII
External Information: any known info by attacker related to some individual from initial microdata.

Protecting microdata means avoid the possibility to infer PII.
Macrodata "cells" are sensitive when the value associated to them is below a certain threshold eg a single person with HIV (threshold rule)

Note: [zipcode, birthday, gender] (quasi-identifier) uniquely identify 87% of population in USA.
→ K-ANONYMITY: info of each person contained in released table cant be distinguished from at least k-1 individuals, any quasi-identifier must appear in at least k records
Generalization: replace q.i. with semantically consistent value eg age=29 → age < 40
Suppression: remove completely the q.i.

HIPAA Privacy Rule: "under the safe harbor method, covered entities must remove all of a list of 18 enumerated identifiers and have no actual knowledge that the information remaining could be used, alone or in combination, to identify a subject of the information."

K-anonymity does not provide privacy if is still possible to "combine" multiple k-anonymous table to infer informations. Also K-anonymity is not sufficient due to its pure syntactical composition.
HIPAA only works if there is no knowledge remaining that can be used.
Both use PII → PII has no guarantee and has no precise definition.

ALTERNATIVES

- l-diversity: each Q.I. class (set of record with same Q.I.) must show at least l distinct values of the sensitive attributes.
- t-closeness: distribution of sensitive attributes within each Q.I group should be "t close" to their distribution in the entire original database.

PRIVACY-PRESERVING DATA MINING

Allow user to query a database without let him infer information.

DATA SANITIZATION

The process of disguising sensitive information in database by overwriting it with realistic looking but false data of similar type.

Many way to achieve this:

- input perturbation: add random noise to DB and release it
- reveal only summary stats. (not data)
- output perturbation: summary stats. with noise

Must find a trade off between privacy and the release of (almost) true informations.

There are 3 concepts of privacy:

① Weak: no single DB entry has been revealed

② Stronger: no single PIECE of information is revealed

③ Strongest: the adversary's belief about the data have not changed

Note: sometime "randomization" of data may still let adversary to infer original data if done not in a good way.

→ New definition of privacy: after each query, adversary's gain in knowledge about any individual database entry should be small

DIFFERENTIAL PRIVACY

Whatever is learned about one respondent A would be learned regardless of whether or not A participates to the database.

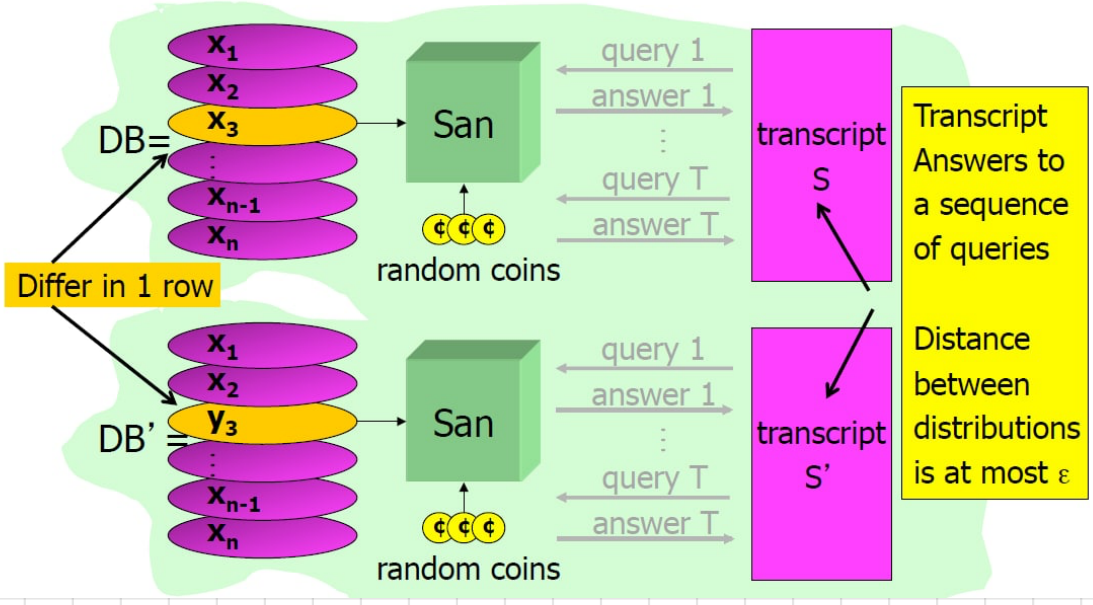
Promise: no one will be affected by allowing to use his data in any study/analysis no matter what other studies/analysis are already available.

→ statistical outcome is not dependent from a single user (record)

For every pair of inputs D_1 and D_2 that differ in one row, for every output O:

$\frac{Pr[A[D_1]=O]}{Pr[A[D_2]=O]} < \exp(\varepsilon)$ with $\varepsilon > 0$ if algorithm A satisfies differential privacy.

⇒ small difference between output



Now the question is: how much "noise" to add?
It depends on sensitivity of query and ε value. More noise more privacy but less valuable data (and viceversa)

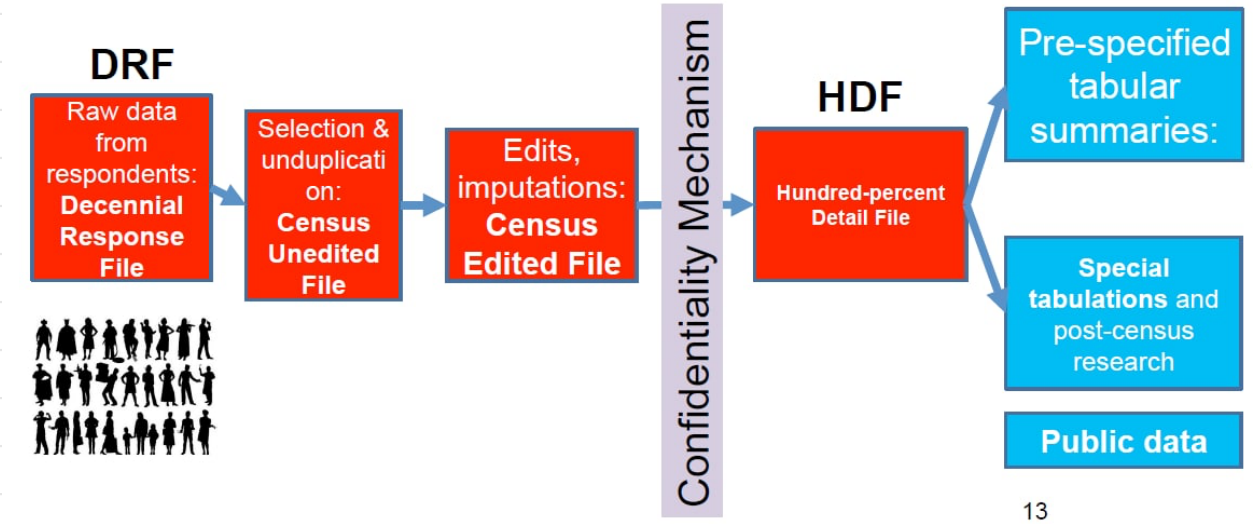
Sensitivity: considering query $q: I \to R$, $S(q)$ is the smallest number s.t. for any neighboring tables D D'
$|q(D) - q(D')| \leq S(q)$ i.e how much can 1 person affect the value of the query?
eg q = how many people in this class have blue eyes? $S(q) = 1$ because 1 person is enough to change the answer from 0.

Alternative to output perturbation may be randomized response: with prob. p return true value, with prob. $1-p$ return flipped value.

DIFFERENTIAL PRIVACY FOR CENSUS

CENSUS: count every person ONLY once grouped by its "characteristic" (age, race, occupacy etc) periodically.

Our aim if Disclosure avoidance.



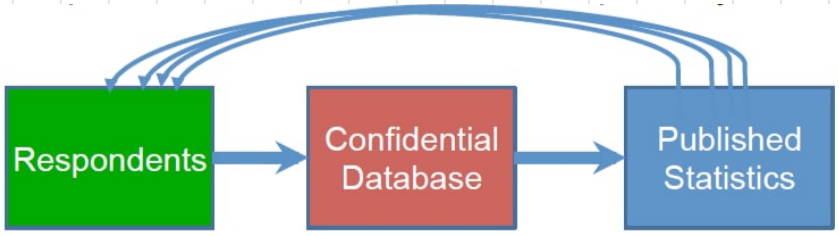
For each person we collect:
- Block
- Sex
- Age
- Race
- Ethnicity
- Relationship

→ 44 bits/person ⟶ 1.7 Gb. for whole USA

⇒ In 2010 census 7.7 billion statistics were collected → 25 per person (lot more than 6)

ATTACKING STATISTICAL DB

Statistical agencies are trusted curators using a trusted curator model that we know may results in some privacy loss after many publications



Publication can be reverse-engineered to reveal the confidential database.
Problem: is hard to know how many statistics is "too many".

Option 1: publish fewer statistics → ✗ we don't know the right threshold
Option 2: less accuracy → ✓ differential privacy (noise)

⇒ Find tradeoff between Accuracy and Privacy loss
⇒ Mathematical definition of privacy loss + mechanism that allow us to add right amount of noise

Main idea is to use "noise" to create uncertainty about private data
Larger the population, lower will be the impact of noise

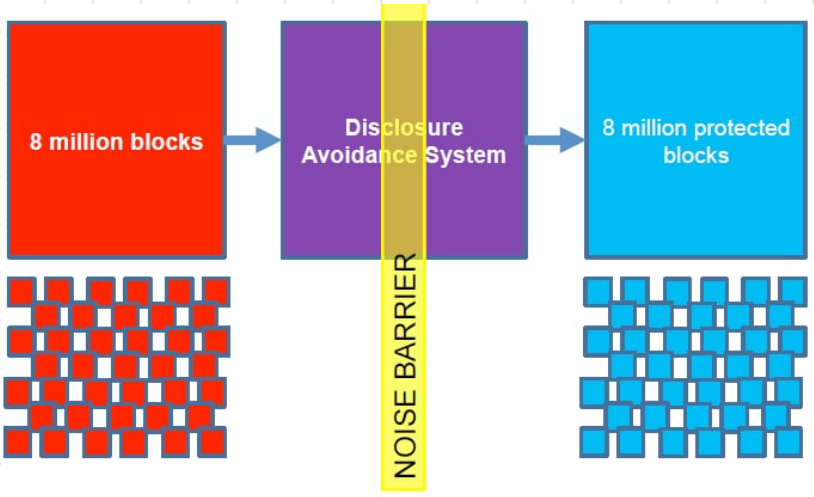
Problem: entire country must be processed at once for best accuracy
↓

2020 CENSUS

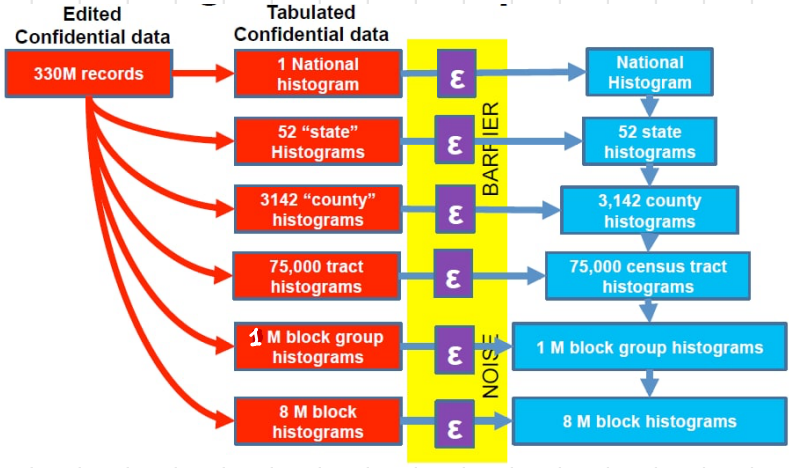
New system that produce higher-quality statistics at more densely populated geographies and consistent tables.
- Every record can be modified without compromise privacy (bounded by global privacy budget)
- No 1-to-1 mapping between publication and confidential data.

ATTEMPT #1: Block-by-Block algorithm (bottom-up)

Indipendently protect each block
Block = smallest geographic unit used for population
Simple to parallelize and privacy cost not depends on # of blocks.
Problem: significant error at higher level and variance of each geounit is proportional to the number of blocks it contains.



NEW ALGORITHM (top-down)



Step 1: generate national histogram without geographic identifier
Step 2: allocate counts to each geography "top down"

$\varepsilon = \varepsilon_{nat} + \varepsilon_{state} + \varepsilon_{country} \dots$

Now:
- What is the correct value of ε?
- Where should the accuracy be allocated (on the graph)?

] → tradeoff (as always!)

ONLINE SOCIAL NETWORKING (OSN)

Everybody knows the good aspects of social networks but there are also bad aspects.
Social networks can be used to spread malware (eg KOOBFACE) also using 3rd party application.
Stalking, harassment etc.
Malicious actors are targeting OSN users to gain information to be used in phishing and other attacks: most of the time users unknowingly share personal information with complete strangers

Information / contents published on osn are public (most of the time) and its hard to completely delete them from internet.

CONTROLLER: company / person who decides how and why personal data is processed.
SENTIMENT POLARITY: infer how much an user like a page / content from its interaction / time spent.

Cambridge Analytica case showed how just using facebook likes the researcher can create very accurate model of persons.

People give up privacy for economic return, immediate gratification (eg free app) overtismated and neglecting future loss

PRIVACY PROTECTION - GDPR

PRIVACY BY DESIGN (PbD)

Approach to system engineering which promotes privacy throghout the whole engineering procen.
Not about data protection but designing so that data dont need protection.
Proactive: plan in advance with privacy as default setting (embedded into design)
Privacy "covers" whole lifecycle. Open and user-centric.

DATA MINIMIZATION

The most important safeguard in protecting personally identifiable information.
In few words what we already said previously: DE-IDENTIFICATION, avoiding to infer personal data from data set or to add unnecessary personal data in the system.
The problem is that this approach is very complicated eg new machine learning techniques are improving RE-IDENTIFICATION day by day. Moreover there are no standard widely accepted!

PbD has been criticized to be "vague" and leaving many open question, similar to a "voluntary compliance".
Most problematic aspect of PbD is the positive sum principle: privacy - security is a positive sum game. This approach is not always supported by experts.
Conclusion: situation is more complicated.

GDPR - GENERAL DATA PROTECTION REGULATION

It considers ALL data on ALL aspects and areas.
Personal data: ANY information relating to a living individual who is, or can be, identified by that information, including data that can be combined with other information.
Data Controller: person/organization who decides the purpose for which data is processed.
Data Processor: person/organization that process data for data controller
Data Subject: owner of data.
Data Protection Impact Assessment (DPIA): describes a process designed to identify risks arising during processing and minimize them as early as possible.

In order to process data, processor must have a lawful basis. This basis is described in several articles of GDPR such as: consent of individual, performance of a contract, legal obligation and so on.

Data processing must comply with 6 principles:
① lawfulness, fairness and transparency
② Purpose limitation: data processed for a single and specified purpose
③ Retention: data kept in an identifiable format not longer than necessary
④ Integrity and Confidentiality
⑤ Data minimization
⑥ Accuracy: only necessary data

If data is anonymised, GDPR does not apply. If data is pseudonymised GDPR does apply.

GDPR also provides tools and techniques to protect data:
- pseudonymization and encryption
- measures to ensure resilience of systems/services processing data
- measures to restore data after breaches
- frequent testing

GDPR allow to impose fines for privacy violation, obligate organizations to demonstrate they are compliant, gives new rigths (eg rigth to be forgotten) etc.

KEY ASPECT FOR BUSINESS

1. One continent - one law: one single set of rule for every company in EU
2. One-stop-shop: business will only have to deal with one single supervisor authority
3. Foreigner companies must follow GDPR rules when offering service inside EU

Note: following some rules not always is good: GDPR has lot of rules to follow and this is not always ok for companies (time/money)
Companies must have a Data Protection Office (DPO). DPO performs DPIAs, implements mechanisms to verify effectiveness.
The head of DPO have to:
- have adequate knowledge of legislation
- carry out its functions in full independence (full access to required resources)
- operate in the employ of the owner or responsible or on the basis of a service contract.

ITALY

In Italy we have two kind of personal data:
Sensitive Data: data revealing race, religion, associations, health, sex etc.
Judicial Data: personal data relating to criminal records.

Regulation in Italy is done by "Garante della Privacy".

Post questions with MY posible onswers. Web Security & Privacy

① What, how and why DPIA? What is DPO?
DPIA is the Data Protection Impact Assessment, part of GDPR schema
Is the procesn of ondysis of possible risks that may orise during dota collection and process and their posible mitigations/solutions. Dato Protection Office is the department who has to evoluate it before dota collection storts.
DPIA must describe scope/context of procesing, onsen complionce meosures, identify risks to individuals and ony odditional countermeosures.
DPO also porforms mechonisms to verify effectiveness of dota protection.
The head of DPO must:
- have knowledge in legislotion obout dota protection
- be fully independent
- operote os the owner/responsible or on the bosis of a service contract

Dota Protection Monoger, the head of DPO will inform ond odvise owner obout its obligotions under Europron Regulotion ond check their implenentotions ond opplicotions.

① What, how, why ARC in emoils?
ARC stonds for Authenticoted Received Chain ond it is on email outhenticotion system.
Forworded nenoges would foil SPF volidotion ond so domoins with strict DMARC policies wont be oble to receive them.
ARC instead ollow multiple intermediories ollowing them to olter the nenoge (bosed on a reputotion system). ARC gives intermediote server o way to sign the original nenoge's volidotion results: even if SPF or DMARC volidotion foil, the receiving service con choose to volidote ARC.
ARC moke use of 3 new mail headers:
- ARC-Authenticotion-Results (AAR) = combinotion of sequence number (i) ond the results of SPF, DKIM ond DMARC volidotion
- ARC-Seol (AS) = combinotion of sequence number (i), DKIM-like signoture of previous AS headers ond the volidity of the prior ARC entries
- ARC-Nenoge-Signoture (AMS) = combinotion of sequence number (i) ond o DKIM-like signoture of entire nenoge except AS header.

Intermediories will:
- copy the "Authenticotion-Results" into o new AAR (storting from i=1) ond prepend to nenoge
- colculote AMS (with AAR) ond prepend to nenoge
- colculote AS of previous AS headers ond prepento to nenoge.

Recipient will:
- volidote AS chain
- volidote lost AMS

(2) Tor circuit creation and why it works?

Tor network is composed by Onion Routers which route traffic in the circuit and Directory Servers which mantain a list of active O.R., their location, public key etc.
When creating a circuit, source node contacts a DS asking a list of available Tor nodes. From it, a random path to destination, composed by 3 hops, is built. Source node shares a key (AES) with each middle node and message is encrypted as many times as the hops are. Each node will remove a single layer of encryption keeping message safe until last node. Every router only knows its predecessor and successor.
The key point of this technology is that only the user knows the origin and destination of the circuit.

(2) BTC double spending and forks

Double spending is the act of spending a single amount of value (in this case BTC) multiple times eg owning 1 BTC, we send 1 BTC to two different addresses. (2 BTC from 1 BTC → impossible)
Blockchain helps to solve this issue: if both transaction are submitted at same time a "fork" of the blockchain happens i.e. blockchain is splitted. After multiple confirmations from other nodes, the one that have most will be accepted, the other discarded.
BTC proof-of-work requires to solve a cryptographically hard problem before publishing a new block (lot of time needed). This is the reason why if someone would be able to overcome 51% of whole network power it would be possible to get control of block creation.
Fork can also happen for other reasons:
HARD FORK = changes in blockchain validation rules needing an update of all nodes, if an huge amount of them won't update a permanent split happen → 2 different blockchains (SPLITCOINS)
SOFT FORK = changes are backward compatibles but more limited.

(3) 3 types of XSS and example of stealing session with XSS

Cross Site Scripting (XSS) is a kind of attack in which attacker inject dangerous code inside a webpage that will be executed when user will visit it.
STORED XSS: webpage, not well developed, allow attacker to inject malicious code into DB. User may later visit the webpage with some requests and occur in the script previously uploaded by attacker
REFLECTED XSS: webpage doesn't perform any processing on HTTP request → attacker may induce the user to click on a link containing the script that will be executed when received by the webpage
DOM BASED XSS: attacker is able to inject script directly into HTML
XSS can also used for session hijacking: malicious script running on victim's browser is able to send session token (cookie) to the attacker that will be able to login as it was the victim.

(3) K-anonimity weaknesses and other solutions

K-ononimity refers to the techniques that aim to protect privacy of individuals when publishing data: generalization (replace quasi identifier with semantically consistent values) and suppresion (completely remove quasi identifiers) grant that personal data cant be distinguished from at least K-1 individual i.e sensible values must appear at least K times in public data.
However K-anon. does not provide privacy if is still possible to "combine" multiple K-anon. tables and usually real world data sets are very sparse → K-anon. datasets useless.
Other solutions:

- l-diversity = each equivalence class must have at least l distinct values of sensitive attributes to avoid attackers to infer informations
- t-closeness = the distance between the distribution of sensitive attributes within each equality class and their distribution in original database shold be close to a threshold t.

(4) How to store BTC? Discuss about mixing.

BTC and crypto currency in general are stored in digital wallets.
HOT wallet: basically are online wallets that are connected to internet. Good for keeping small amount of coin for their simplicity of use but lack in security. Keys are maged by a third party ("not your key, not your money") and this may lead to fraud
COLD wallet: safest storage method, a wallet not connected to internet (hardware). They store address and private key. Cons: machine broke, money lost.
BTC mixing is the act of using intermediaries to send BTC to a destination address hiding your trace: tokens from source are MIXED with others to avoid the possibility of traceback the blockchain.

(4) Discuss TOR hidden services

Tor makes possible to deploy a server on the internet that anyone can connect to without knowing where it is or who runs it.
How:

1) Bob generates public key to identify his service, chooses some entry points from available nodes and advertises them also providing the public key
2) Build a circuit to each of them and tell them to wait requests
3) Alice in some way learns about the existance of Bob's service, choose a rendezvous point and a secret cookie, then opens a connection to one of the entry point notifying the RP, the cookie and send them to Bob (encrypted with Bob's public key).
4) Bob connects to RP sending back the cookie as identification and concludes the handshake
5) Bob and Alice have now established an anonymous stream and can communicate as normal.

⑤ Discuss DMARC.

Domain-based Message Authentication, Reporting and Conformance is a standard that helps protect email senders and recipients from spam, spoofing and phishing. Its built over SPF and DKIM.

1) Domain admin publish a policy defining email authentication practices what to do with the one who fail that check.

2) When mail server receive an email it checks:

- DKIM signature validity
- if the sender IP is allowed by domain to send the message
- DOMAIN ALIGNEMENT:
  - SPF's FROM domain = return path domain
  - DKIM's FROM domain = "d" field of DKIM header

3) Decide if accept or not

4) After checking DMARC the results are sent to sending domain owner.

⑥ DKIM and its exploitation by DMARC

Domainkey Identified Mail, standard for specifying how to cryptographically sign messages permitting to signing domain to to claim responsability for it.

Receiving domain will query the signing one for its publickey to check signature of message When a message is received, "d" field, the signing domain, is extracted from DKIM header, then signing domain DNS is queried for the public key.

+ Answere ⑤

⑤ Differential privacy: pro, cons and where it is useful.

DP is an approach to privacy regarding published data. It consists of learning nothing about a single individual while learning useful statistical information about a population.

Key point of DP is the fact that the outcome is indistinguishable regardless whether a single record is present or not. How to achieve this? Output Perturbation i.e. adding noise before publishing data, decide to change single records at random.

Pro of DP is that is impossible to infer information of a single individual also using external source, while the cons are:

- since we add noise, to have a dataset representing real world we need lot of more data
- deciding how much noise to add is hard (the more noise the more privacy but less valuable data)

DP was used for US census 2020 to avoid disclosure of personal data

⑥ Differential privacy and US Census 2020.

Answer ⑤ +

During US census 2020, government had to find a solution to information disclosure and it was found in DP. The problem was to find a proper way to apply it at national level, with densely populated geographies.

The TOPDOWN mechanism used consists of splitting the privacy-loss budget ε in 6 pieces: ε_{nat}, ε_{state} ... (1 for each geography unit kind), start from national histogram H_0 using ε_{nat}, proceed to state histograms with $\sum_{\varepsilon_{state}} H_s = H_0$ and recurse down the hierarchy.

This algorithm ensures that error does not increase with number of contained Census blocks and improve lower levels of geography eg. for sparsity.

⑦ Discuss CSRF.

Answer ③ +

Cross Site Request Forgery is a particular type of XSS attack in which the victim is the application itself and the user is the attack vector.

A vulnerable website may have too much "trust" on authenticated users and may not validate their requests. Attacker can build a fake request that mimics a valid user request and lure an authenticated user to send the request to web server. Application noting the fact that the user is authenticated will accept the request.

A possible solution is to ensure that forms require some sort of additional random token on every request submission.

⑧ Discuss SQL injection.

SQL injection is part of Injection flaws attacks. When a web application takes user input without validation into a database, server shell command or operating system calls, the attackers may append additional information that may be read as command by server, allowing all kind of action over data.

SQL injection in particular allows to execute malicious SQL code inside database

Injection is #1 in OWASP top 10 critical risks.

Possible countermeasures in addition to input validation are:

- parameterized queries = structure of queries is already defined and can't be altered, in this case injection will be handled as invalid input/error
- stored procedures = more complex validation at server side (increasing its load).

⑨ Third party cookies, technical aspect of tracking users behaviour.

Cookies are small pieces of information stored at client side that allow HTTP protocol to be stateful. In particular 3rd party cookies are received from contents hosted inside visited web pages eg. advertisement.

Cookies contain multiple fields: name and value (mandatory) + expiration date, path, domain etc. Third party cookies enable user tracking i.e determine web users interests and behaviour aiming to provide most personalized advertisement.

How:

- connect to site A that hosts site B (advertisement server)
- (automatically) ask to connect to B to download them eg images
- B knows the referer (if no https is used) i.e A website and can send back cookies with sensitive informations

B now knows that you visited A website and will notice you whenever you will visit again a website containing B contents.

This can be useful for normal websurfing since targeted adv. may be good for the user but may also result in leak of personal data (employment status, sexual orientation, financial problems, health conditions etc)