CS459/698 Privacy, Cryptography, Network and Data Security

Network Steganography and Information Hiding

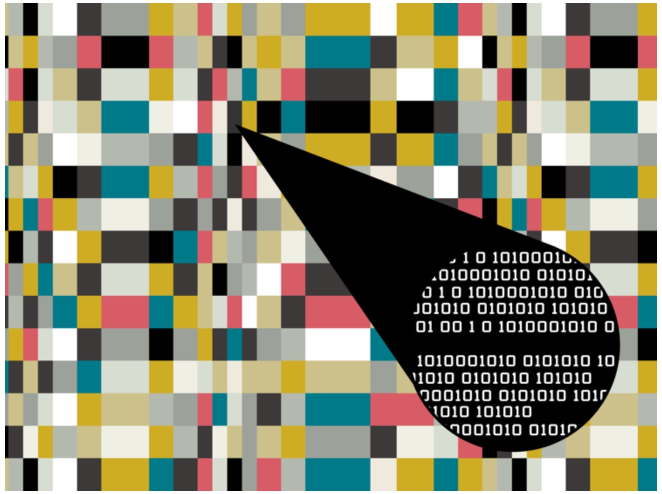
Fall 2024, Tuesday/Thursday 02:30pm-03:50pm

Definitions

Steganography

- Art and science of communicating in a way that hides the existence of a message
    - From the Greek words *steganos* and *graphy*

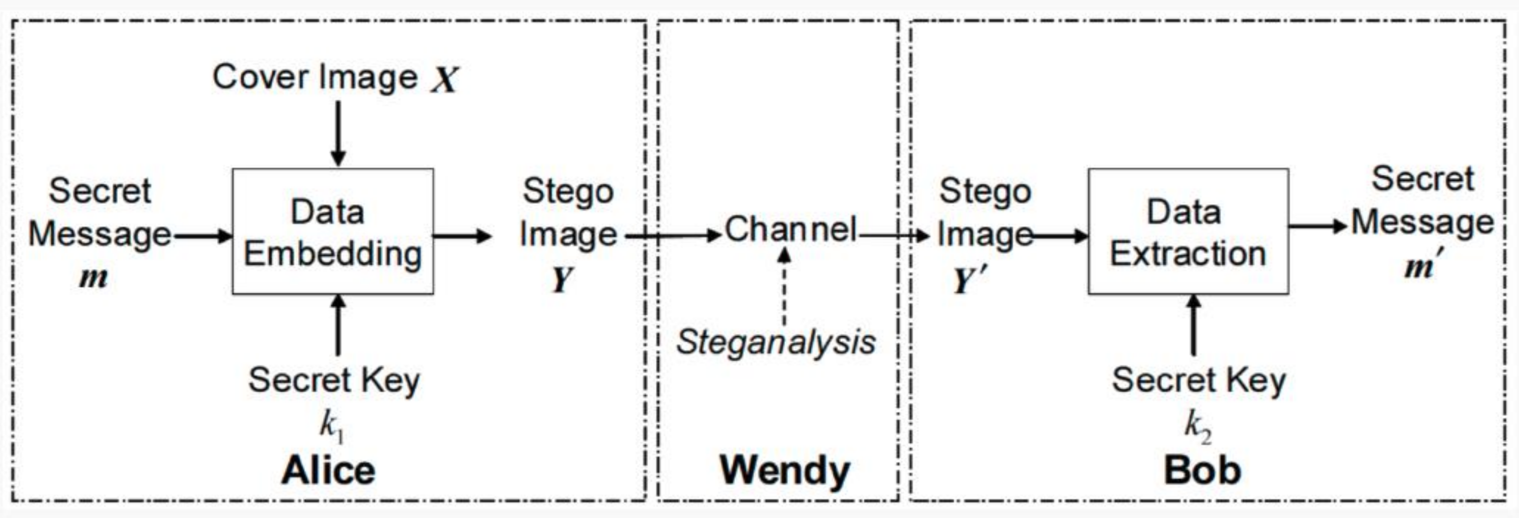
- Steganography takes one piece of (*secret*) information and hides it within another (*carrier / cover*)



Cryptography vs. Steganography

- **Cryptography:** <u>protects</u> the contents of messages
    - You will still be suspected by the authorities for sending a coded message.
- **Steganography:** <u>conceals</u> the existence of messages
    - You can hide the fact that communication is going on at all.

Steganography system model



- Wendy can be seen as a warden, and can be:
    - **Passive:** attempts to detect whether Y carries secret content
    - **Active:** modifies stego image Y into Y’ in hopes of destroying the secret content

Why are we studying covert channels?

- Transfer sensitive/unauthorized information through a channel that is not supposed to transmit that information
    - Makes it more difficult to detect data exchanges

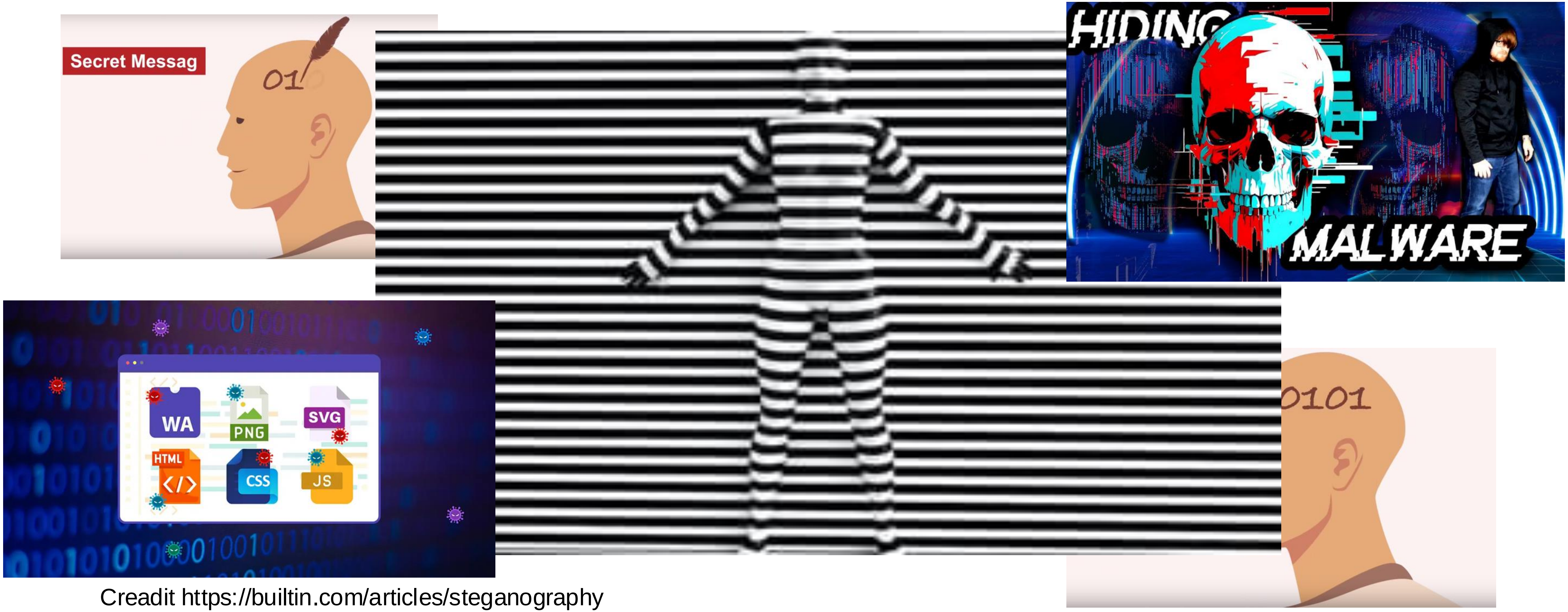
Croissant-based covert channel

Why are we studying covert channels?

The alleged Russian spies arrested by the FBI are accused of encoding messages into otherwise innocuous pictures, marking the first confirmed use of this high-tech form of data concealment in real life.

According to the FBI, this picture has the map of an airport hidden in its data.

The history of Steganography



Creadit https://builtin.com/articles/steganography

Why should we care?

- Corporate espionage
- Government or military activities
- Criminal activities
- Censorship circumvention
  - Whistleblowers, and journalist in danger zones.
  - Protecting free speech and free press
  - Exfiltrating data out of “secure” environments

Covert channel

- A covert channel is a path for the illegal flow of information between subjects within a system, utilizing system resources that were not designed to be used for inter-subject communication.
  - A path of illegal information flow using mediums not intended for communication” (a liberal paraphrasing)
  - (general) Communication through a medium that violates a global security policy without violating local ones.
  - “two human users talking over coffee is not a covert channel”

Overt channel

- This is an example of an overt channel: communication channels being used as intended.
  - ➢ downloading web content from a public web server
  - ➢ using emailing to submit a class assignment to your TA
  - ➢ talking to a relative on the telephone
  - ➢ waving to a friend

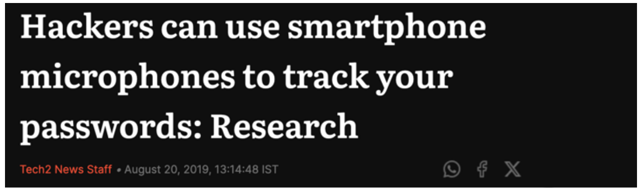
Types of covert channel

- Several dimensions to be considered:
    - Local vs. remote
    - Storage vs. timing
    - Noisy vs. Noiseless

- Important characteristics:
    - Bandwidth: how many bps can be transmitted through the covert channel?
    - Noise: Is the information transmitted through the covert channel distorted in any way?

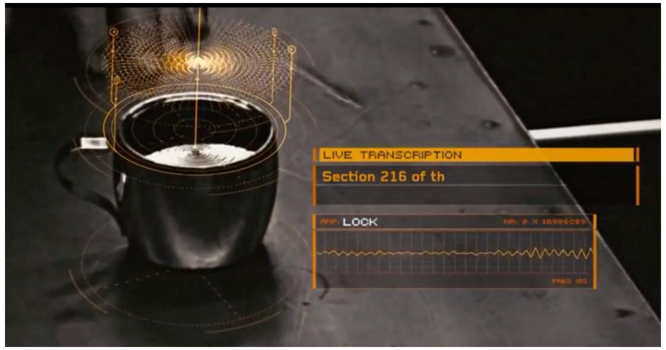
Local vs. remote covert channels

- Local covert channels leverage a machine’s shared resources:
    - CPU, RAM, Disk...
- Remote covert channels leverage transmission mechanisms
    - Typically the network (but also others...)



Hackers can use smartphone microphones to track your passwords: Research

Tech2 News Staff • August 20, 2019, 13:14:48 IST



ETHERLED: Air-gapped systems leak data via network card LEDs

By Bill Toulas

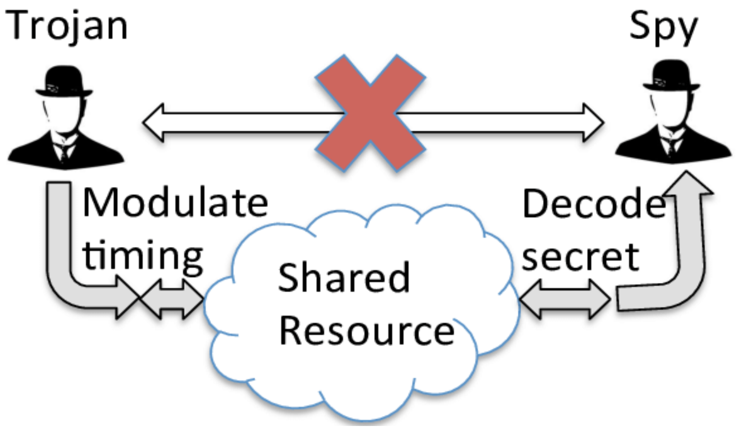
August 23, 2022 07:28 AM

Covert timing channels

- To use a covert timing channel:
  - Both sender and receiver must have access to some attribute of a shared object.
  - Both sender and receiver have access to a time reference (real-time clock, timer, events order).
  - The sender must be able to control the timing of the detection of a change in the attribute of the receiver.

Both use the same network

receiver knows to watch his local network traffic starting at the top of every hour, for 5 minutes.



sender wants to share the following Message:
“Dogs barking. Can't fly without umbrella”

Covert timing channels

D	o	g	s		b	a	r	k	i	n	g	.	C	a	n	'	t		f	l	y	

...

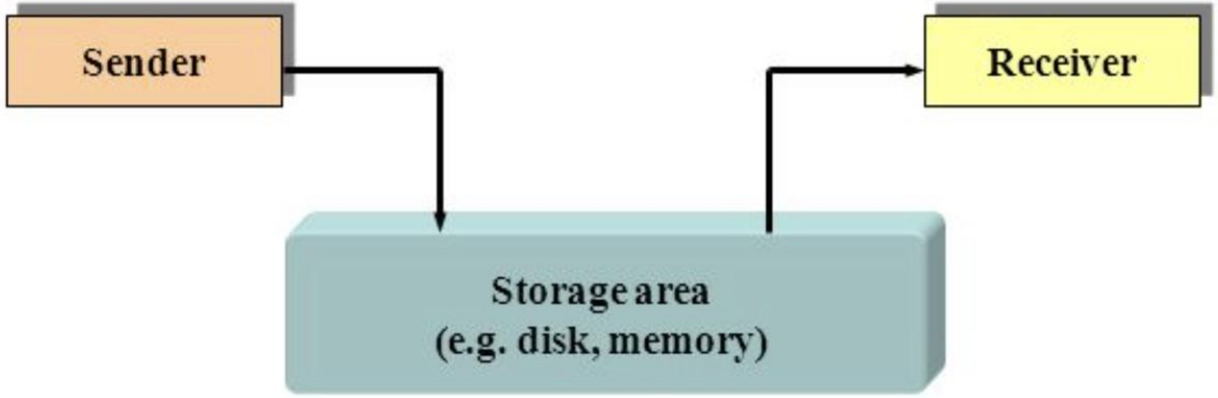
68	111	103	115	32	98

...

- The sender converts his message into ASCII decimal values:
  - The sender pings (the ICMP one) Receiver's computer 68 times, then 111, then 103, ….
  - The receiver observes the network, counts the pings, and looks up the values in an ASCII table to recover the text.

Covert storage channels

- To use a covert storage channel:
    - Both sender and receiver must have access to some attribute of a shared object.
    - The sender must be able to modify the attribute.
    - The receiver must be able to view that attribute
    - A mechanism must be in place for initiating the sender and receiver processes, and there must be a way to sequence their accesses to the shared resource (e.g., sync header)



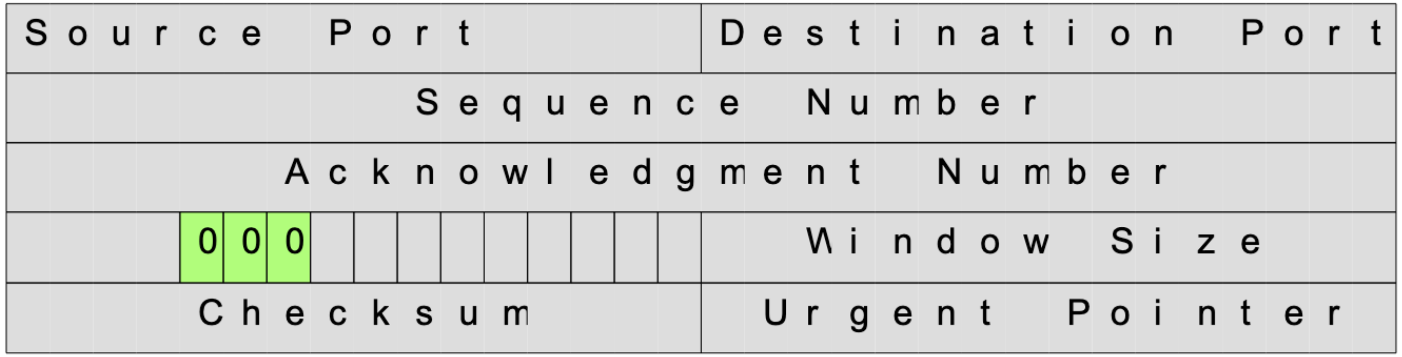
Covert storage channels

- This time the receiver is in a foreign country, on a network that blocks the ICMP protocol, so the timing channel won’t work

- The receiver is in another time zone, and the two won't be online at the same time
  - The sender must leave the receiver the message via some Storage Channel, so he may retrieve it later, when he is online.

Covert storage channels

- This sender runs a web server. The receiver has access to the server via HTTP, and can download pages without raising suspicion.
  - The sender sticks are reverse proxy in front of the web server that modifies outgoing TCP packets to store custom bit patterns in a reserved (unused) field of the packet header.
  - As these bits are usually ignored, they will remain when sent
  - The receiver connects to the server and uses a plug-in that reads these bits to reconstructs the message.

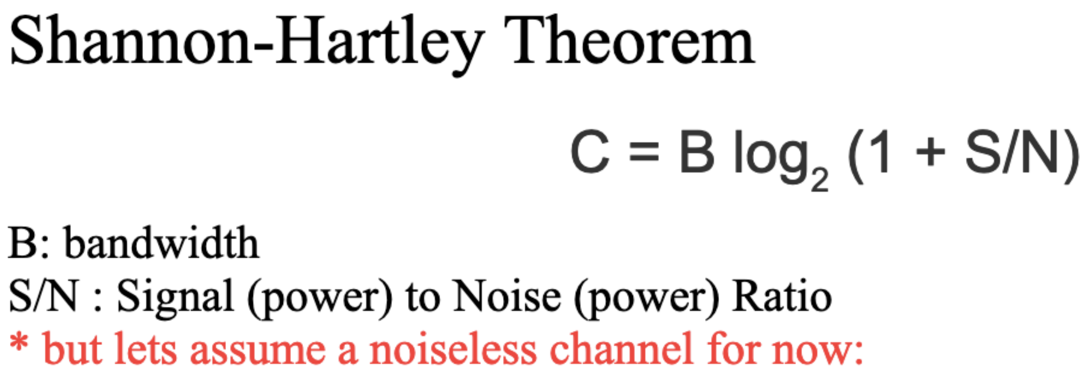


3 bits can only hold values 0 through 7

What can be done?

Covert storage channels

- This sender and receiver must mitigate an issue with the channel’s bandwidth (capacity).



Shannon-Hartley Theorem

$$C = B \log_2 (1 + S/N)$$

B: bandwidth
S/N : Signal (power) to Noise (power) Ratio
* but lets assume a noiseless channel for now:

  - The sender Packets from Web Server: 10 packets / sec
  - The sender’s storage channel capacity: 3 bits * 10 packets / sec = 30 bits / sec
  - “D” = 68, needs a minimum of 7 bits to send.

Can’t we just get rid of covert channels?

- It is typically infeasible to eliminate every potential covert channel in a (networked) computer system, but we can:
  - Eliminate them by modifying the system implementation.
  - Reduce their bandwidth by introducing noise into the channel.
  - Monitor for usage patterns that indicate someone is trying to exploit a covert channel.

- Not-So-Obvious Covert Channels
  - “a” could mean: “a” , “61”, “Attack at dawn”, “Get to the embassy asap”
  - Pre-shared knowledge between sender and receiver dictates the contents of the channel

Detecting covert channels

- Anomaly detection often fails
  - Enumerating all “good” behavior is difficult, and expensive
  - Covert Channels are design to “look like” overt channels.
  - False Positives are expensive
- Network malfeasance is often a reaction to defensive capability.
- Network defense is often a reaction to detected malfeasance.

Adversaries, usually have a time advantage, unless the defender can see into the future.

Defenders also typically have more operational constraints. Adversaries rarely “play by the rules”

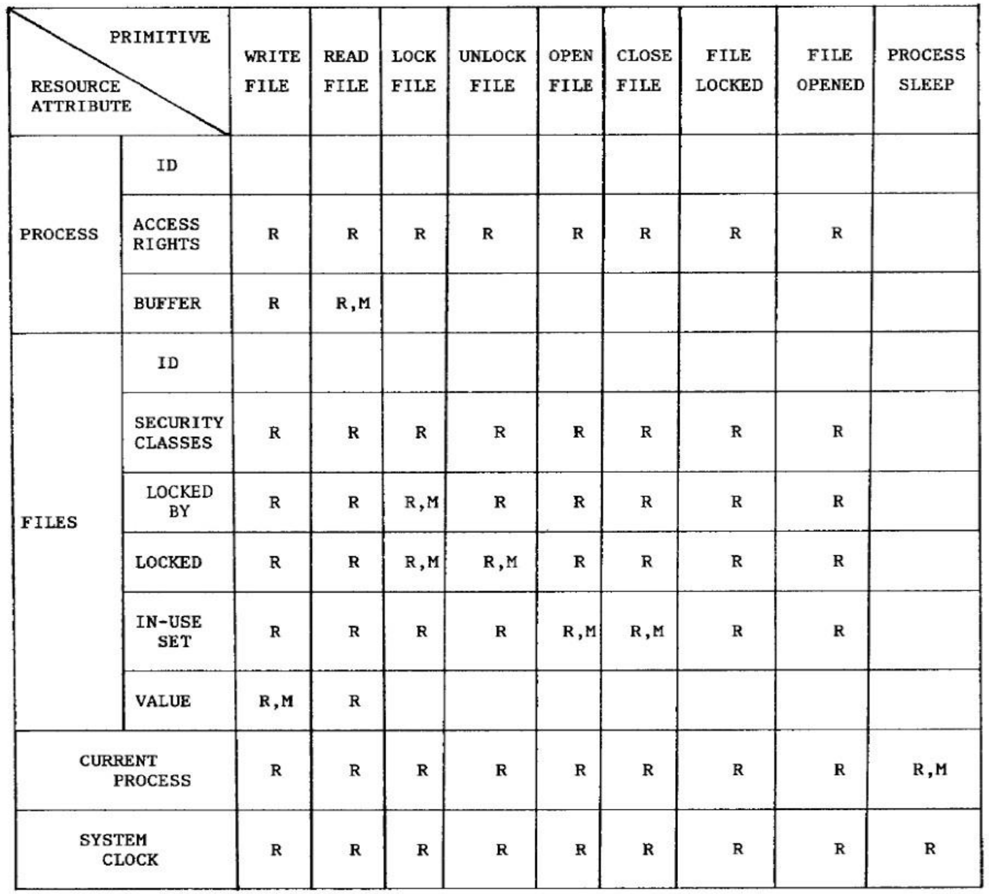
Defenders often times, do or must.

Some attempts at detection

- Kemmerer’s Shared Resource Matrix

All shared resources that can be referenced or modified by a subject are enumerated, and then each resource is carefully examined to determine whether it can be used to transfer information from one subject to another covertly.

  - Systematic way to investigate potential covert channels
  - Requires substantial knowledge about the semantics and implementation of system operations.

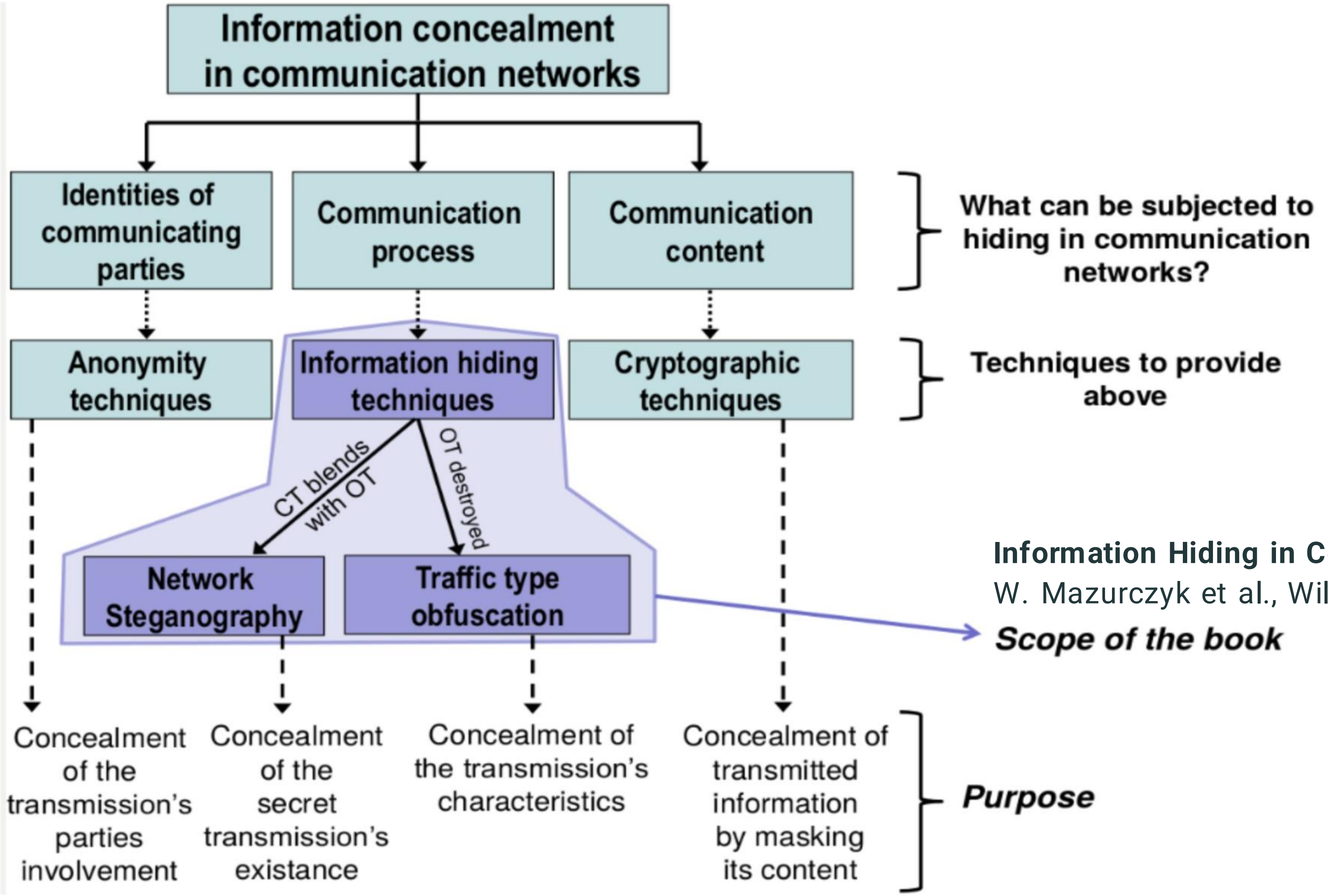


RESOURCE ATTRIBUTE	PRIMITIVE	WRITE FILE	READ FILE	LOCK FILE	UNLOCK FILE	OPEN FILE	CLOSE FILE	FILE LOCKED	FILE OPENED	PROCESS SLEEP
PROCESS	ID									
	ACCESS RIGHTS	R	R	R	R	R	R	R	R	
	BUFFER	R	R,M							
FILES	ID									
	SECURITY CLASSES	R	R	R	R	R	R	R	R	
	LOCKED BY	R	R	R,M	R	R	R	R	R	
	LOCKED	R	R	R,M	R,M	R	R	R	R	
	IN-USE SET	R	R	R	R	R,M	R,M	R	R	
	VALUE	R,M	R							
CURRENT PROCESS		R	R	R	R	R	R	R	R	R,M
SYSTEM CLOCK		R	R	R	R	R	R	R	R	R

https://dl.acm.org/doi/pdf/10.1145/357369.357374

Network Information Hiding

Information hiding in the network


Information concealment in communication networks
Identities of communicating parties
Communication process
Communication content
What can be subjected to hiding in communication networks?
Anonymity techniques
Information hiding techniques
Cryptographic techniques
Techniques to provide above
CT blends with OT
OT destroyed
Network Steganography
Traffic type obfuscation
Concealment of the transmission's parties involvement
Concealment of the secret transmission's existance
Concealment of the transmission's characteristics
Concealment of transmitted information by masking its content
Purpose

Information Hiding in Communication Networks,
W. Mazurczyk et al., Wiley 2016

Scope of the book

Network Information Hiding

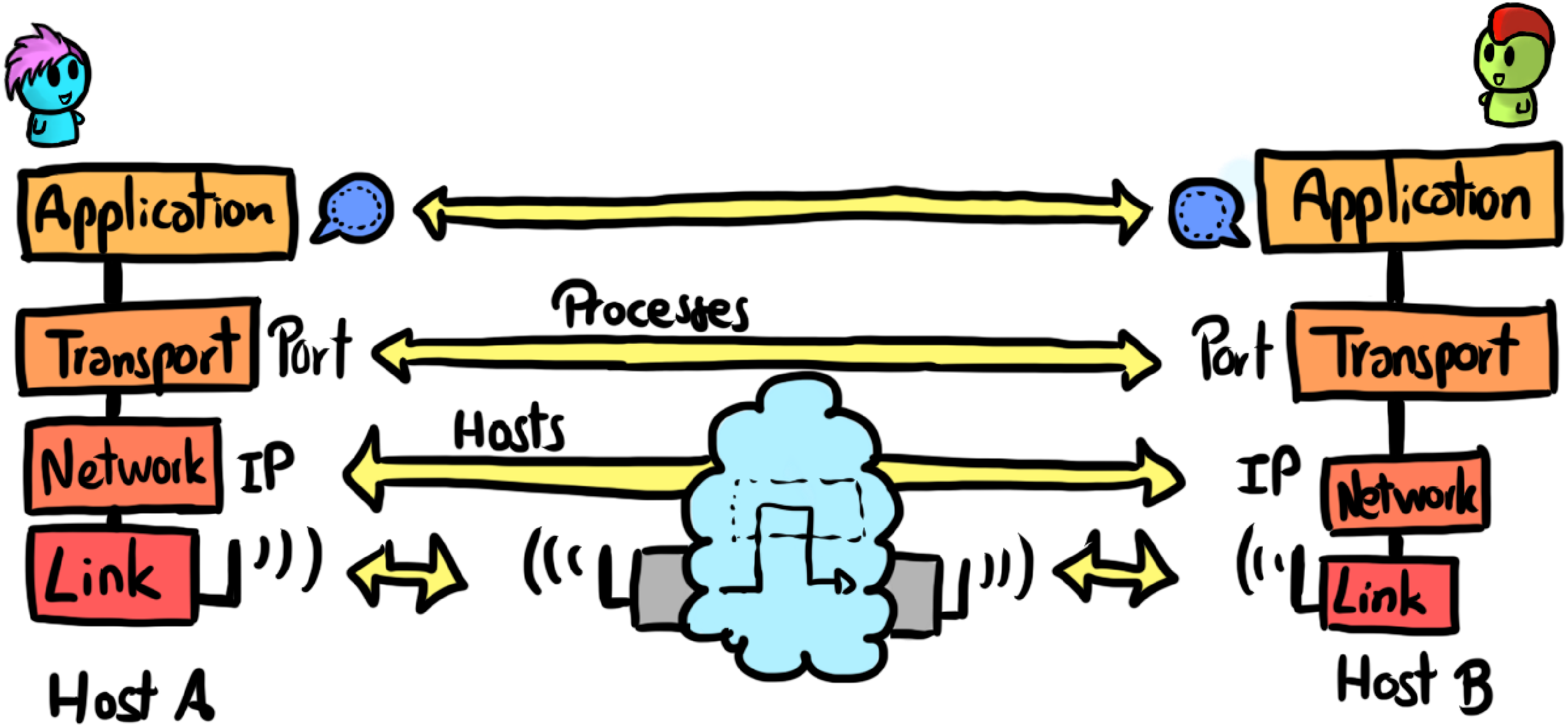
Network covert channels

How do we create a network covert channel?

- Storage
  - e.g., packet header manipulations
- Timing
  - e.g., timing between packets
- What about steganography?
  - We may say that steganographic methods are used to create a network covert channel
- In a network covert channel:
  - Covert data is hidden in overt network transmissions
  - The “cover” medium is called a “carrier”

OSI Layers

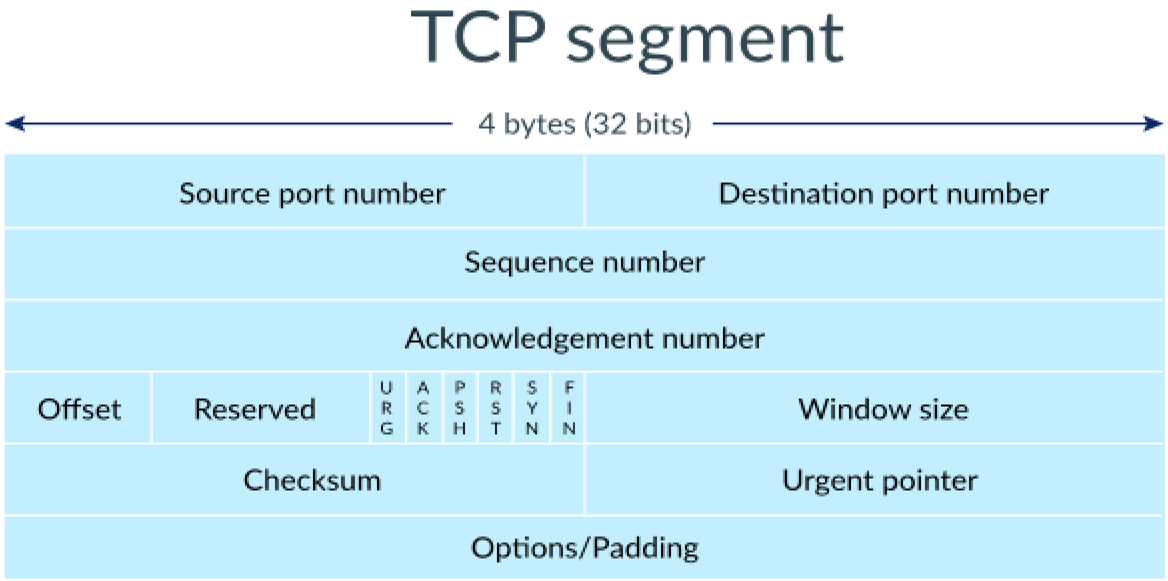
- Where can we implement covert channels across the OSI stack?



Covert storage channels on TCP/IP

- TCP/IP packets have headers that provide extra information
  - Headers have different fields that are optional or disregarded in usual transmissions

- These fields can be used for hiding information!
  - IP identification
  - Offset
  - Options
  - TCP Checksum
  - TCP Sequence Numbers



TCP segment

4 bytes (32 bits)

Source port number | Destination port number

Sequence number

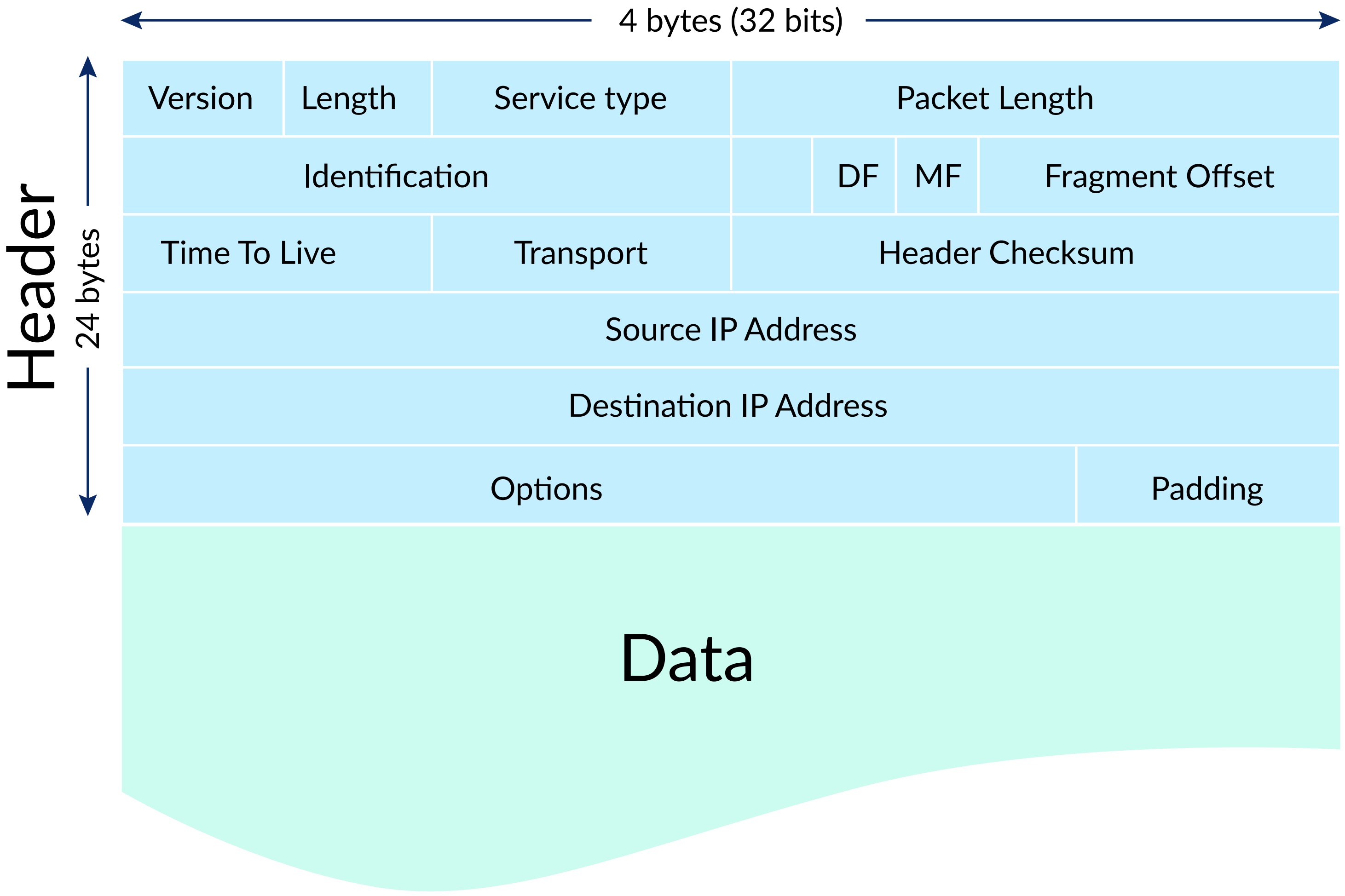
Acknowledgement number

Offset | Reserved | URG | ACK | PSH | RST | SYN | FIN | Window size

Checksum | Urgent pointer

Options/Padding

IP Header



4 bytes (32 bits)

Header

24 bytes

Version | Length | Service type | Packet Length

Identification | | DF | MF | Fragment Offset

Time To Live | Transport | Header Checksum

Source IP Address

Destination IP Address

Options | Padding

Data

Covert storage channels on IP

- IP ID: a value assigned by the sender to aid in assembling a packet’s fragments

- Detection approaches:
  - OpenBSD toggles the most significant bit of the IP ID every 3 minutes or 30,000 IP IDs, so the MSB can be examined to check if it matches this pattern.
  - Within a rekey interval, the OpenBSD IP ID is nonrepeating

Embedding Covert Channels into TCP/IP, Murdoch and Lewis, International Workshop on Information Hiding, 2005

Covert storage channels on IP

- It uses SHA-1 to hash a block of 16 32-bit words.
- Words 9–11 set to the source and destination IP address and port..
- The remaining 13 words filled with a cryptographically secure, random secret, initialized on boot.

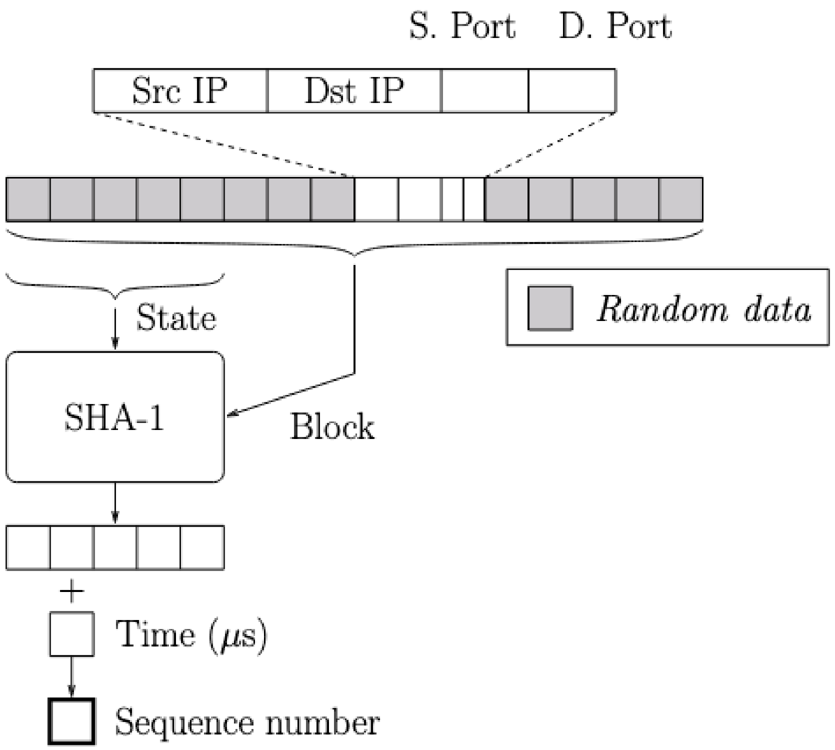


Fig. 2. Linux 2.0 ISN generator

Covert storage channels on IP

To limit the impact of secret compromise.

- The random data is rekeyed every 300 seconds (5 minutes)
  - ➢ MD4 algorithm is insecure
  - ➢ The MSB is replaced with a counter incremented on rekeying and initialized to the current time divided by 300

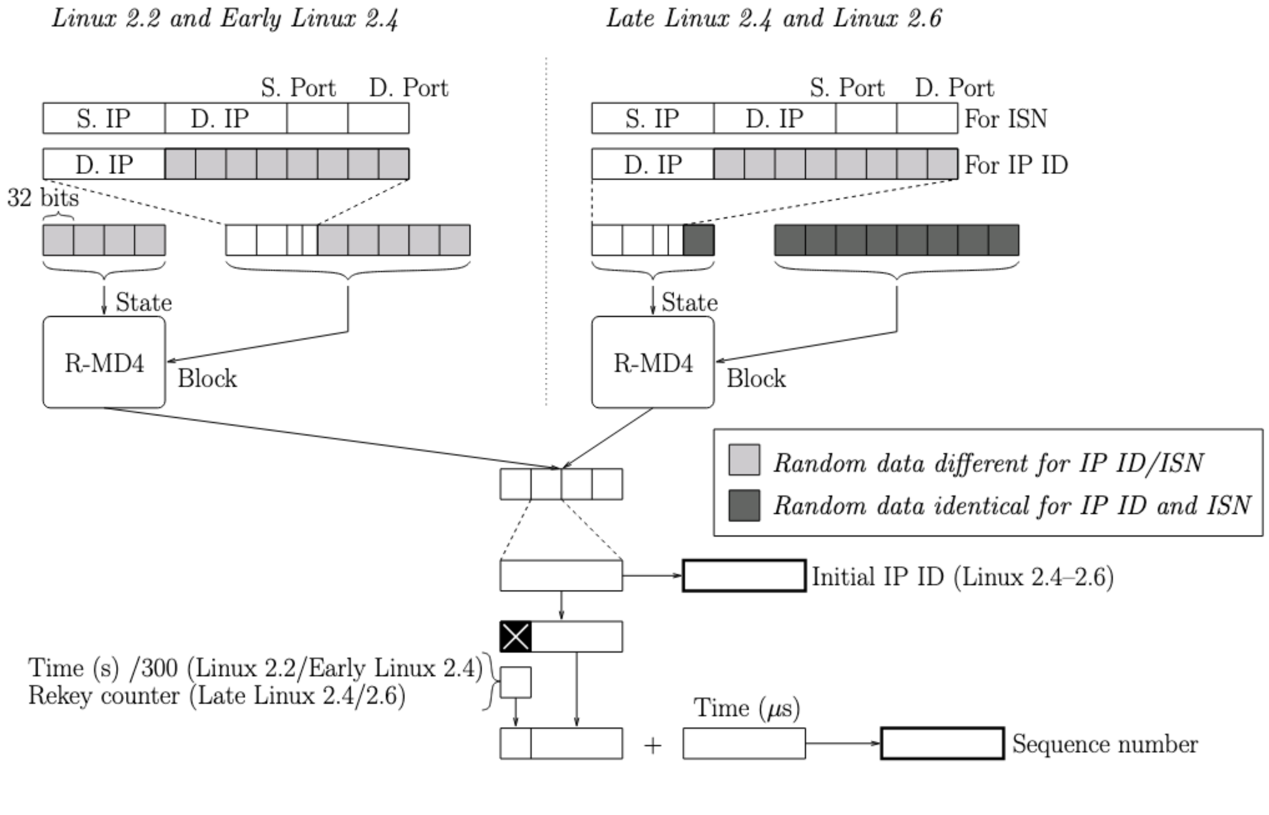

Linux 2.2 and Early Linux 2.4
Late Linux 2.4 and Linux 2.6
S. Port
D. Port
S. IP
D. IP
For ISN
For IP ID
32 bits
State
R-MD4
Block
Random data different for IP ID/ISN
Random data identical for IP ID and ISN
Initial IP ID (Linux 2.4–2.6)
Time (s) /300 (Linux 2.2/Early Linux 2.4)
Rekey counter (Late Linux 2.4/2.6)
Time (μs)
+
Sequence number

Fig. 3. Linux 2.2–2.6 ISN generator and Linux 2.4–2.6 IP ID generator

Covert storage channels on TCP/IP

- TCP ISN: initial sequence number on TCP connections

The MSB set to either ‘1’ or ‘0’, depending on weather OS is in an ‘odd’ or ‘even’ rekey interval

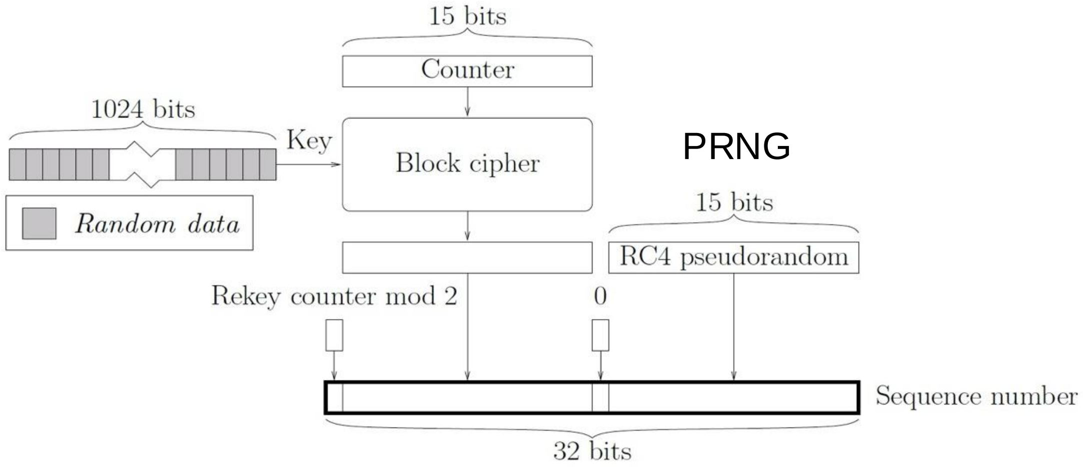


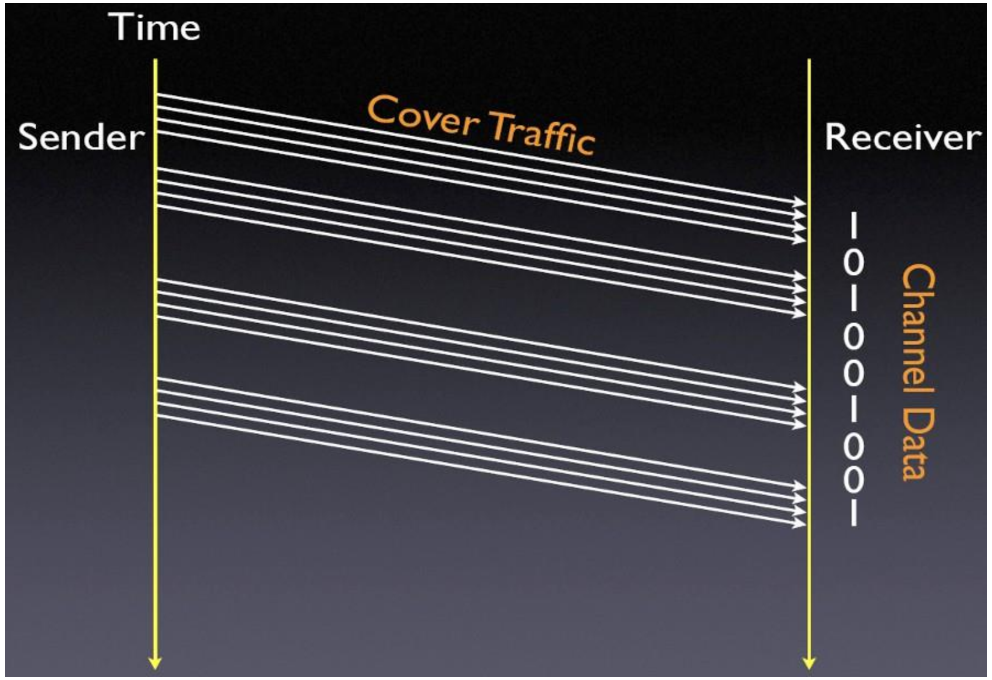
Fig. 4. OpenBSD ISN generator

- Several constraints make steganography easily detectable

➢ IP ID Characteristics, TCP ISN Characteristics, Other Anomalies(e.g., excessive fragmentation/re-ordering etc …)

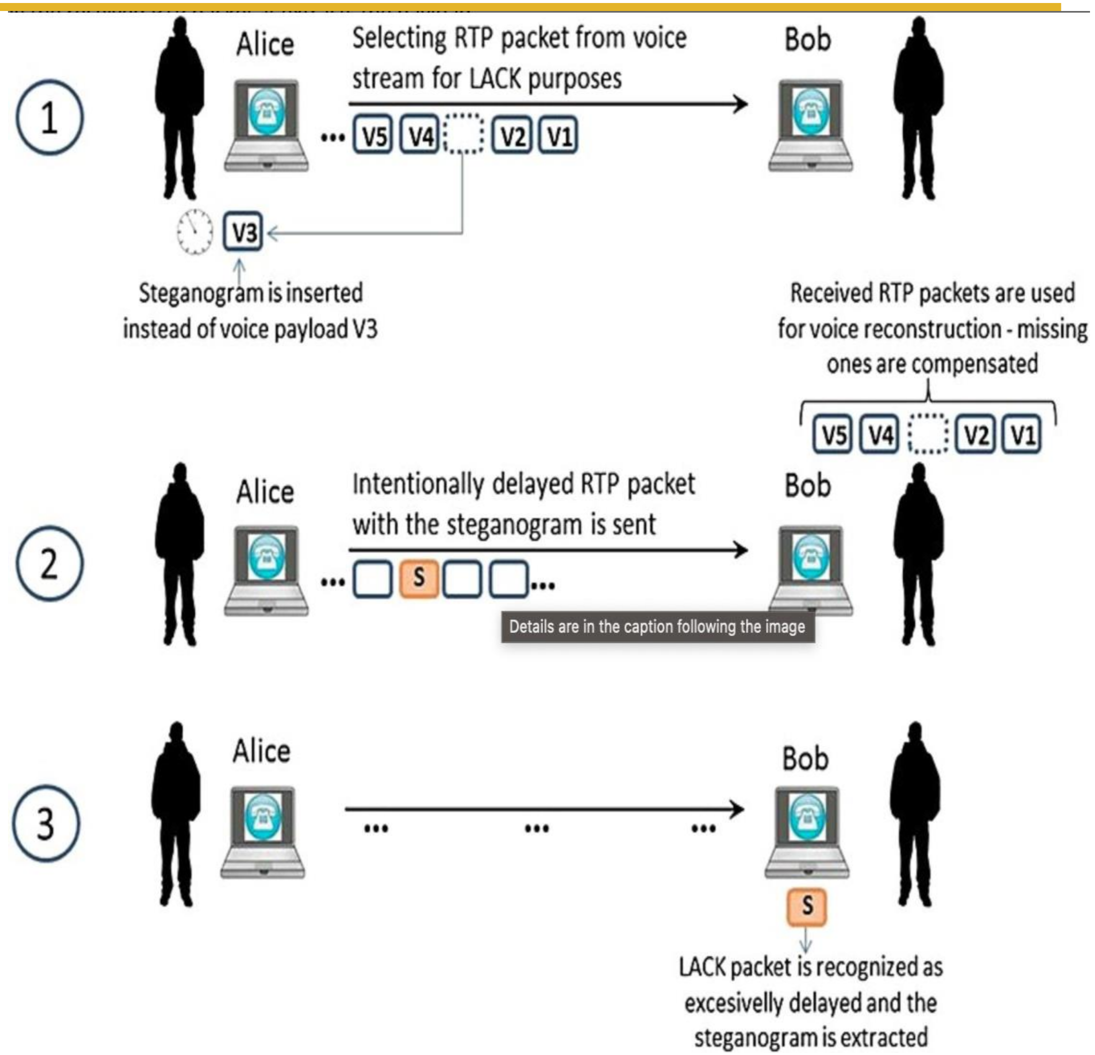
Covert timing channels on TCP/IP

- These typically propagate covert information by crafting delays between certain events
    - e.g., modify usual inter-packet delay, introduce losses by skipping sequence numbers



son TCP/IP

- We may also have hybrids of storage and timing (e.g., LACK*)
  - Replace encrypted packet contents with covert data and use delays for signaling the receiver about specific packets
  - Sending an RTP packet slightly earlier or later can signify a '1' or '0', respectively


1
Alice
Selecting RTP packet from voice stream for LACK purposes
Bob
V5
V4
V2
V1
V3
Steganogram is inserted instead of voice payload V3
Received RTP packets are used for voice reconstruction - missing ones are compensated
V5
V4
V2
V1
2
Alice
Intentionally delayed RTP packet with the steganogram is sent
Bob
S
Details are in the caption following the image
3
Alice
Bob
S
LACK packet is recognized as excesivelly delayed and the steganogram is extracted

*Latency Attack on Covert Channels

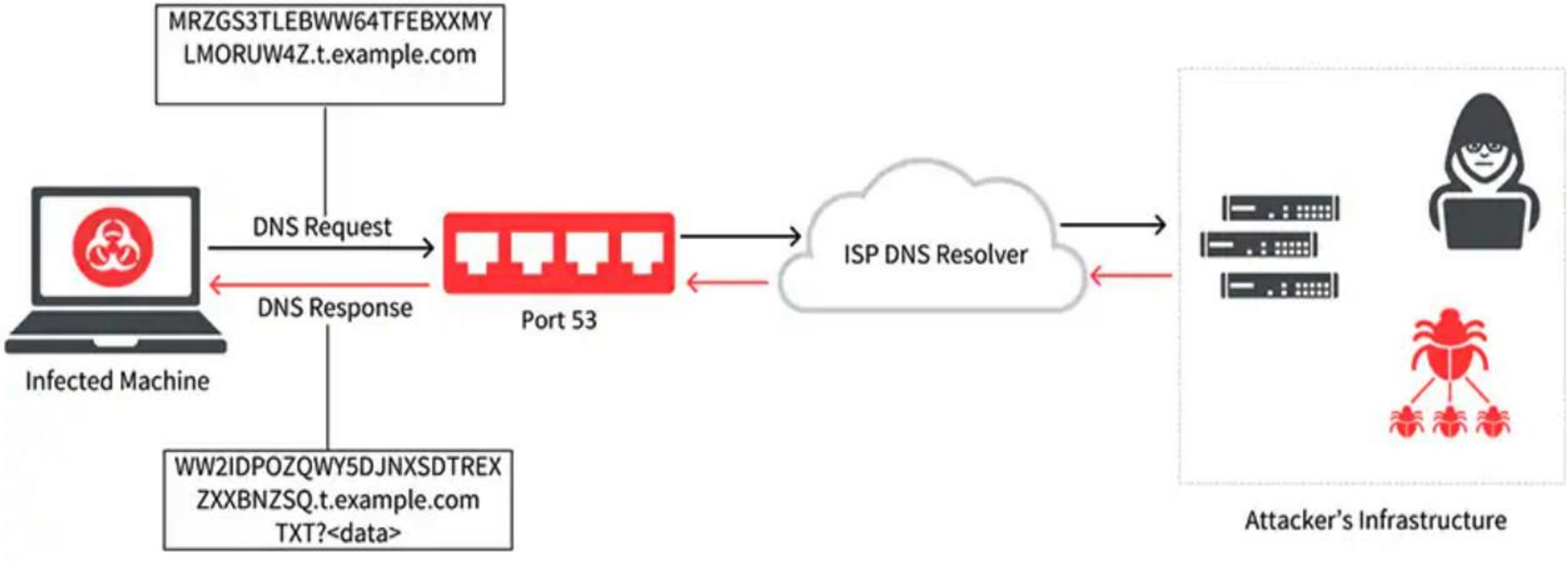
Covert channels at the application level

- Many examples:
    - HTTP
    - DNS
    - Games
    - VoIP/video traffic
    - Push notifications
    - ...

Example: DNS Tunneling

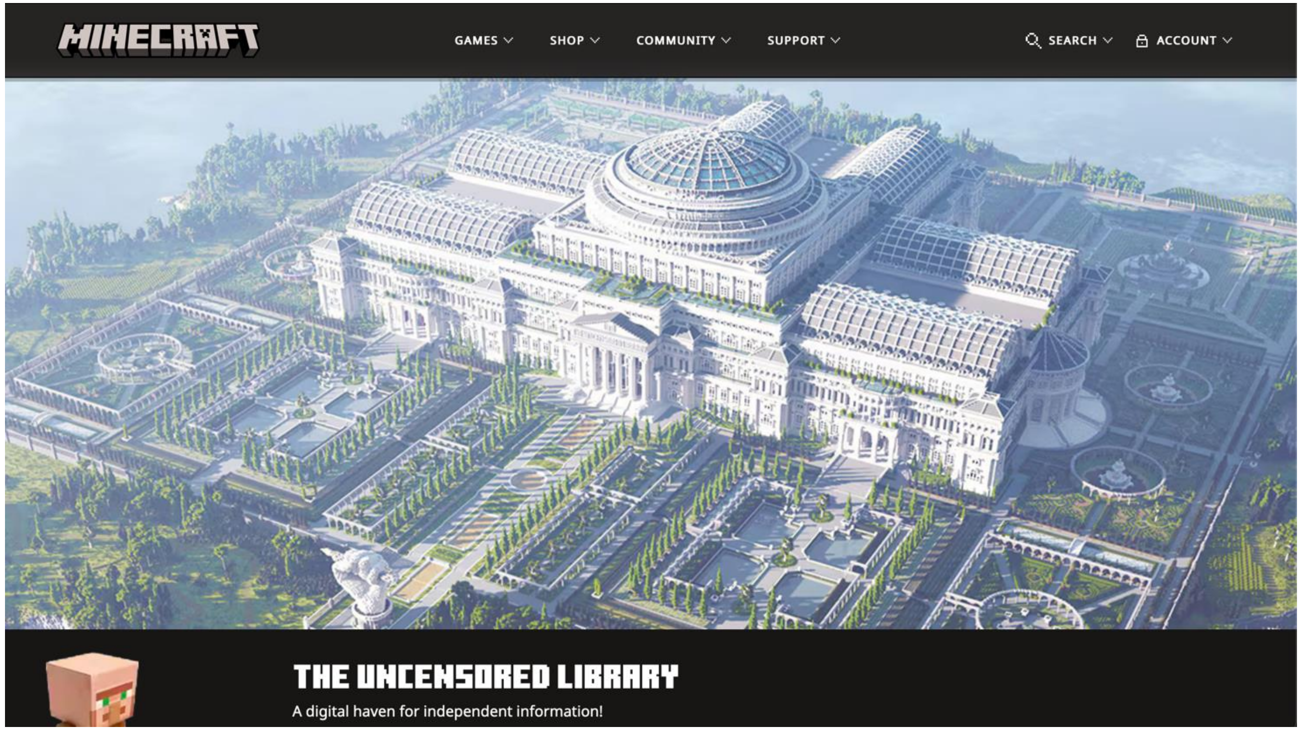
- DNS Tunneling is based on encoding sensitive data(e.g., secret data) into a series of DNS queries.

Exfiltrate sensitive data without raising alarms.



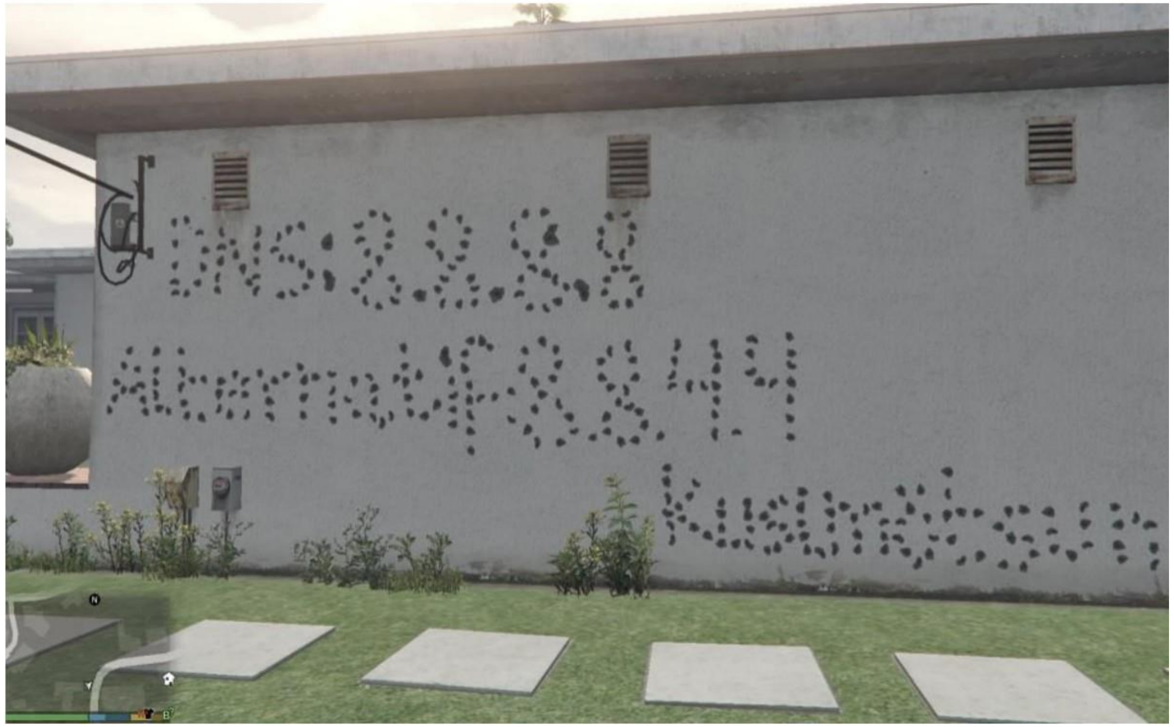
Example: Games

- We can create covert channels by encoding information in games’ virtual worlds which are shared by multiple users



THE UNCENSORED LIBRARY

A digital haven for independent information!



How to detect/prevent network covert channels

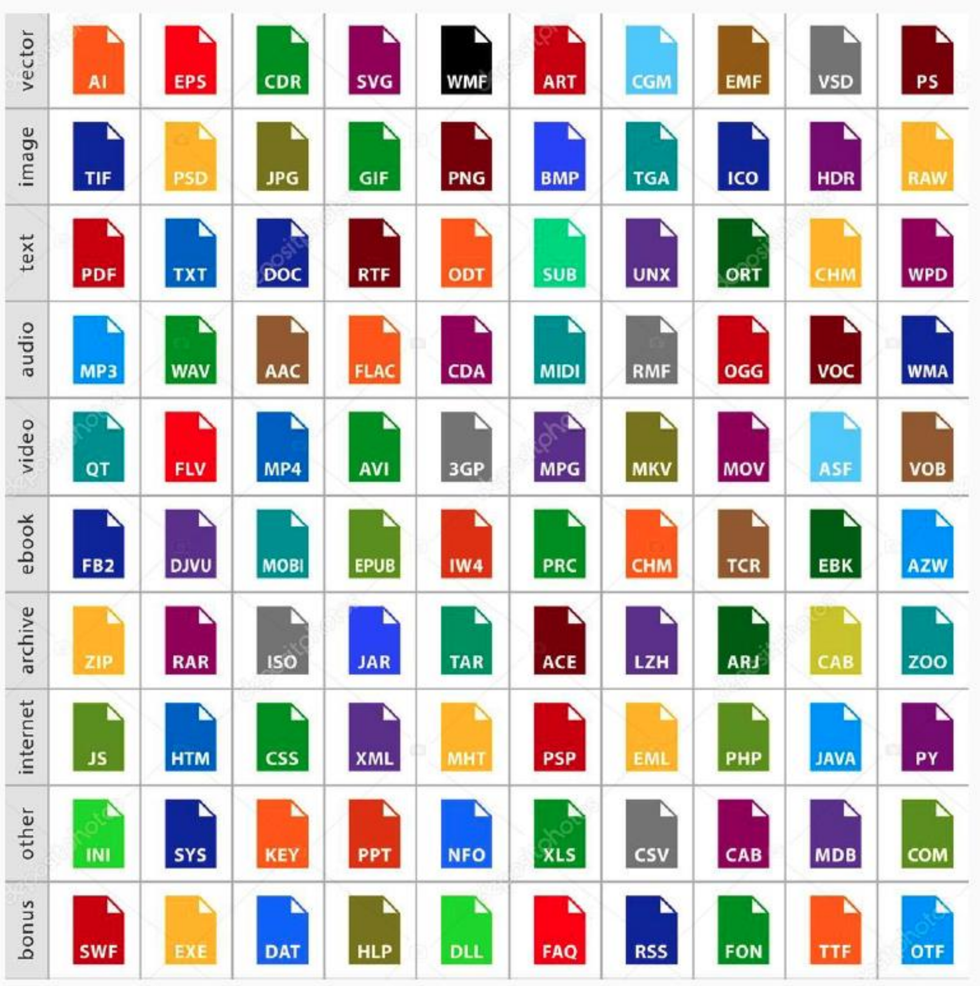
- A warden inspects (and/or manipulates) traffic to detect (and/or break) covert channels
- Storage channels
  - Passive: Analyze transmitted data for anomalies.
  - Active: Normalize data in header fields
- Timing channels
  - Passive: Analyze packet timing for inconsistencies
  - Active: Shape traffic (e.g., constant rate)

File Formats (and a little help for A2)

A Primer on File Formats

- A file format is a standard way that information is encoded for storage in a computer file

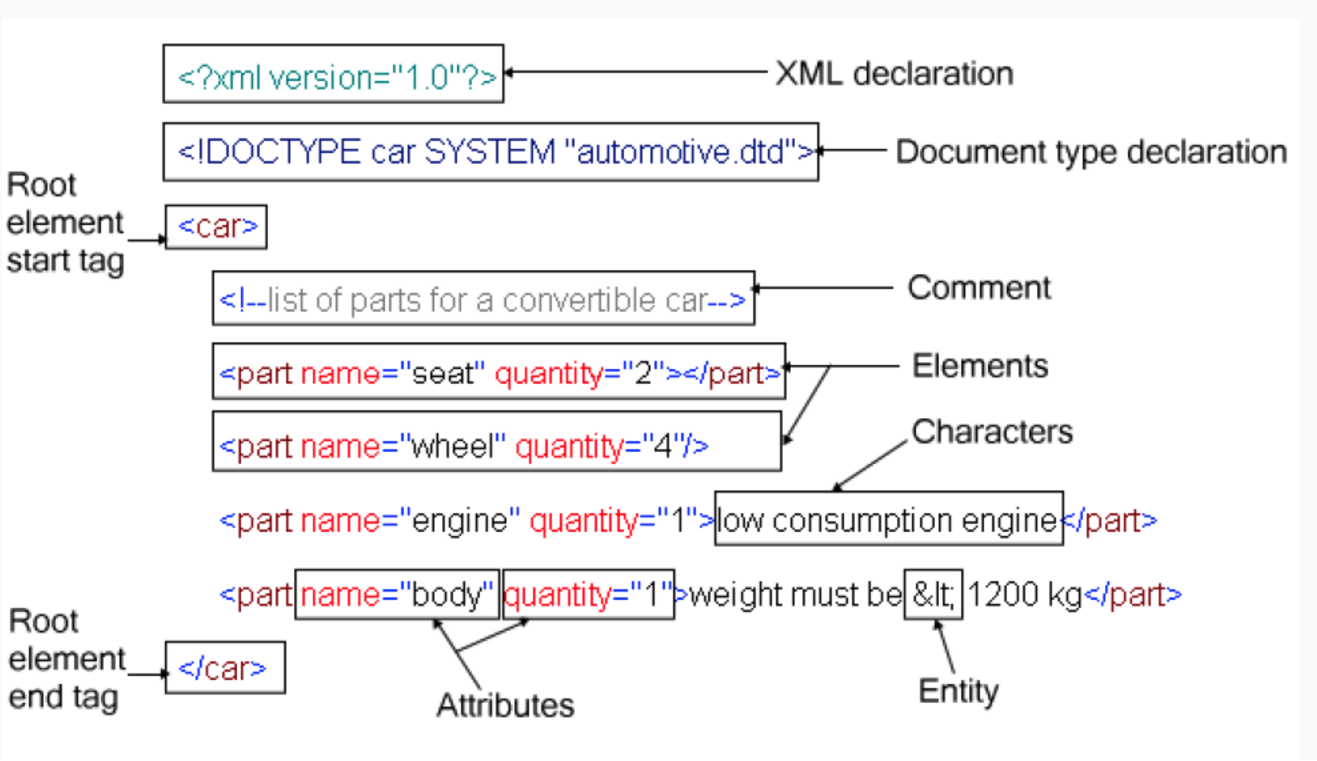
- There are two broad file format families:
  - Text files: Essential to determine the text encoding scheme and structure (if any)
  - Binary files: Essential to determine the file format



vector	AI	EPS	CDR	SVG	WMF	ART	CGM	EMF	VSD	PS
image	TIF	PSD	JPG	GIF	PNG	BMP	TGA	ICO	HDR	RAW
text	PDF	TXT	DOC	RTF	ODT	SUB	UNX	ORT	CHM	WPD
audio	MP3	WAV	AAC	FLAC	CDA	MIDI	RMF	OGG	VOC	WMA
video	QT	FLV	MP4	AVI	3GP	MPG	MKV	MOV	ASF	VOB
ebook	FB2	DJVU	MOBI	EPUB	IW4	PRC	CHM	TCR	EBK	AZW
archive	ZIP	RAR	ISO	JAR	TAR	ACE	LZH	ARJ	CAB	ZOO
internet	JS	HTM	CSS	XML	MHT	PSP	EML	PHP	JAVA	PY
other	INI	SYS	KEY	PPT	NFO	XLS	CSV	CAB	MDB	COM
bonus	SWF	EXE	DAT	HLP	DLL	FAQ	RSS	FON	TTF	OTF

Text Files

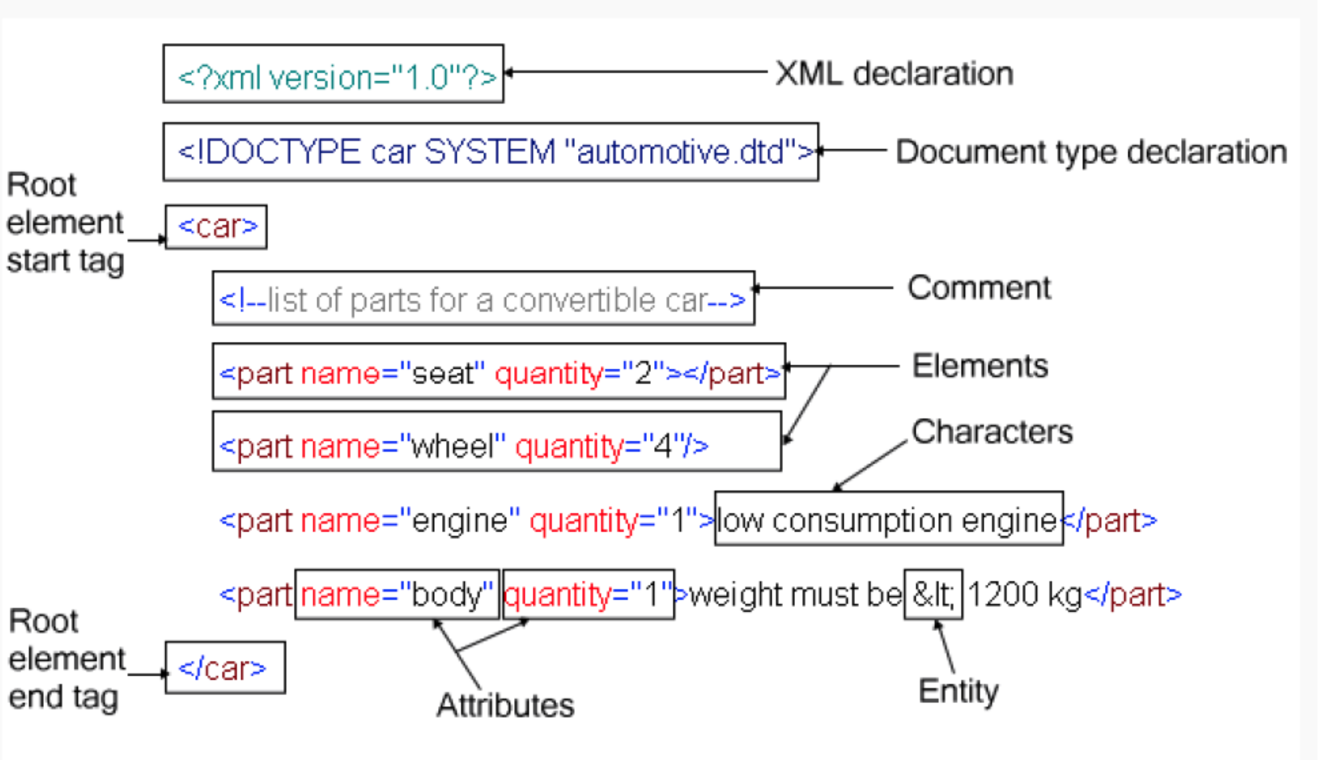
- Text files can have some structure on their own
  - E.g., XML, HTML, JSON, etc.



```
<?xml version="1.0"?>
<!DOCTYPE car SYSTEM "automotive.dtd">
<car>
  <!--list of parts for a convertible car-->
  <part name="seat" quantity="2"></part>
  <part name="wheel" quantity="4"/>
  <part name="engine" quantity="1">low consumption engine</part>
  <part name="body" quantity="1">weight must be &lt; 1200 kg</part>
</car>
```

Text Files

- Text files can have some structure on their own
  - E.g., XML, HTML, JSON, etc.

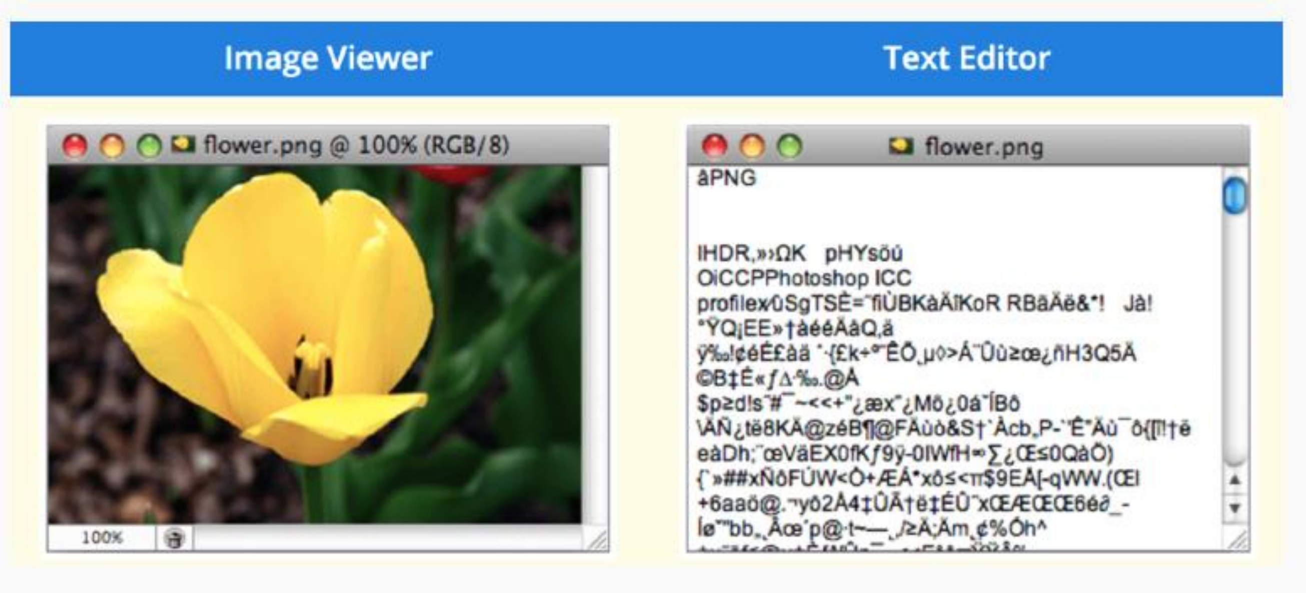


```
<?xml version="1.0"?>
<!DOCTYPE car SYSTEM "automotive.dtd">
<car>
  <!--list of parts for a convertible car-->
  <part name="seat" quantity="2"></part>
  <part name="wheel" quantity="4"/>
  <part name="engine" quantity="1">low consumption engine</part>
  <part name="body" quantity="1">weight must be &lt; 1200 kg</part>
</car>
```

Some of these elements may be used to store covert data as part of a covert storage channel…

Binary Files

- In binary files, bytes represent custom data
- Binary file formats may include multiple types of data in the same file, such as image, video, and audio data
  - This data can be interpreted by supporting programs, but will show up as garbled text in a text editor

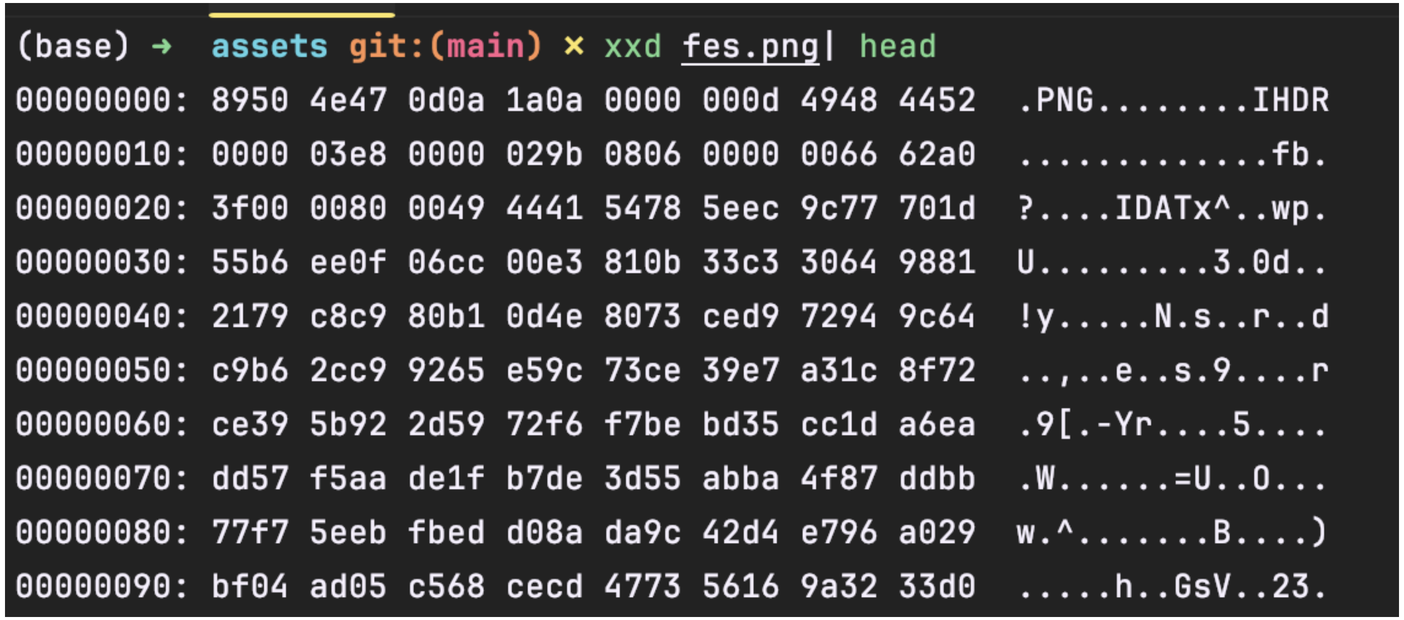


```
âPNG


IHDR,»›ΩK   pHYsöû
OiCCPPhotoshop ICC
profilexúSgTSÈ=˜fiÙBKàÄîKoR RBãÄë&*!   Jà!
*ŸQ¡EE»†àéé ÄâQ,ä
ÿ‰!¢éÉ£àä ˙·{£k+°˜ÊÕ¸µ◊>Á˜Ûù≥œ¿ñH3Q5Ä
©B‡Ê«ƒΔ·‰.@Å
$p≥d!s˜#˘~<<+"¿æx˝¿Mõ¿0á˜ÎBô
\ÄÑ¿të8KÄ@zéB¶@FÄùò&S†˙Àcb¸P-˙Ê˝Äû¯ô{[!†ë
eàDh;˝œVäEX0fKƒ9ÿ-0IWfH∞∑¿Œ≤0QàÕ)
{˙»##xÑôFÚW<Ô+ÆÁ*xô≤<π$9EÄ[-qWW.(ŒI
+6aaö@.¬yô2Å4‡ÛÃ†ë‡ÉÛ˜xŒÆŒŒ6é∂_-
Iø˜"bb.¸Åœ´p@·t~—¸/≥Ä;Äm¸¢%Ôh^
```

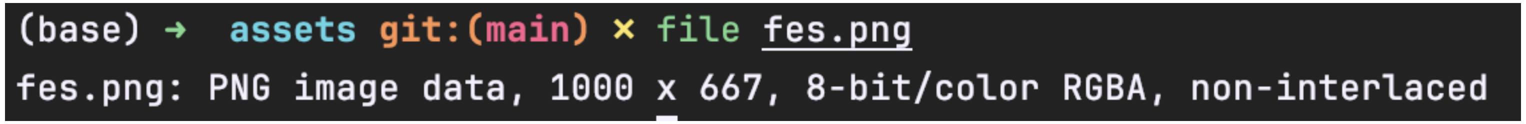
Inspection of a file’s raw bytes

- Use a hex editor to read file contents, e.g., xxd



```
(base) ➜  assets git:(main) ✗ xxd fes.png| head
00000000: 8950 4e47 0d0a 1a0a 0000 000d 4948 4452  .PNG........IHDR
00000010: 0000 03e8 0000 029b 0806 0000 0066 62a0  .............fb.
00000020: 3f00 0080 0049 4441 5478 5eec 9c77 701d  ?....IDATx^..wp.
00000030: 55b6 ee0f 06cc 00e3 810b 33c3 3064 9881  U.........3.0d..
00000040: 2179 c8c9 80b1 0d4e 8073 ced9 7294 9c64  !y.....N.s..r..d
00000050: c9b6 2cc9 9265 e59c 73ce 39e7 a31c 8f72  ..,..e..s.9....r
00000060: ce39 5b92 2d59 72f6 f7be bd35 cc1d a6ea  .9[.-Yr....5....
00000070: dd57 f5aa de1f b7de 3d55 abba 4f87 ddbb  .W......=U..O...
00000080: 77f7 5eeb fbed d08a da9c 42d4 e796 a029  w.^.......B....)
00000090: bf04 ad05 c568 cecd 4773 5616 9a32 33d0  .....h..GsV..23.
```

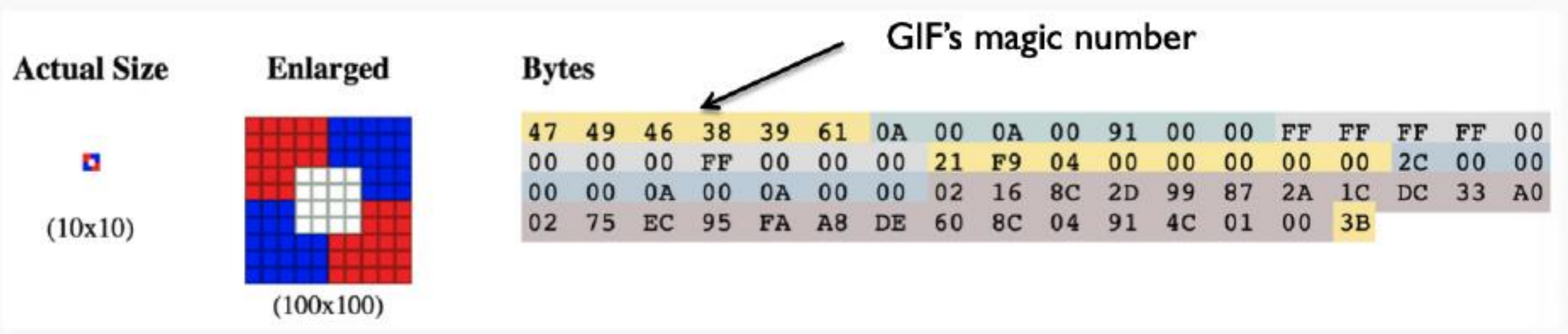
- Use the file utility to match a file’s signature



```
(base) ➜  assets git:(main) ✗ file fes.png
fes.png: PNG image data, 1000 x 667, 8-bit/color RGBA, non-interlaced
```

Magic Numbers

- When in doubt, look for magic numbers
  - Numerical/text values used to identify a file or protocol
  - E.g., GIF files start with the sequence **0x 47 49 46 38 39 61**

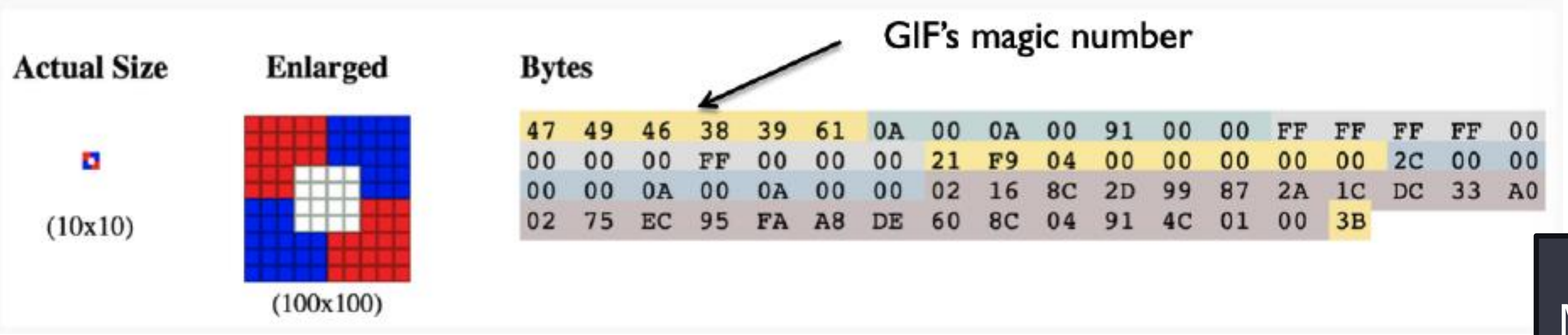


```
47 49 46 38 39 61 0A 00 0A 00 91 00 00 FF FF FF FF 00
00 00 00 FF 00 00 00 21 F9 04 00 00 00 00 00 2C 00 00
00 00 0A 00 0A 00 00 02 16 8C 2D 99 87 2A 1C DC 33 A0
02 75 EC 95 FA A8 DE 60 8C 04 91 4C 01 00 3B
```

- Magic numbers of common file formats:
  - ❖ http://www.garykessler.net/library/file_sigs.html

Magic Numbers

- When in doubt, look for magic numbers
  - Numerical/text values used to identify a file or protocol
  - E.g., GIF files start with the sequence **0x 47 49 46 38 39 61**



```
47 49 46 38 39 61 0A 00 0A 00 91 00 00 FF FF FF FF 00
00 00 00 FF 00 00 00 21 F9 04 00 00 00 00 00 2C 00 00
00 00 0A 00 0A 00 00 02 16 8C 2D 99 87 2A 1C DC 33 A0
02 75 EC 95 FA A8 DE 60 8C 04 91 4C 01 00 3B
```

- Magic numbers of common file formats:
  - ❖ http://www.garykessler.net/library/file_sigs.html

Maybe I can use this to make sense out of what's being transmitted within a covert storage channel…

Network Information Hiding

Traffic obfuscation

Information concealment in networks

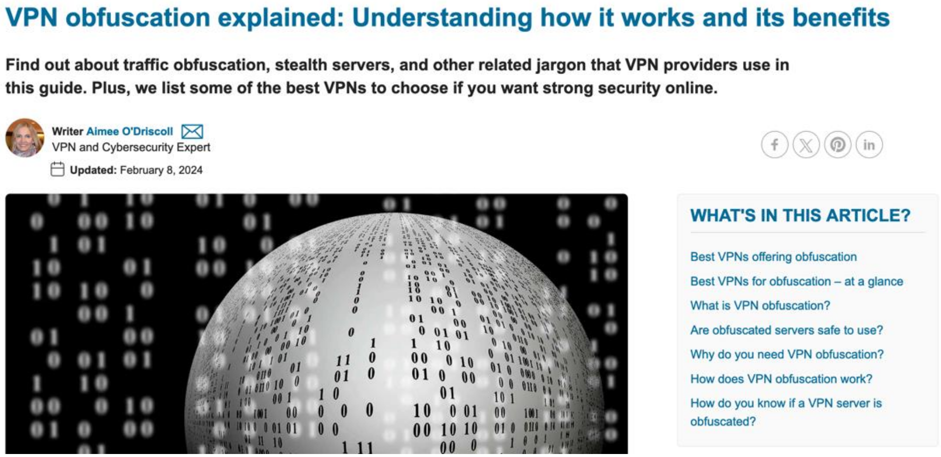
- Timing and content anomalies may be an effective way to detect covert channels

- Are there better ways to hide the existence of covert data transmissions?

Information concealment in networks

- Well, yes!

- Traffic obfuscation:
  - Hide the characteristics of a covert data transmission by shaping the “look” of data exchanges
  - e.g., used to hide malware communication with a C&C server, evade censorship, etc.



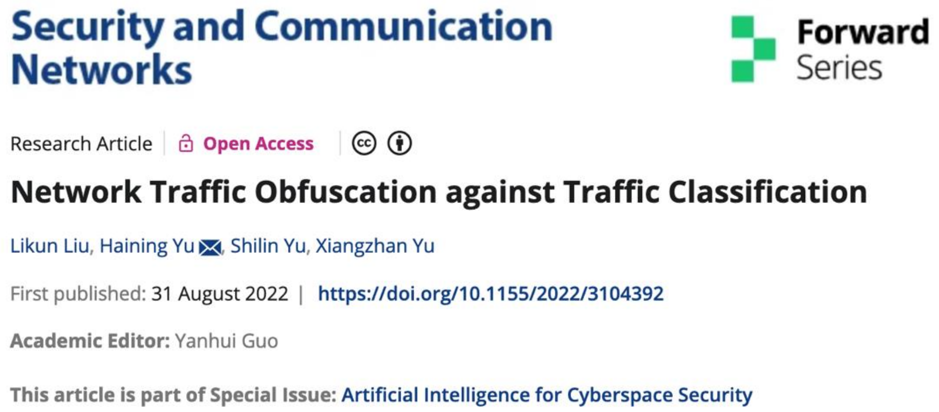
VPN obfuscation explained: Understanding how it works and its benefits

Find out about traffic obfuscation, stealth servers, and other related jargon that VPN providers use in this guide. Plus, we list some of the best VPNs to choose if you want strong security online.

Writer Aimee O'Driscoll
VPN and Cybersecurity Expert
Updated: February 8, 2024

WHAT'S IN THIS ARTICLE?

- Best VPNs offering obfuscation
- Best VPNs for obfuscation – at a glance
- What is VPN obfuscation?
- Are obfuscated servers safe to use?
- Why do you need VPN obfuscation?
- How does VPN obfuscation work?
- How do you know if a VPN server is obfuscated?



Security and Communication Networks

Forward Series

Research Article | Open Access

Network Traffic Obfuscation against Traffic Classification

Likun Liu, Haining Yu, Shilin Yu, Xiangzhan Yu

First published: 31 August 2022 | https://doi.org/10.1155/2022/3104392

Academic Editor: Yanhui Guo

This article is part of Special Issue: Artificial Intelligence for Cyberspace Security

Different techniques for traffic obfuscation

- Randomize traffic
    - Don’t look like any particular protocol
- Mimic traffic
    - Attempt to look like some other protocol
- Tunnel traffic
    - Piggyback on another protocol’s execution

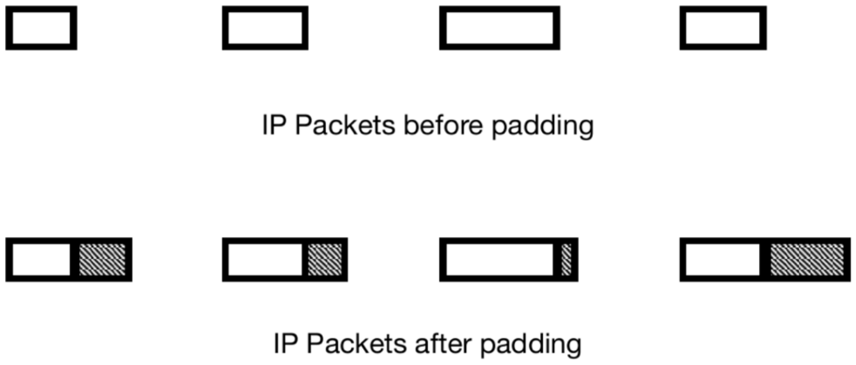


Figure 5.2: Padding network packets to de-identify packet sizes.

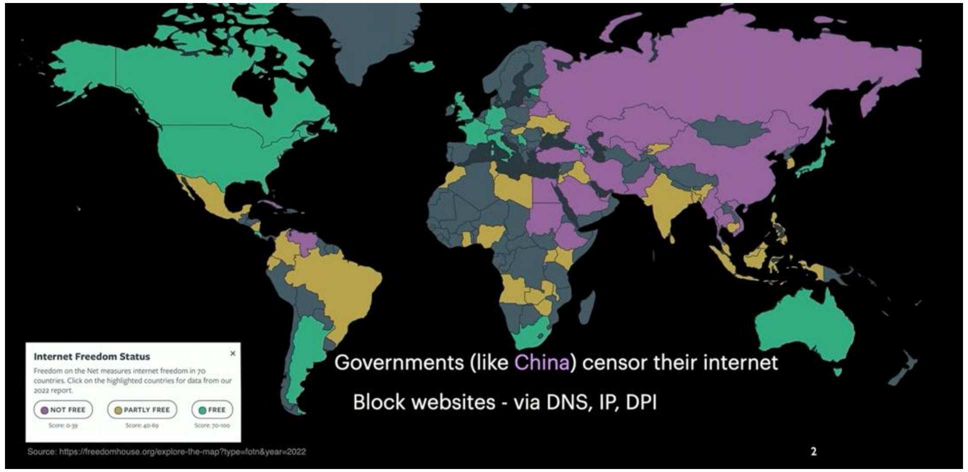
Traffic randomization

- Idea: evade inspection by generating traffic that does not conform to any known protocol specification
  - Randomize packet sizes and timings
  - Randomize packet contents (no signatures)

- Examples:
  - Shadowsocks
  - V2Ray
  - OutlineVPN

Issues with traffic randomization systems

- "Look-like-nothing" might be a signature in itself
- Does not work if wardens have protocol allow-lists in place
- Can be detected via cryptographic flaws and entropy tests
  - Security Notions for Fully Encrypted Protocols, Fenske and Johnson, FOCI 2023
  - How the Great Firewall of China Detects and Blocks Fully Encrypted Traffic, Wu et al., USENIX Security 2023



Freedom on the Net measures internet freedom in 70 countries. Click on the highlighted countries for data from our 2022 report.

Governments (like China) censor their internet

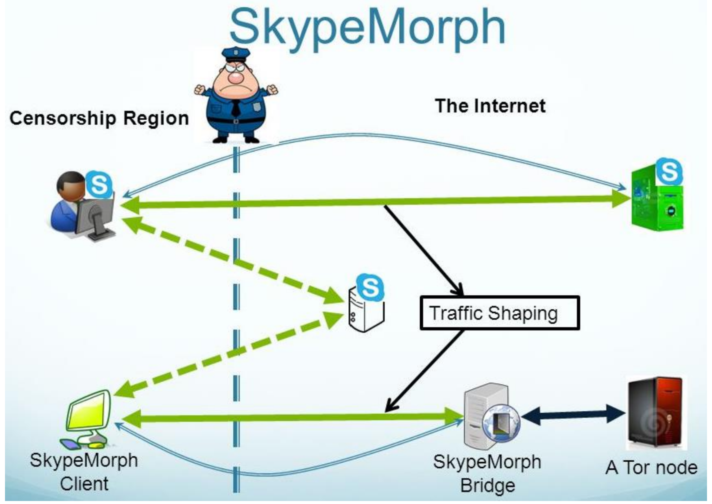
Block websites - via DNS, IP, DPI

Source: https://freedomhouse.org/explore-the-map?type=fotn&year=2022

Traffic mimicking

- Idea: Hide a protocol’s execution by mimicking another innocuous protocol’s characteristics (e.g., Skype)
  - Leverage steganography or encrypted carrier protocols
  - Embed covert data in specific protocol fields
  - Mimic how an encrypted cover protocol sends its traffic

- Examples:
  - SkypeMorph
  - StegoTorus
  - CensorSpoofer



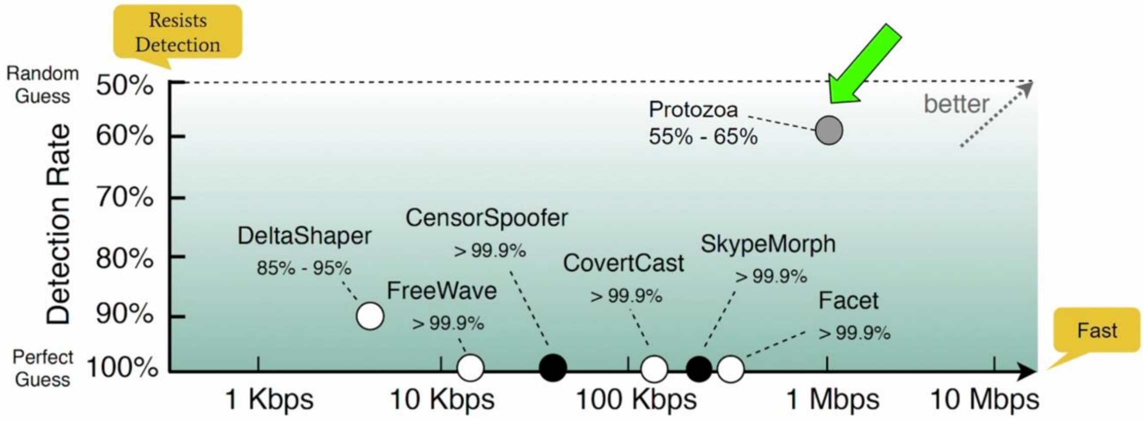
Issues with traffic mimicking systems

- It is very difficult to build a perfect imitation
  - Respond to network perturbations
  - Cover all corner cases and error conditions (and bugs!)
  - Mimic relationships between sub-protocols
  - Keep up with the cover protocol’s updates
- Now believed to be a fundamentally flawed approach
  - The Parrot is Dead: Observing Unobservable Network Communications, Houmansadr et al., S&P 2013

Traffic tunneling

- Idea: Piggyback covert data on the execution of a protocol
  - Send covert data as the protocol’s application messages
  - Avoids mimicking issues
  - Still needs to ensure the cover protocol does not generate “weird” traffic patterns

- Examples:
  - VoIP/video: FreeWave, DeltaShaper, Protozoa
  - HTTPS: meek, decoy routing, Balboa
  - IM/e-mail: Camoufler, SWEET
  - Cellphones: Dolphin



Issues with traffic tunneling systems

- Oftentimes, there is a disconnect between the usage patterns of the cover protocol and the covert protocol
  - Times of use, duration, etc.
  - The “greedy” tunneling of covert data may change the cover protocol’s typical traffic patterns
    - e.g., exchanging very large IMs very frequently on both directions
  - Covert data embedding mechanisms may slow down the cover’s protocol activity, leading to noticeable changes in traffic patterns
    - e.g., when replacing media data with covert content

Takeaways

- Covert channels allow for the clandestine transfer of information, both within processes of a given machine or across machines

- Network covert channels are increasingly hard to detect, but can also be used for commendable purposes (e.g., censorship evasion within repressive environments)