Execute_Command

- Execute system command on target.
- Ie:
  - if program is executed on Windows → execute windows commands.
  - If program is executed on Mac OS X → execute Unix commands.

After packaging:

- Execute any system command on any OS using a single file.

Execute_And_Report

- Execute system command on target and send result to email.
- Ie:
  - if program is executed on Windows → execute windows commands.
  - If program is executed on Mac OS X → execute Unix commands.

After packaging:

- Execute any system command on any OS using a single file.

Download_File

- Download file on system.
- Once packaged properly will work on all operating systems.
- Simple but powerful.

- Can be used in many situations:
  - download_file + execute_command = download_and_execute
  - download_file + execute_and_report = download_execute_and_report
  - .....etc

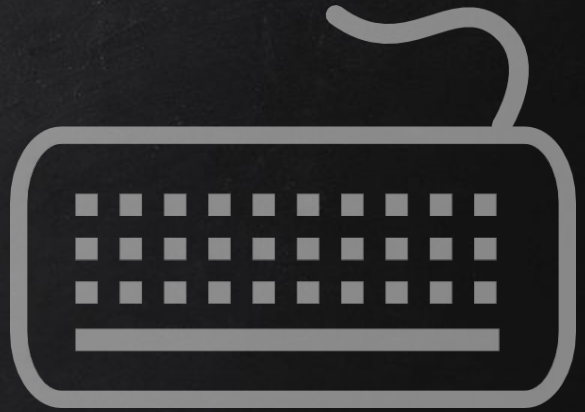
DOWNLOAD_EXECUTE_AND_REPORT

- Download file on system.
- Execute a command that uses this file.
- Report result to our email.
- Cross platform!!

- Ex: remotely steal all stored passwords on a computer!

Keylogger

Program that records keys pressed on the keyboard.



Common features:

- Store logs locally (local keyloggers).
- Report logs to email or remote server (remote keyloggers).
- Log screenshots.
- Start with system startup.

Keylogger

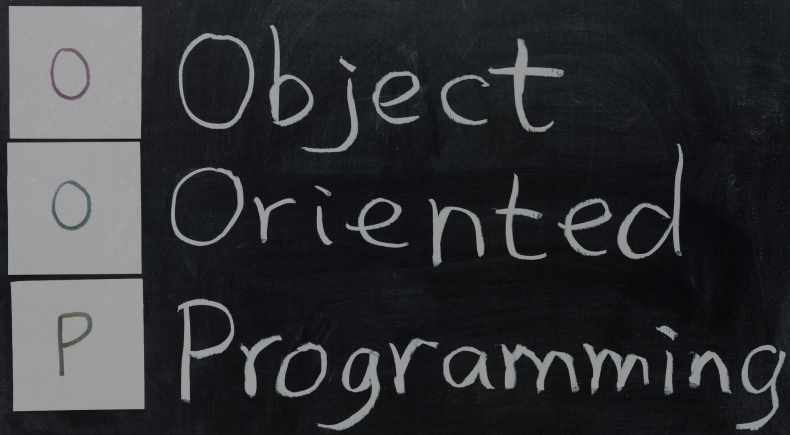
Report function:

- Run in the background.
- Don't interrupt program execution.
- Every X seconds, send report.

→ Great case for threading.

Keylogger

Classes



- Way of modeling program (blueprint).
- Logically group functions and data.
  - Makes code more readable.
  - More reusable.
  - Separate implementation from usage (encapsulation).
  - Easier to extend.
  - Easier to maintain.

Keylogger

Constructor Method

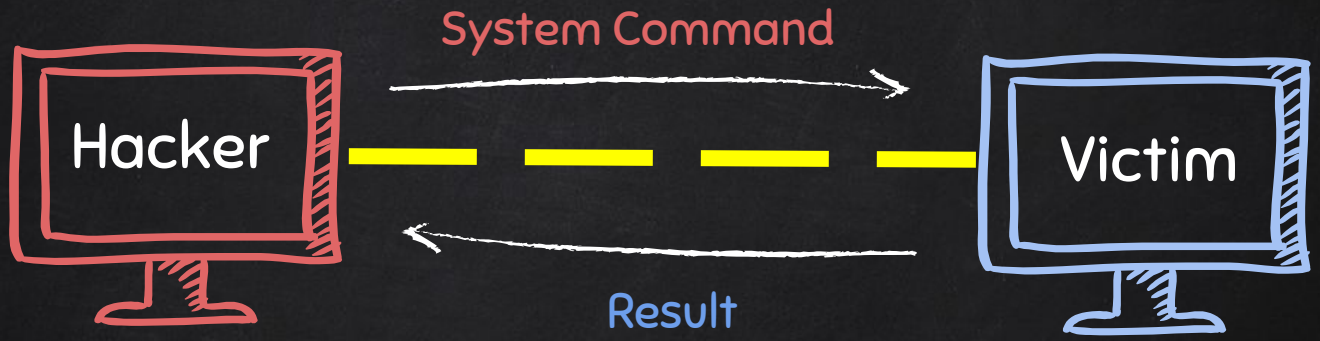
- AKA initialisation method.
- Gets executed automatically when a class is created.

Backdoors

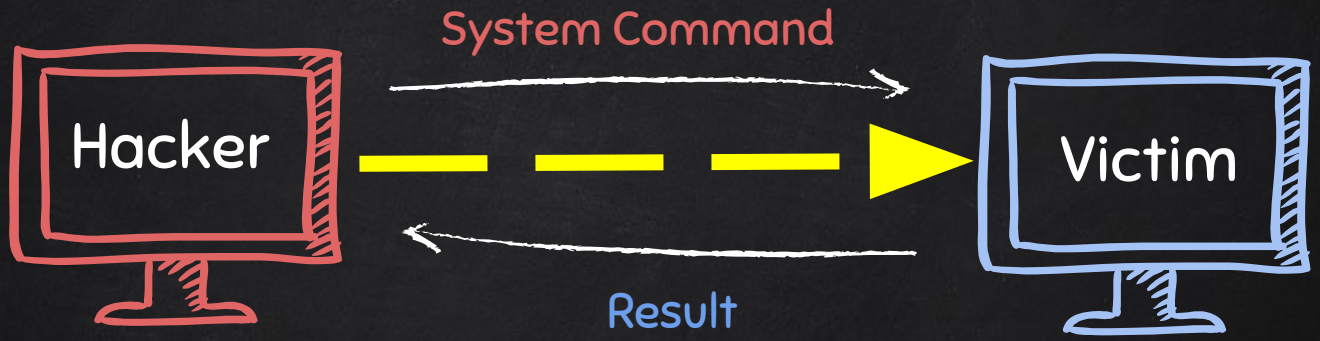
Interactive program gives access to system its executed on.

- Command execution.
- Access file system.
- Upload/download files.
- Run keylogger.
- .....etc

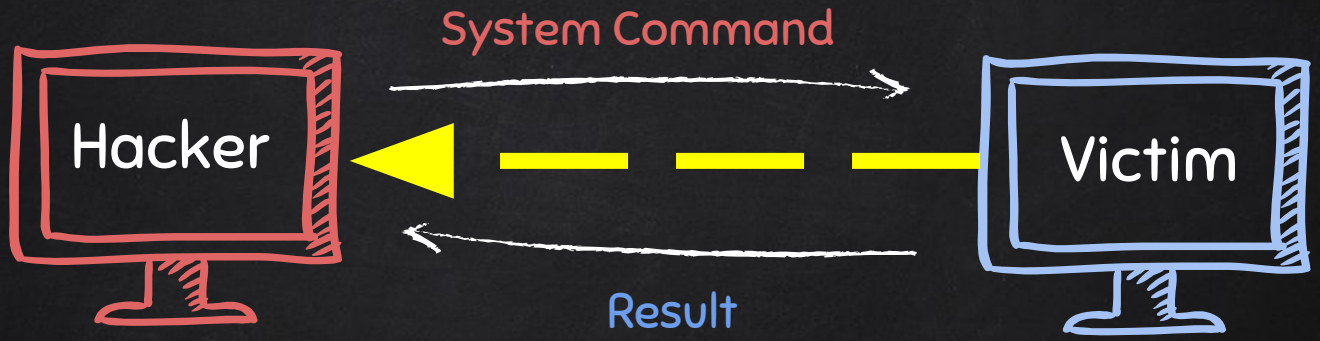
Backdoors



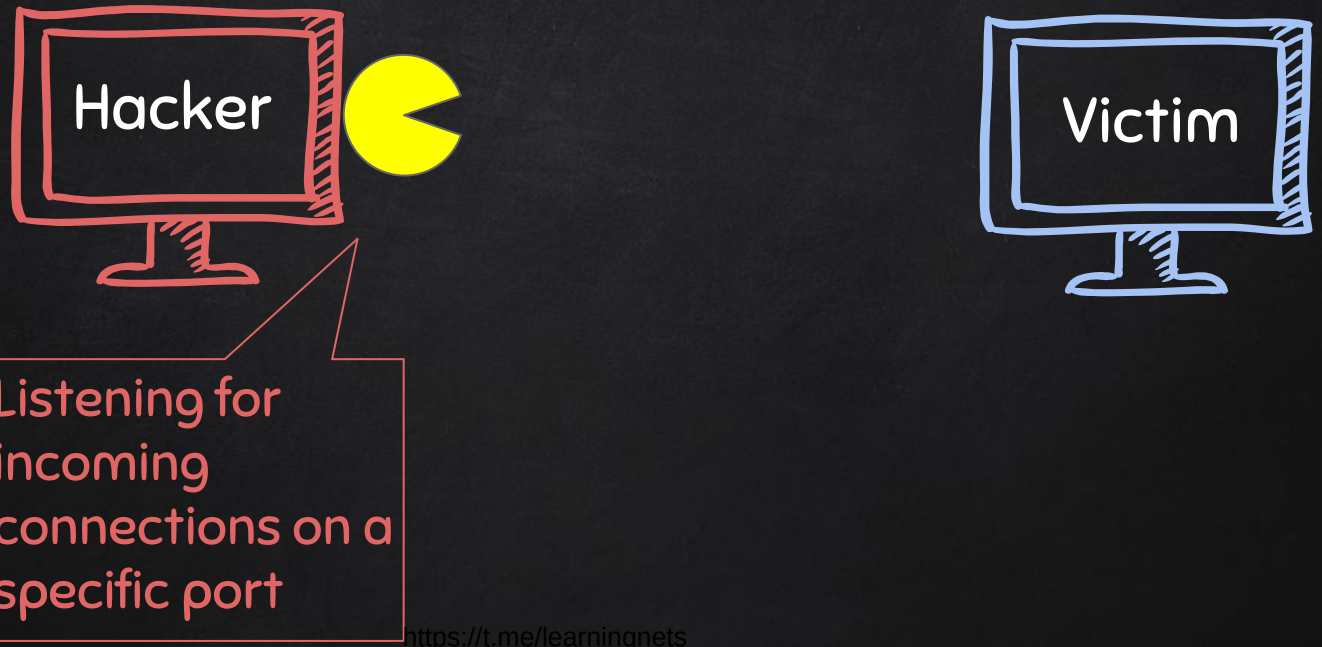
Backdoors – Bind / Direct Connection



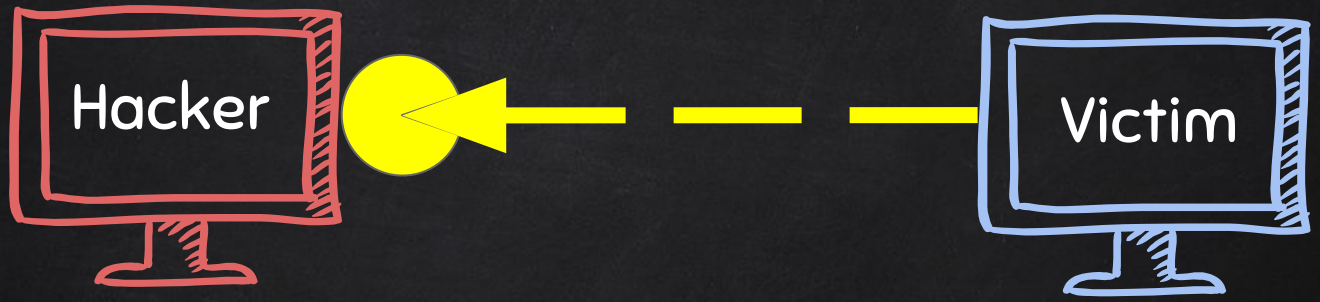
Backdoors – Reverse Connection



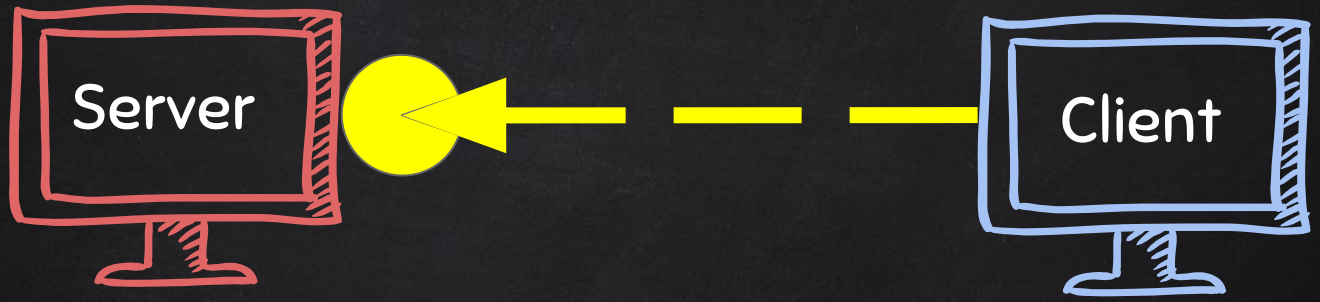
Backdoors – Reverse Connection



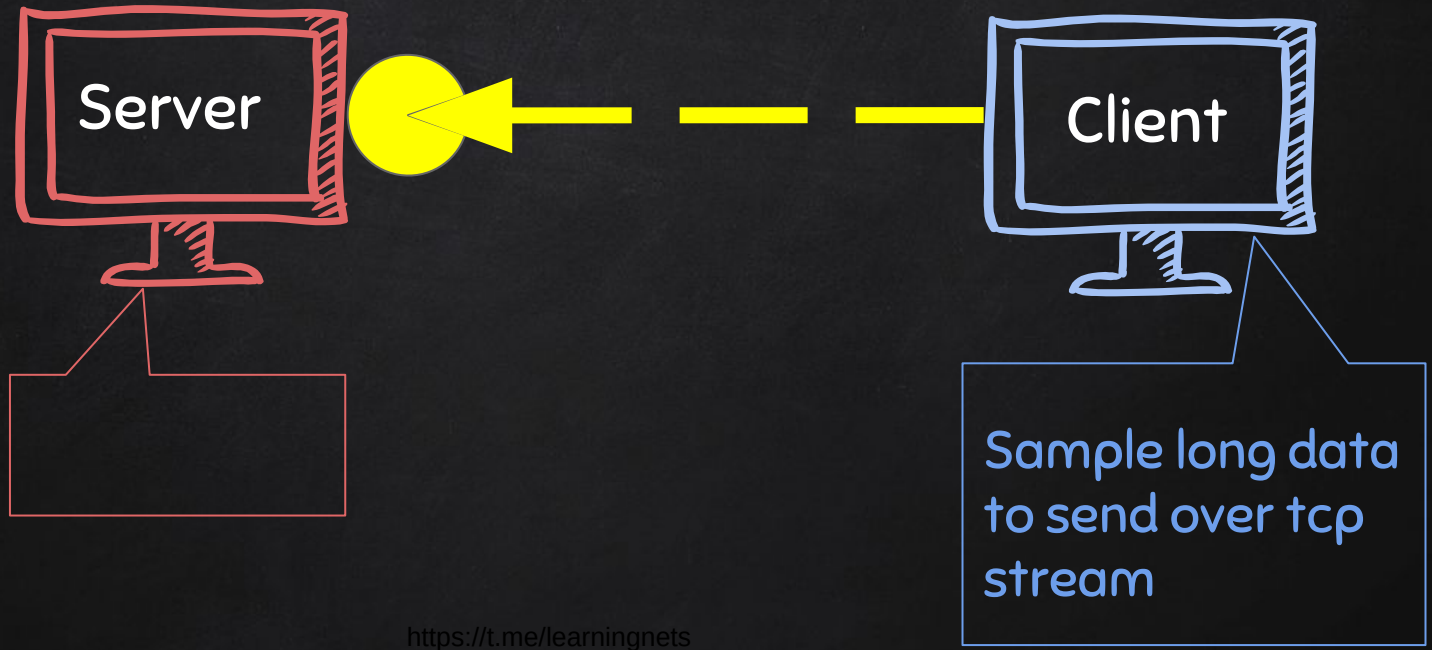
Backdoors – Reverse Connection



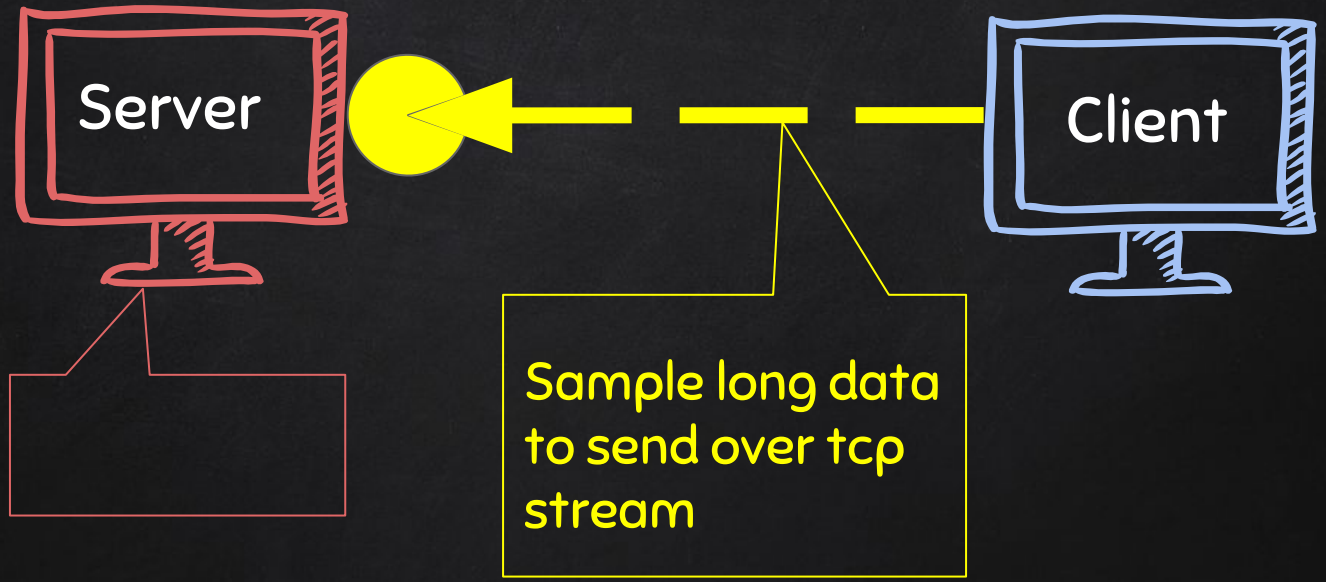
Transferring Data Over TCP



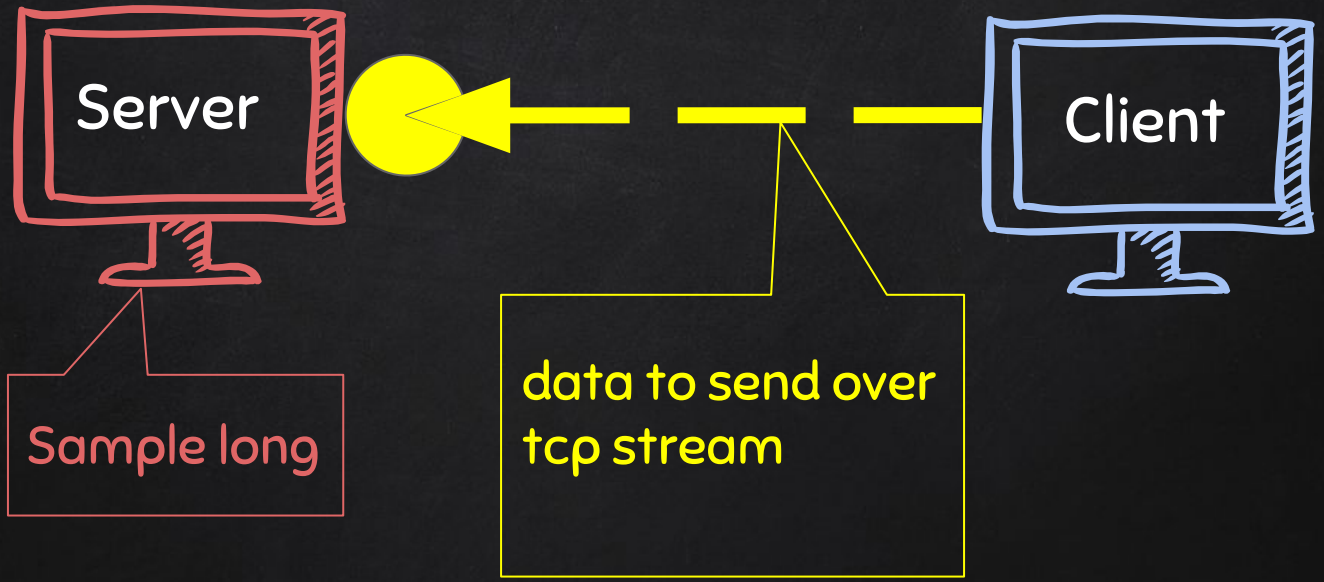
Transferring Data Over TCP



Transferring Data Over TCP



Transferring Data Over TCP



Backdoors

Sockets

Problem

- TCP is stream based.
- Difficult to identify the end of message/batch.

Solution:

- Make sure the message is well defined.
- Implement a protocol that send and receive methods conform to.
  - Send size of message as header.
  - Append a end-of-message mark to the end of each message.
  - Serialize the message .

Backdoors

Serialization

Benefits:

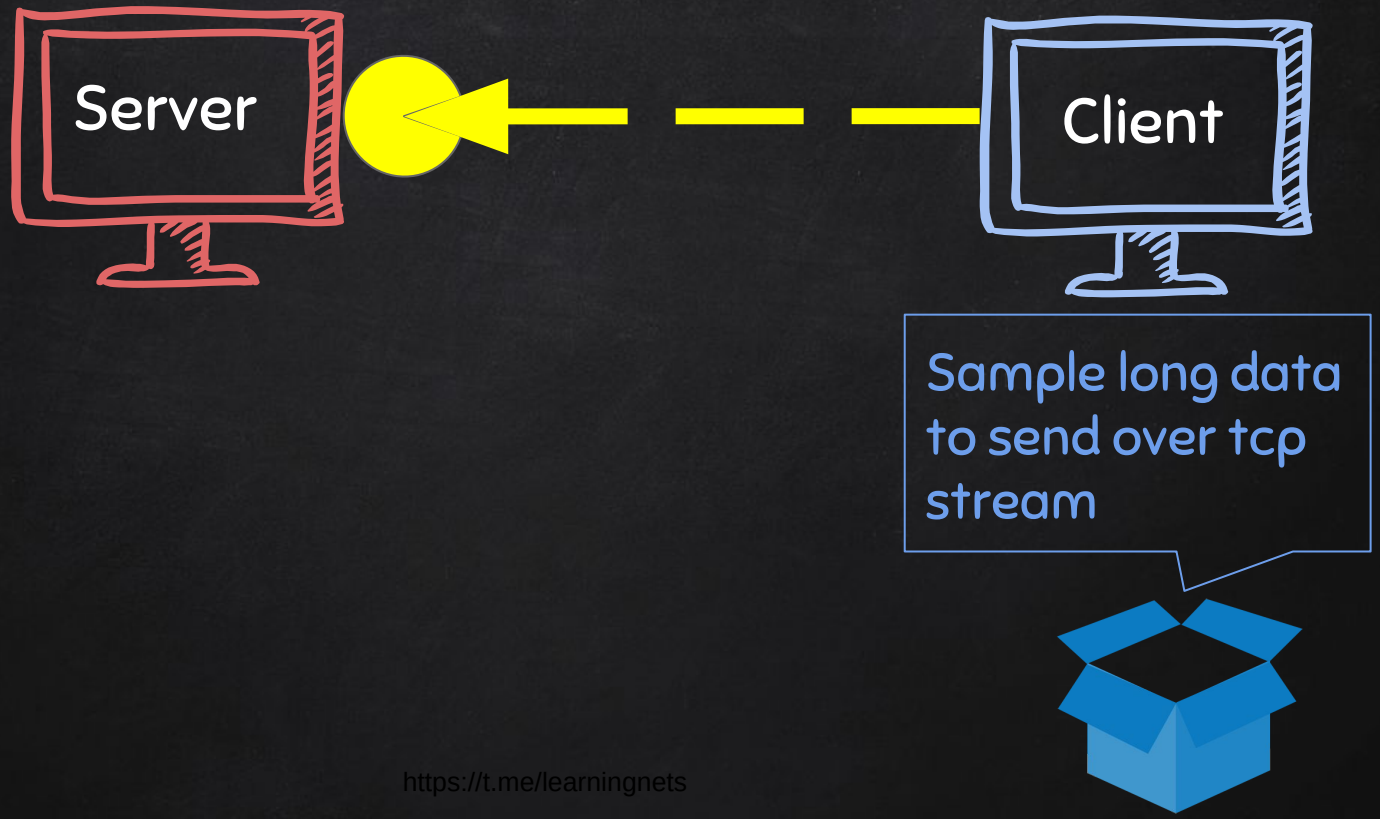
1. Message is well defined, receiver knows if message is incomplete.
2. Can be used to transfer objects (lists, dicts ...etc).

Backdoors – Serialization

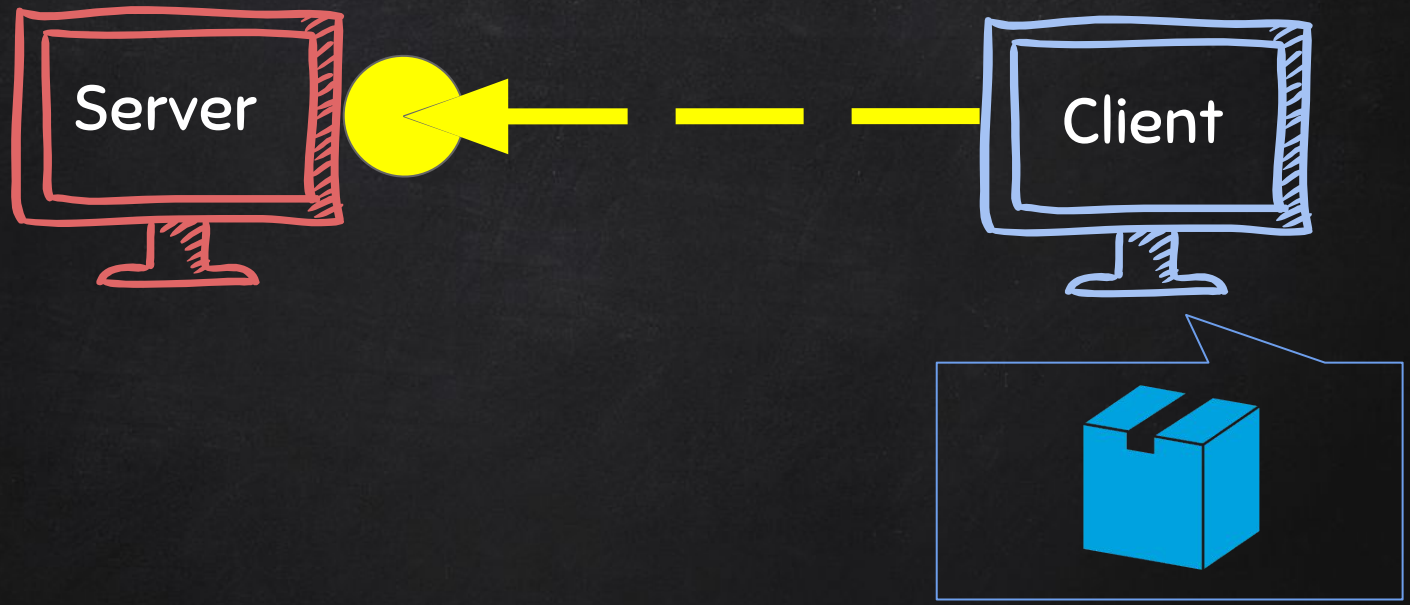
converts well-defined stream of bytes back into an object

converts object to a stream of well-defined bytes

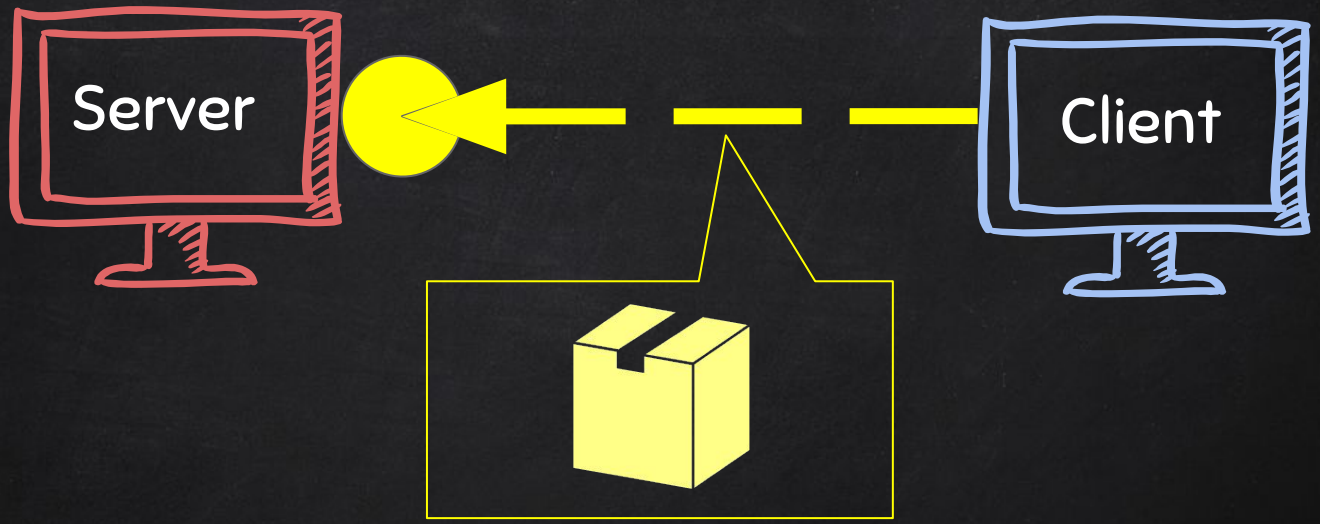
Serialization



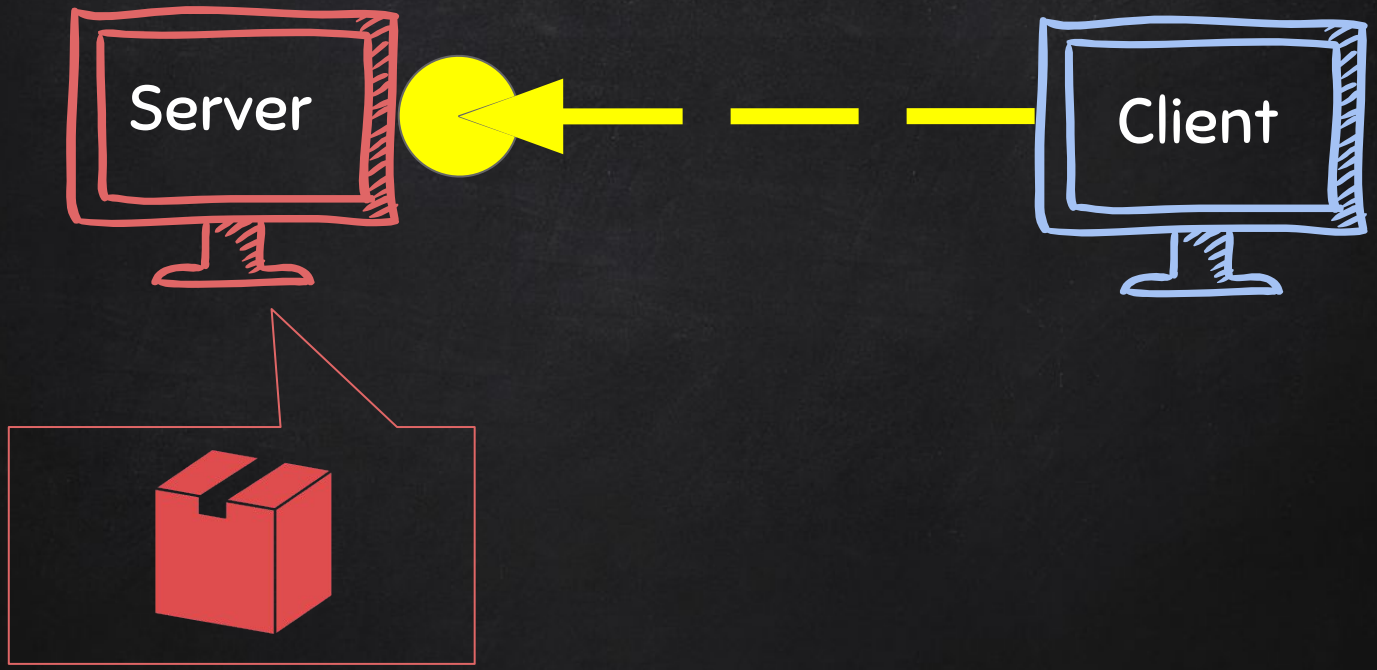
Serialization



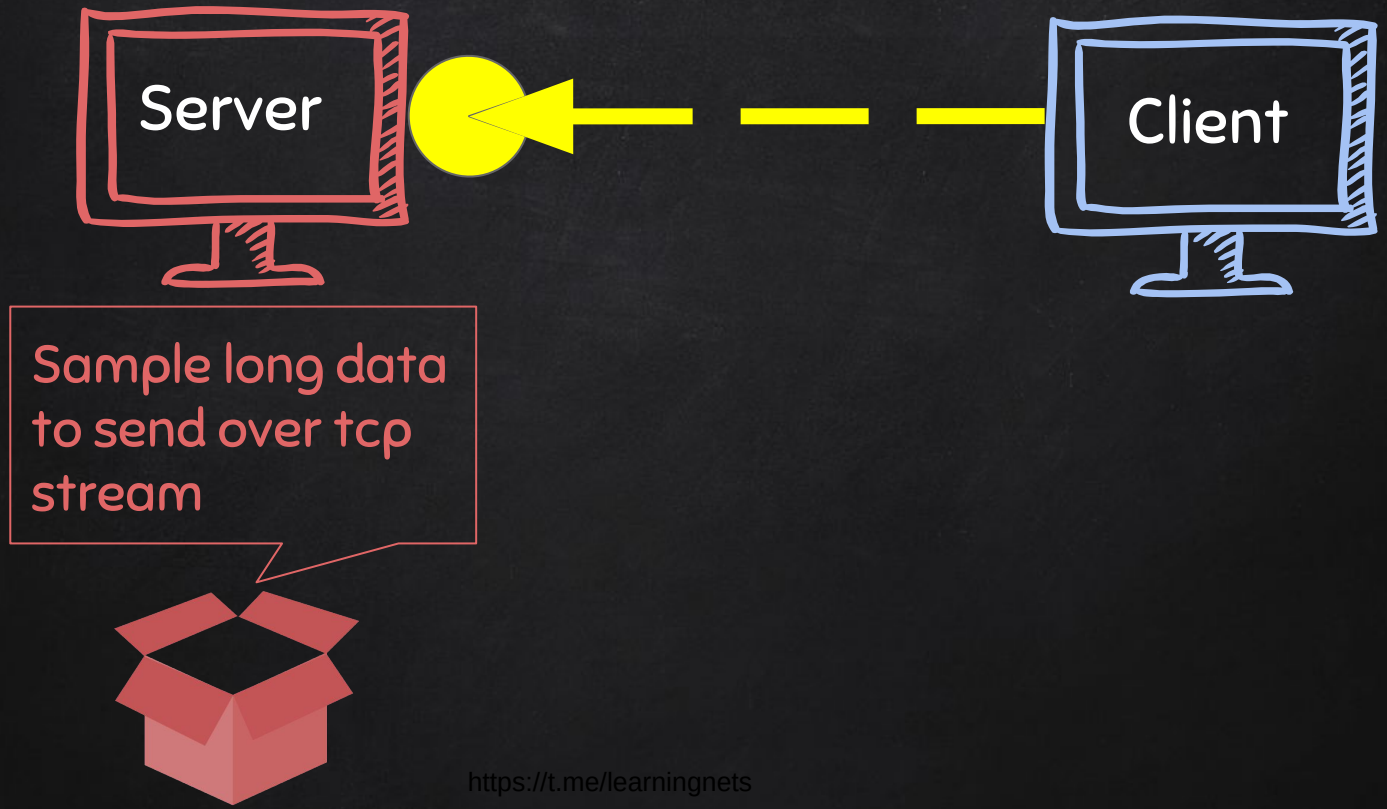
Serialization



Serialization



Serialization



Backdoors

Serialization

Implementation:

- Json and Pickle are common solutions.
- Json (Javascript Object Notation) is implemented in many programming languages.
- Represents objects as text.
- Widely used when transferring data between clients and servers.

Reverse_Backdoor

Features:

- Command execution.
  - dir
- Access file system.
  - cd DirectoryName
- Upload files.
  - upload filename.txt
- Download files.
  - Download filename.txt

Reverse_Backdoor

Access file system:

- Cd command changes current working directory.
- It has 2 behaviours:
  - cd → shows current working directory.
  - cd DirectoryName → changes current working directory to DirectoryName.

Reverse_Backdoor

File Download:

- A file is a series of character.

- Therefore to transfer a file we need to:
  1. Read the file as a sequence of characters.
  2. Send this sequence of characters.
  3. Create a new empty file at destination.
  4. Store the transferred sequence of characters in the new file.

Reverse_Backdoor

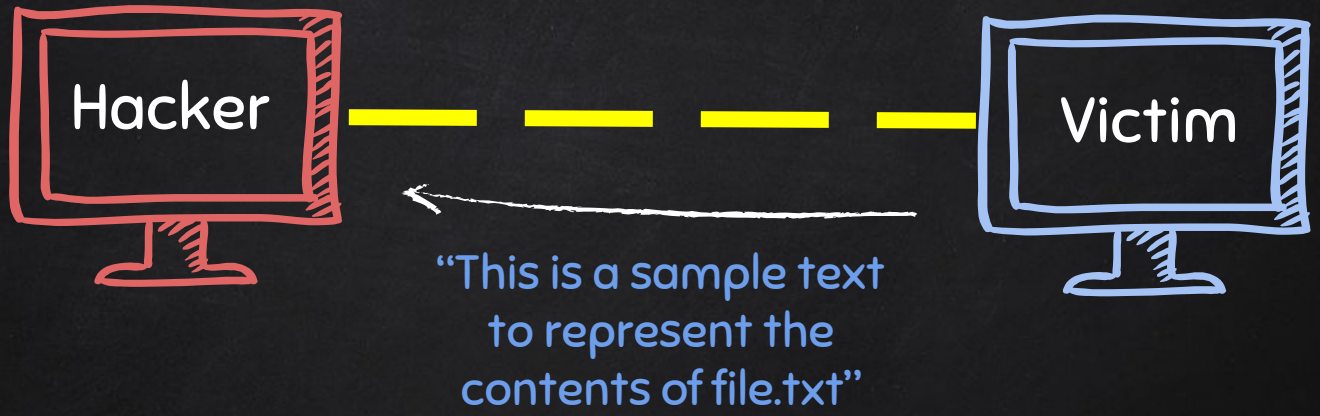
File Upload:

- A file is a series of character.
- Uploading a file is the opposite of downloading a file.

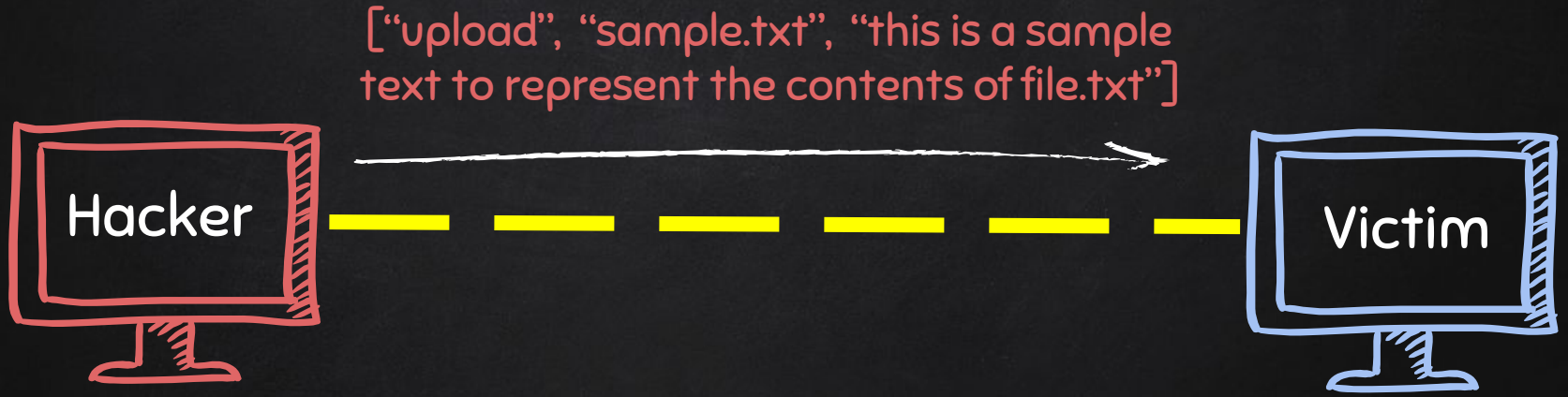
- Therefore to transfer a file we need to:
  1. Read the file as a sequence of characters.
  2. Send this sequence of characters.
  3. Create a new empty file at destination.
  4. Store the transferred sequence of characters in the new file.

Downloading Files

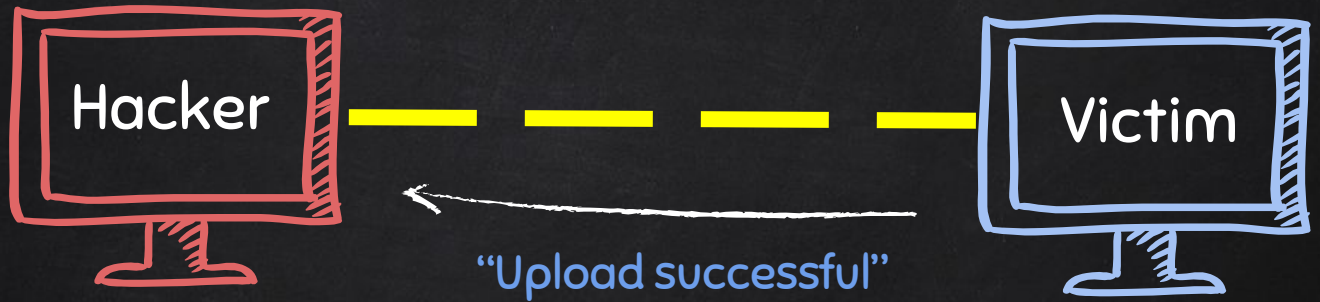
Downloading Files



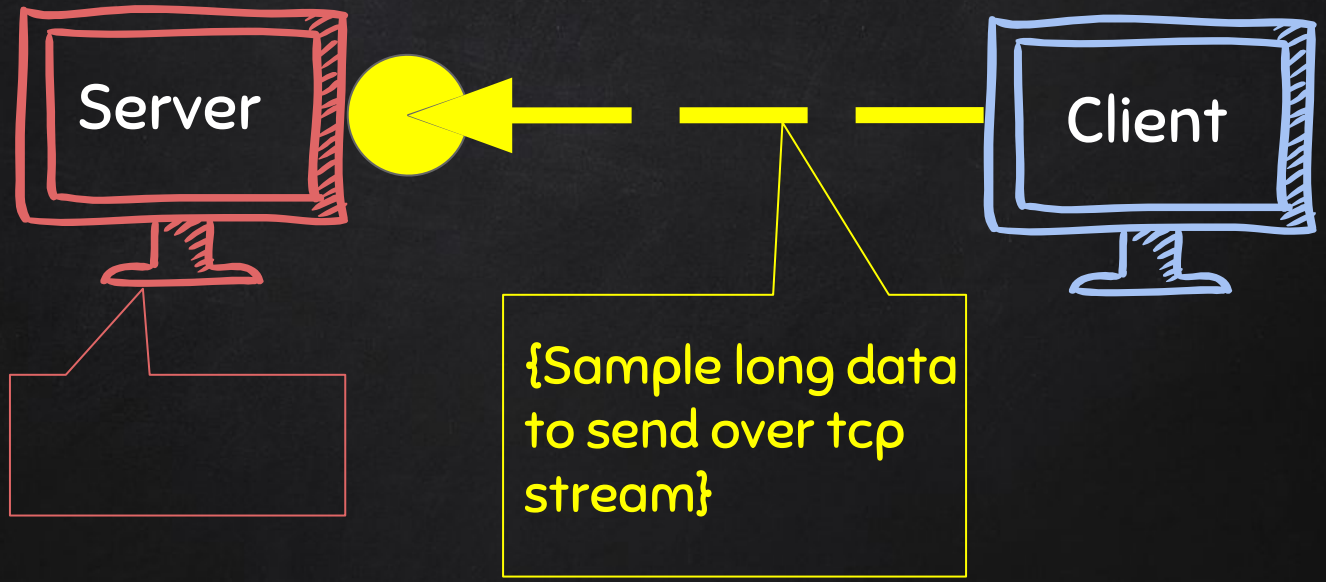
Uploading Files



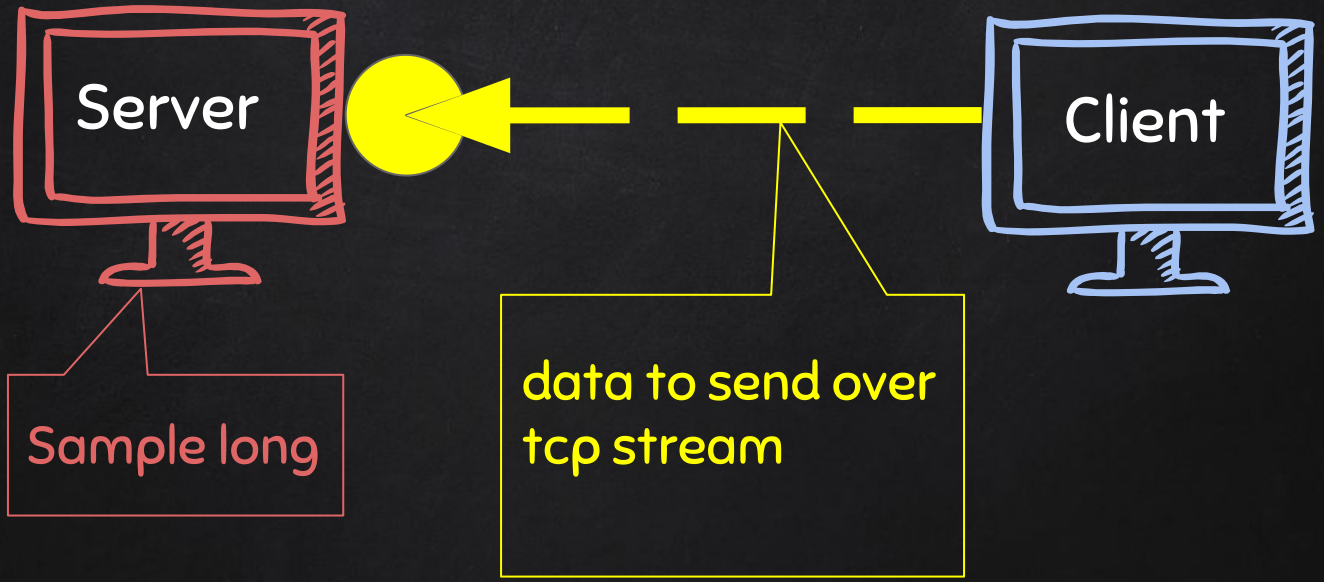
Uploading Files



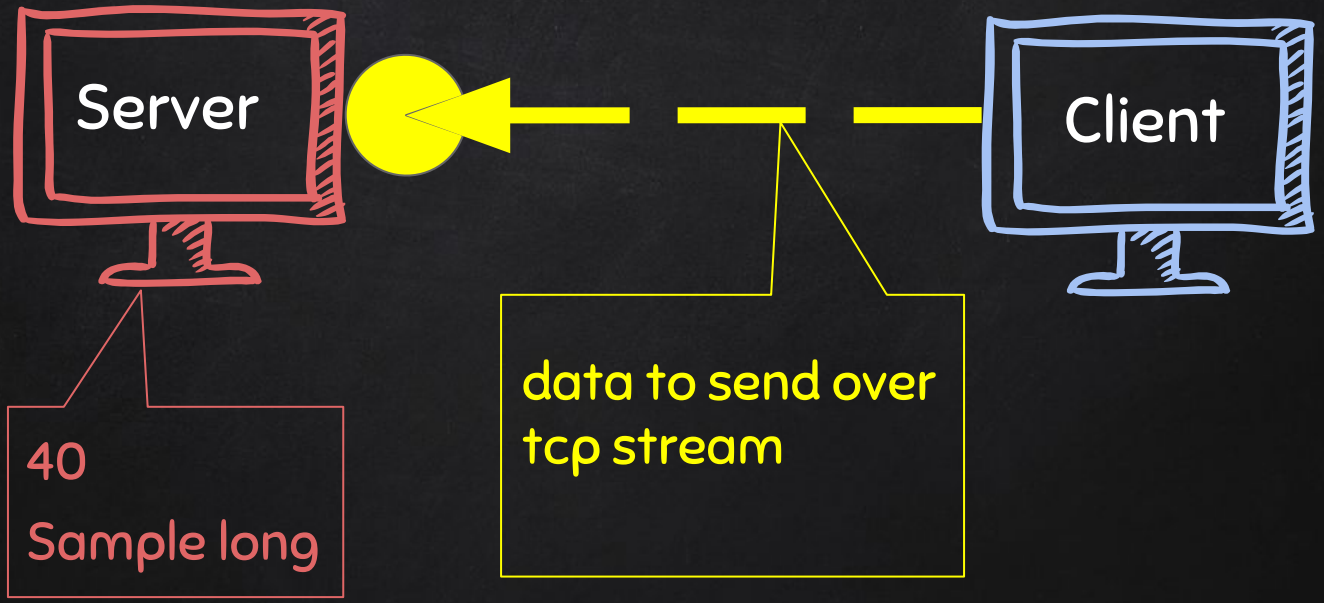
Backdoors – Reverse Connection



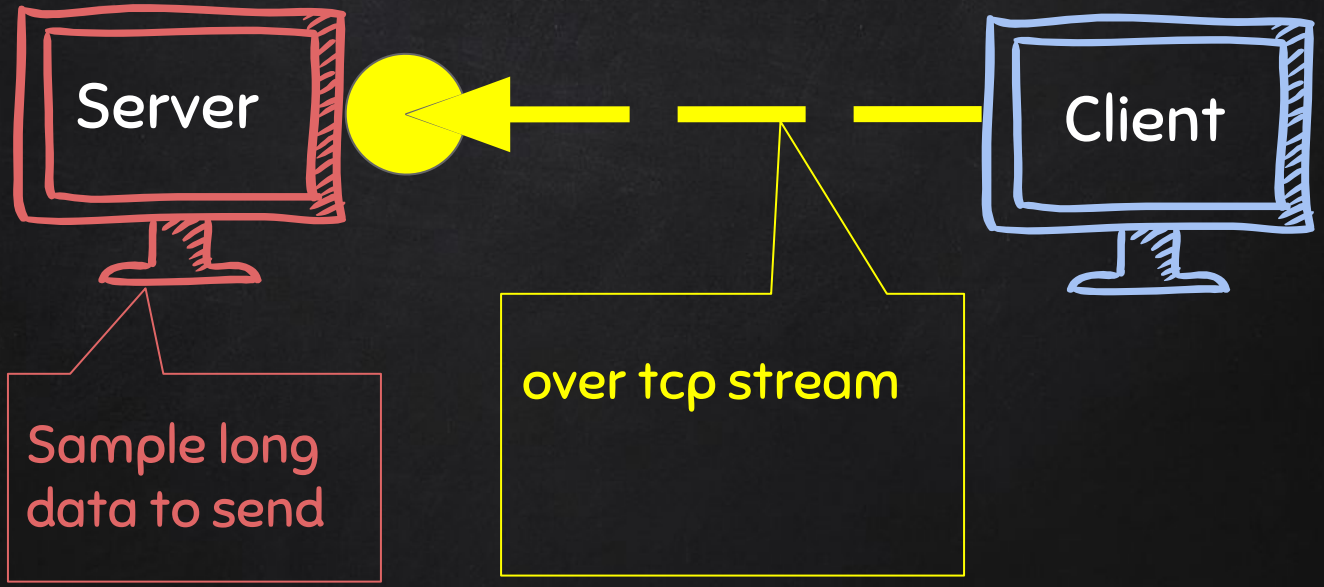
Backdoors – Reverse Connection



Backdoors – Reverse Connection



Backdoors – Reverse Connection

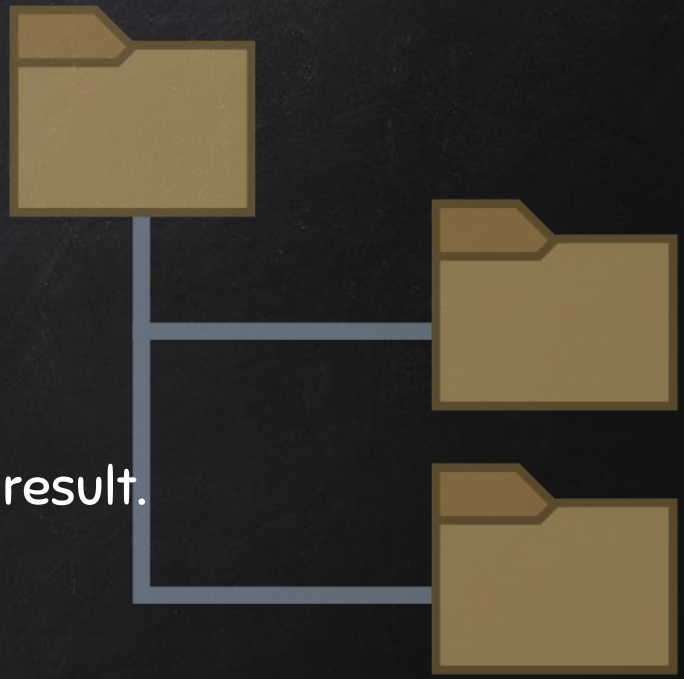


Backdoors – Reverse Connection

Sample long
data to send
over tcp stream

Backdoors

cd command



Problem

- *cd* changes working directory.
- check_output will execute it and return its result.
- Working directory will not change!

Solution:

- Implement this command using python.
- Use os.chdir(*path*)

Backdoors

Handling Errors

- If the client or server crashes, the connection will be lost.
- Backdoor crashes if:
  - Incorrect command is sent.
  - Correct command is miss-used.

Cross-Platform Hacking

- All programs we wrote are pure python programs
- They do not rely on OS specific resource.

Result:

- They work on any OS with a python interpreter.
- If packaged, they will work on any OS if even if python is NOT installed.

Packaging

- Convert python program into an executable that:
  - Packages all program files into a single executable.
  - Works without a python interpreter.
  - Get executed when double-clicked.

- For best results package the program from the same OS as the target.
  - EG if target is Windows then package the program from a Windows computer with a python interpreter.

Persistence

- Persistence programs start when the system starts.
  - Backdoors → maintain our access.
  - Keylogger → spy on target.
  - Reporters → send report on startup.
  - ....etc.

Trojans

- A trojan is a file that looks and functions as a normal file (image, pdf, song ..etc).

- When executed :
  1. Opens the normal file that the user expects.
  2. Executes evil code in the background (run a backdoor/keylooger ...etc).

Download & Execute Payload

- Generic executable that downloads & executes files.

- Ideas:
  - Download backdoor + keylogger.
  - Download keylogger + password recovery tool.
  - Download keylogger + password recovery tool + backdoor.
  - Use it as a trojan -- evil file + a normal file.

Download & Execute Payload

- Generic executable that downloads & executes files.

- Disadvantages :
  - User needs internet connection.
  - Files have to be uploaded and accessible via a direct URL.

Packaging – Creating Trojans

- Package front file with evil file.
- Extract front file at run time.
- Run front file from evil code.

Bypassing Anti-Virus Programs

AV programs detect viruses based on:

1. Code – compare files to huge database of signatures.
2. Behaviour – run file in a sandbox and analyse it.

Bypassing Anti-Virus Programs

AV programs detect viruses based on:

1. Code – compare files to huge database of signatures.

   → Use own code, obfuscation, useless operations, encode, pack ….etc
2. Behaviour – run file in a sandbox and analyse it.

Bypassing Anti-Virus Programs

AV programs detect viruses based on:

1. Code – compare files to huge database of signatures.

   → Use own code, obfuscation, useless operations, encode, pack ….etc
2. Behaviour – run file in a sandbox and analyse it.

   → Run trusted operations before evil code.

   → Delay execution of evil code.

Creating a Trojan

→ Combine evil file with normal file (image, book, song ...etc).

→ Configure evil file to run silently in the background.

→ Change file icon.

→ Change file extension.