Roey Gross - CTF Writeup – Hit&Run

Dear solver, please read some background information before starting:

https://en.wikipedia.org/wiki/Hit_%26_Run_(TV_series)

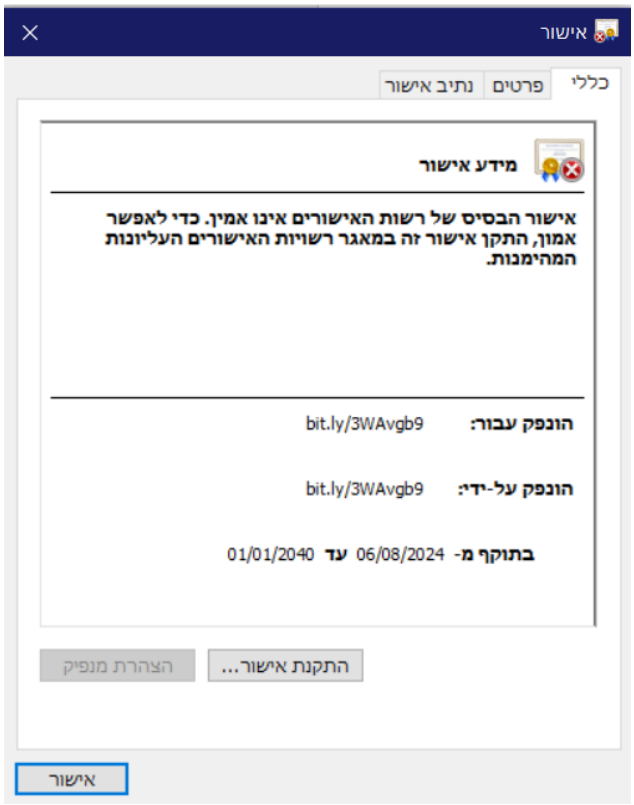
Now, lets starts solving what happened to Danielle!

There are two files.

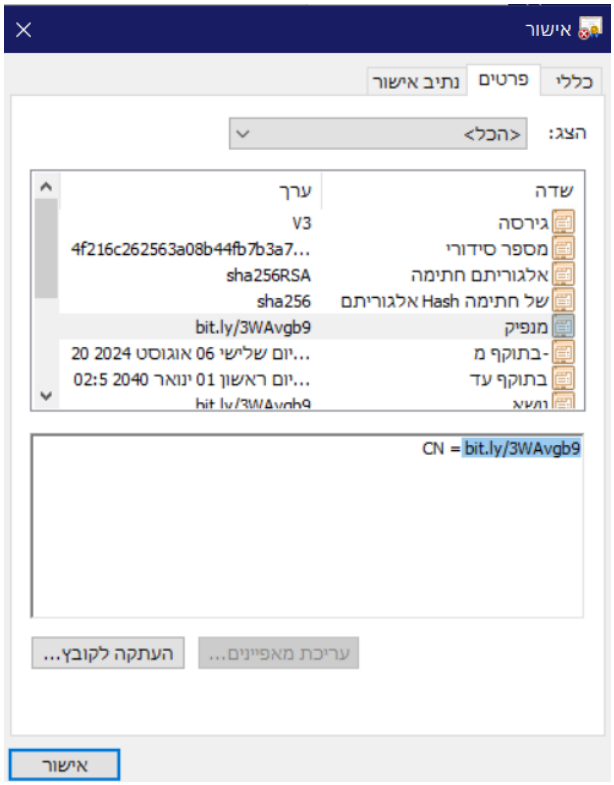
Danielle'sDiary.cer

Hit And Run.pdf

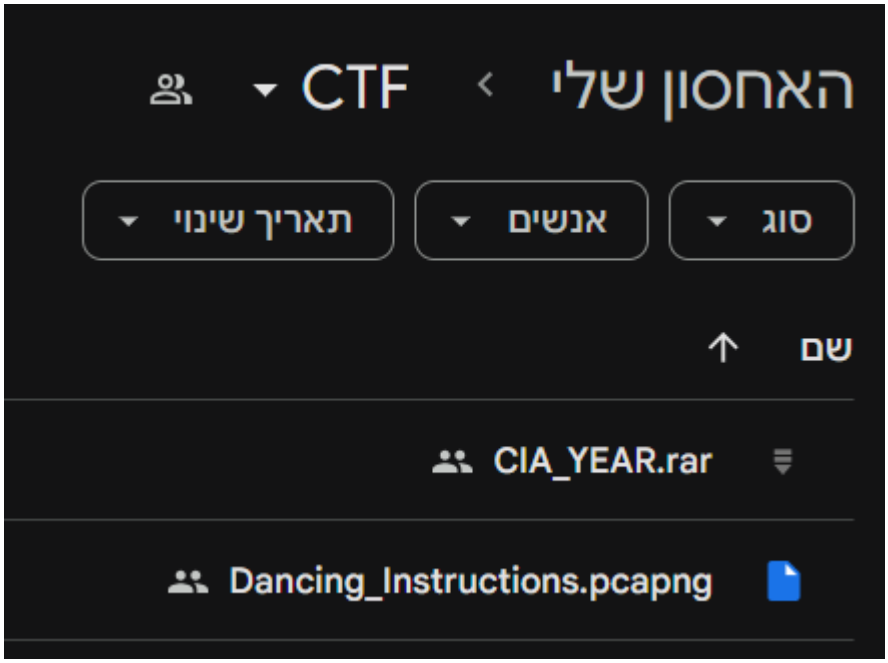
The PDF has a background story. It says that we need to investigate the certificate file. Reminder: We need to help Doron discover what happened to his wife. She left after her a certificate.
We can see that the certificate CN doesn't have a real CN name, but a link to unknown website:



To copy the CN we can move to the Properties tab and copy the name:



The link leads to a google drive page with 2 files, a rar which has an executable, and a Wireshark network capture:



(To download and solve the CTF, **disable** any antivirus. Any. If you are afraid use VM…).

We opened the Wireshark capture. First there is a DNS request for the CIA website.

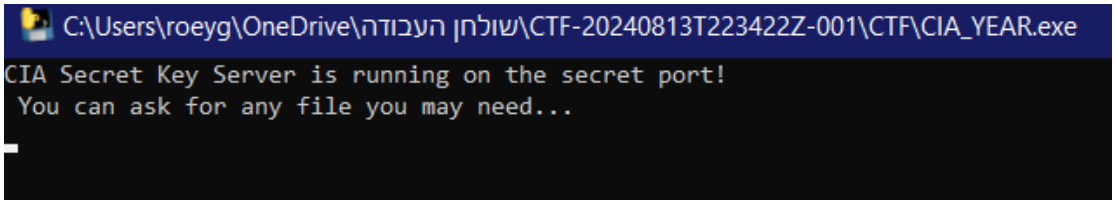
DNS	71				Standard query 0xd279 A www.cia.gov

Second there is an establishment of TLS communication between the CIA and local host. The connection is established, and encrypted data starts to flow between local host and CIA:

10.0.0.7	e6221.dscna.akamaie…	TLSv1.3	387	53825	Client Hello (SNI=www.cia.gov)
e6221.dscna.akamaie…	10.0.0.7	TLSv1.3	1506	443	Server Hello, Change Cipher Spec, Application Data
e6221.dscna.akamaie…	10.0.0.7	TLSv1.3	943	443	Application Data, Application Data, Application Data
10.0.0.7	e6221.dscna.akamaie…	TLSv1.3	134	53825	Change Cipher Spec, Application Data
10.0.0.7	e6221.dscna.akamaie…	TLSv1.3	146	53825	Application Data
10.0.0.7	e6221.dscna.akamaie…	TLSv1.3	586	53825	Application Data
e6221.dscna.akamaie…	10.0.0.7	TLSv1.3	325	443	Application Data
e6221.dscna.akamaie…	10.0.0.7	TLSv1.3	325	443	Application Data
e6221.dscna.akamaie…	10.0.0.7	TLSv1.3	115	443	Application Data

Probably, Danielle sent an encrypted message to the CIA. There is no further information in the pcapng. Let's move on to the second file.

We will run CIA_YEAR.exe



It looks like the exe created a new server running on local host on some port. The message says we need to find what port the exe connected to. It says also that the server is giving back any files needed.

We need to find the port number. Before searching the port number, we need to find the local Ip address which its ports we need to search. We will use ipconfig for that:

```
C:\Users\roeyg>ipconfig

Windows IP Configuration
```

```
IPv4 Address. . . . . . . . . . . : 192.168.56.1
```

To find the specific port we need to check which port is open. We can use Nmap for this. Nmap is sending SYN packets (the first step in a TCP handshake) to each port. If a port is open, the target responds with a SYN-ACK packet, indicating the port is open. This way we can find the open ports.

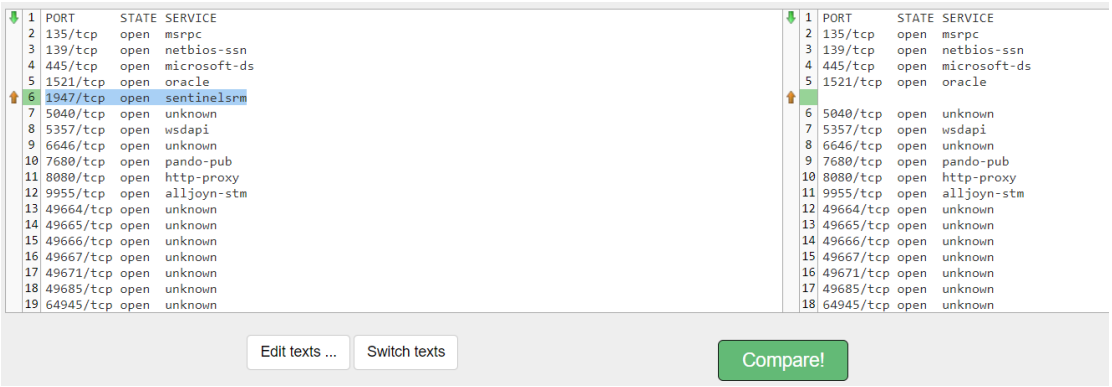
I opened WSL and run Nmap over there. The best way to discover what new ports has been opened after running the exe is to run Nmap before opening the server, after opening, and compare the results, to see what changed.
Before:

```
roeygross@DESKTOP-DF3CMB8:/mnt/c/Users/roeyg$ nmap -p-  192.168.56.1
Starting Nmap 7.80 ( https://nmap.org ) at 2024-08-14 02:00 IDT
Nmap scan report for DESKTOP-DF3CMB8 (192.168.56.1)
Host is up (0.0022s latency).
Not shown: 65518 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
1521/tcp  open  oracle
5040/tcp  open  unknown
5357/tcp  open  wsdapi
6646/tcp  open  unknown
7680/tcp  open  pando-pub
8080/tcp  open  http-proxy
9955/tcp  open  alljoyn-stm
49664/tcp open  unknown
49665/tcp open  unknown
49666/tcp open  unknown
49667/tcp open  unknown
49671/tcp open  unknown
49685/tcp open  unknown
64945/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 4.92 seconds
```

After:

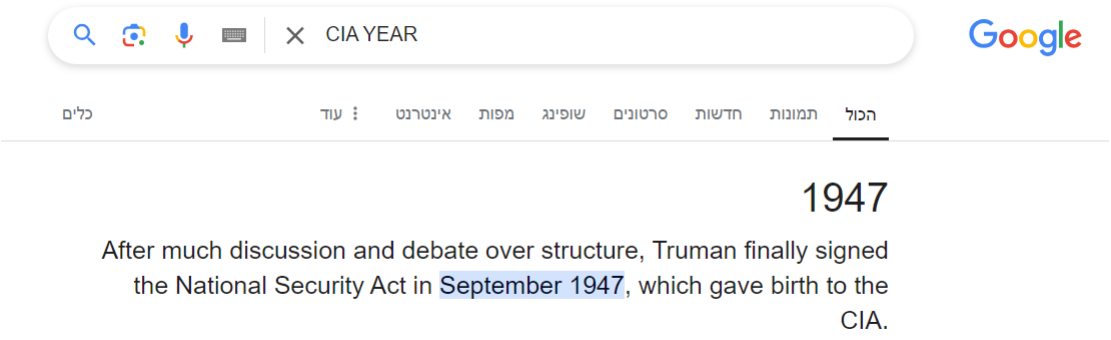
```
roeygross@DESKTOP-DF3CMB8:/mnt/c/Users/roeyg$ nmap -p-  192.168.56.1
Starting Nmap 7.80 ( https://nmap.org ) at 2024-08-14 02:01 IDT
Nmap scan report for DESKTOP-DF3CMB8 (192.168.56.1)
Host is up (0.0081s latency).
Not shown: 65517 closed ports
PORT      STATE SERVICE
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
1521/tcp  open  oracle
1947/tcp  open  sentinelsrm
5040/tcp  open  unknown
5357/tcp  open  wsdapi
6646/tcp  open  unknown
7680/tcp  open  pando-pub
8080/tcp  open  http-proxy
9955/tcp  open  alljoyn-stm
49664/tcp open  unknown
49665/tcp open  unknown
49666/tcp open  unknown
49667/tcp open  unknown
49671/tcp open  unknown
49685/tcp open  unknown
64945/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 6.84 seconds
```

We can check the differences on an online text compare:



The difference is that after opening the server, there is a new open port in 1947. It means that the server connected on local host on port 1947.

Shortcut for dummies: you can simply look for the clue in the exe name – CIA_YEAR, assume this is the port number, and try connecting:

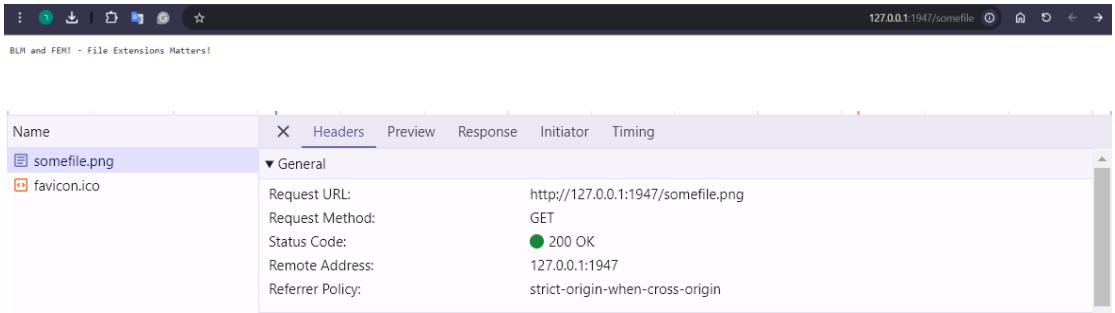


We will open the server and start to request files as the message said.
We can use tools like Scapy for creating, sending and receiving packets, but we will use chrome browser since the process is automated.

In our case Chrome browser simply creates packets, sends them and receives the response. We will try connecting local host on port 1947.

Chrome sends GET request to the server, then we get:

We can see that it is successfully connecting to the server and getting response from it. Let's try asking for some file from the server:



We asked for "somefile". We can see a returned message.

It seems that we need to request a file with its full extension. Let's try requesting a file called somefile.png:

The same message.

It looks like a dead end. We can write a python code and brute force with some common file names to see if there is any different response, **or** we can rethink.

We remember that the encrypted communication was a dead end too.

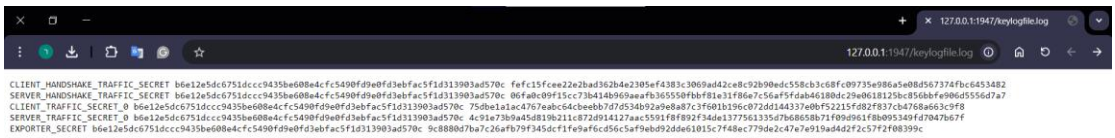
We remember that the exe server message was that the server is a key-providing server:

```
CIA Secret Key Server is running on the secret port!
You can ask for any file you may need...
```

Maybe we can receive the encrypted communication key from the server? We know that Wireshark can decrypt TLS encryption using a key file. If we will request the key file from the server and use it in Wireshark to decrypt the TLS communication, we will be able to see the messages Danielle sent to the CIA!

Usually, the key file is called keylogfile. Let's request it from the key server! We request keylogfile:

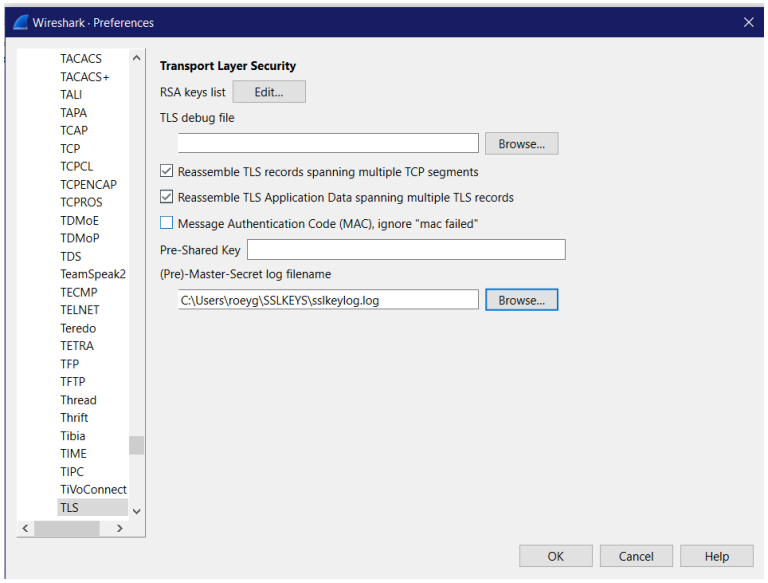
Nothing new. But we forgot that FEM – file extensions matters! Let's request keylogfile.log



BINGO! We got the keys! Now we can use them to decrypt the communication!

We created a new keylogfile.log

Then we decrypted the TLS communication using Wireshark:
Wireshark, Edit, Preferences, Protocols, TLS, then browse for our log file and press ok.



Amazing!

No.	Time	Source	Destination	Protocol	Length	Source Port	Info
1	0.000000	a184-25-50-168.depl…	10.0.0.7	TLSv1.2	78	443	Application Data
74	2.556958	10.0.0.7	e6221.dscna.akamaie…	TLSv1.3	387	53825	Client Hello (SNI=www.cia.gov)
77	2.716892	e6221.dscna.akamaie…	10.0.0.7	TLSv1.3	1506	443	Server Hello, Change Cipher Spec, Encrypted Extensions
79	2.718928	e6221.dscna.akamaie…	10.0.0.7	TLSv1.3	943	443	Certificate, Certificate Verify, Finished
81	2.721948	10.0.0.7	e6221.dscna.akamaie…	TLSv1.3	134	53825	Change Cipher Spec, Finished
82	2.722208	10.0.0.7	e6221.dscna.akamaie…	HTTP2	146	53825	Magic, SETTINGS[0], WINDOW_UPDATE[0]
83	2.722393	10.0.0.7	e6221.dscna.akamaie…	HTTP2	586	53825	HEADERS[1]: GET /send/reece/bit.ly/3ytoHiC

As you can see in green, the encrypted TLS communication is decrypted to readable HTTP communication!

Further investigation on the HTTP packets leads us to this odd GET request from Danielle to the CIA.

10.0.0.7	e6221.dscna.akamaie…	HTTP2	586	53825	HEADERS[1]: GET /send/reece/bit.ly/3ytoHiC

Local host sent to the CIA website a GET request about sending an odd link to Reece (her boss). Why was Danielle using a GET to send a message? she should have used POST! Anyway, the CIA receives a GET for Reece…

Later in the capture, we see that the CIA "doesn't know" about Reece and its link:

HTTP2	537	443	HEADERS[1]: 404 Not Found

Anyways, it looks like the link may has the secret data were looking for. Let's open the link.
The link leads to google docs page, with some gibberish:

TOP SECRET

OSITNAQ: SMRNGQ: PFFNEPCQN NWQMCAQPHG CGE ONASMPQX IMNCAY
ENCM EPMNAQHM MNNAN,
P YHJN QYPO FNOOCRN KPGEO XHS JMHFJQDX. P CF VMPQPGR SGENM
NWQMNFN ESMNOO CGE SMRNGAX. FX ASMMNGQ FPOOPHG PG QND CUPU YCO
QCBNG C QSMG QYCO MNLSPMNO PFFNEPCQN CQQNGQPHG CGE CAQPHG.
CO XHS CMN CVCMN, FX CDPCO YNMN YCO INNG ECGPNDDN VNWDNM, C
ECGANM VPQY C ONNFPGRDX PGGHASHSO AHUNM. YHVNUNM, MNANGQ
ENUNDHJFNGQO YCUN AHFJMHFPONE FX JHOPQPHG CGE YCUN JSQ FX DPKN
CGE QYN FPOOPHG CQ OPRGPKPACGQ MPOB. QYN ENQCPDO HK QYNON
ENUNDHJFNGQO CMN CO KHDDHVO:

It contains repeated sequences and words, which suggests that common English words or letter patterns are being replaced by consistent ciphertext equivalents.

The structure of the sentences, with apparent punctuation and capitalization, mimics natural language, which is characteristic of substitution ciphers.

Let's try decrypting this substitution cipher. We can write python code and find each letter frequency to find the substitution key, and then substitute the letters, or we can use online decoder:

MONOALPHABETIC SUBSTITUTION DECODER

ALPHABETIC SUBSTITUTION CIPHERTEXT

```
QH CUHPE C DCMRNM PGQNMGCQPHGCD PGAPENGQ.
CVCPQPGR XHSM PFFNEPCQN MNOJHGON.
MNRCMEO,
CRNGQ O.
```

SPACES ARE RELEVANT AND MUST BE KEPT (ARISTOCRAT CIPHER)

CAN BE IGNORED OR ARE MISSING (PATRISTOCRAT CIPHER)

PLAINTEXT LANGUAGE English

DECRYPT AUTOMATICALLY

Results

dCode tried to find the correct alphabet and its substitution automatically. The result is a draft that should allow you to perform the decryption manually by indicating letters in each cell.

```
SUBJECT: URGENT: IMMEDIATE EXTRACTION AND
SECURITY BREACH DEAR DIRECTOR REECE, I HOPE
THIS MESSAGE FINDS YOU PROMPTLY. I AM WRITING
UNDER EXTREME DURESS AND URGENCY. MY CURRENT
MISSION IN TEL AVIV HAS TAKEN A TURN THAT
REQUIRES IMMEDIATE ATTENTION AND ACTION. AS
```

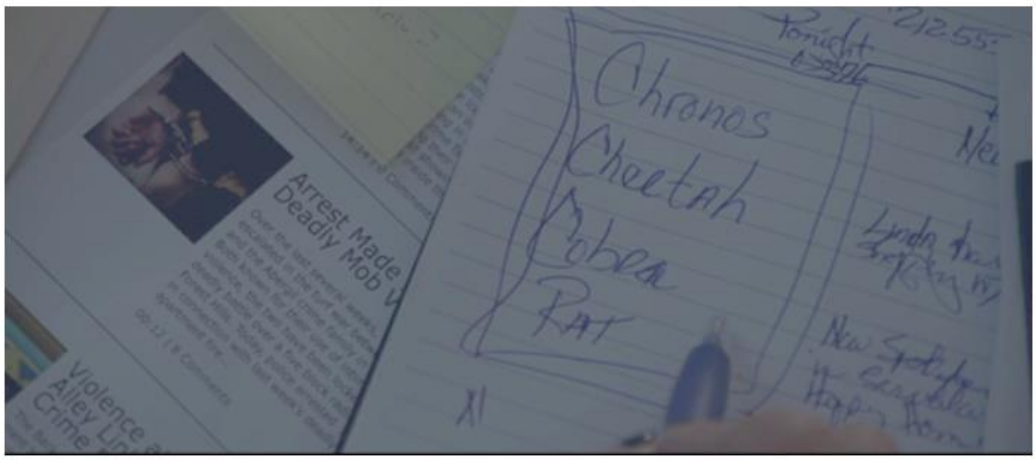
It found that the key is:

1 CIAENKRYPTBDFGHJLMOQSUVWXZ

And decrypted the cipher. We can see that the message has a link:

```
SENSITIVE INFORMATION: THIS INFORMATION IS
TOO SENSITIVE AND CANNOT FALL INTO THE WRONG
HANDS. HERE IS A LINK TO THE INFORMATION I
FOUND: BIT.LY/3LX9KSA CURRENT SITUATION: I AM
```

The link leads to an image:



RickRolled

It looks like someone is pointing at a square that has the words "Chronos Cheetah Cobra RAT". It sounds like a cipher. Let's try look for it on google.

Ending! A quick look at the websites explains what the cipher means:

Chronos is US president, Cheetah is the president daughter, and Cobra is the son-in-law. But who was RAT?
R-A-T is an acronym for Remote Access Trojan, a piece of spyware used by hackers. The Israelis were spying on the White House through RAT.

Israeli Mossad managed to decrypt the communication between Danielle and her boss Reece, but for some reason they monitored Danielle's POST messages <u>only</u> and not her GET messages. By sending a GET request for a pre-known data (a letter which is saved on the CIA servers), Danielle bypassed the Mossad monitoring and managed to warn Reece that the Israelis are spying on the white house. Reece had multiple pre-calculated

scenarios on his page on the CIA website. Daniel just needed to send GET to the right one to acknowledge Reece about what was really going on!

Congratulations! We have figured out what CIA agent Danielle discovered before she was killed by Israeli Mossad by a "hit run accident"!
Segev wants to thank you for helping him discover the truth.

See you in season 2 of Hit&Run.