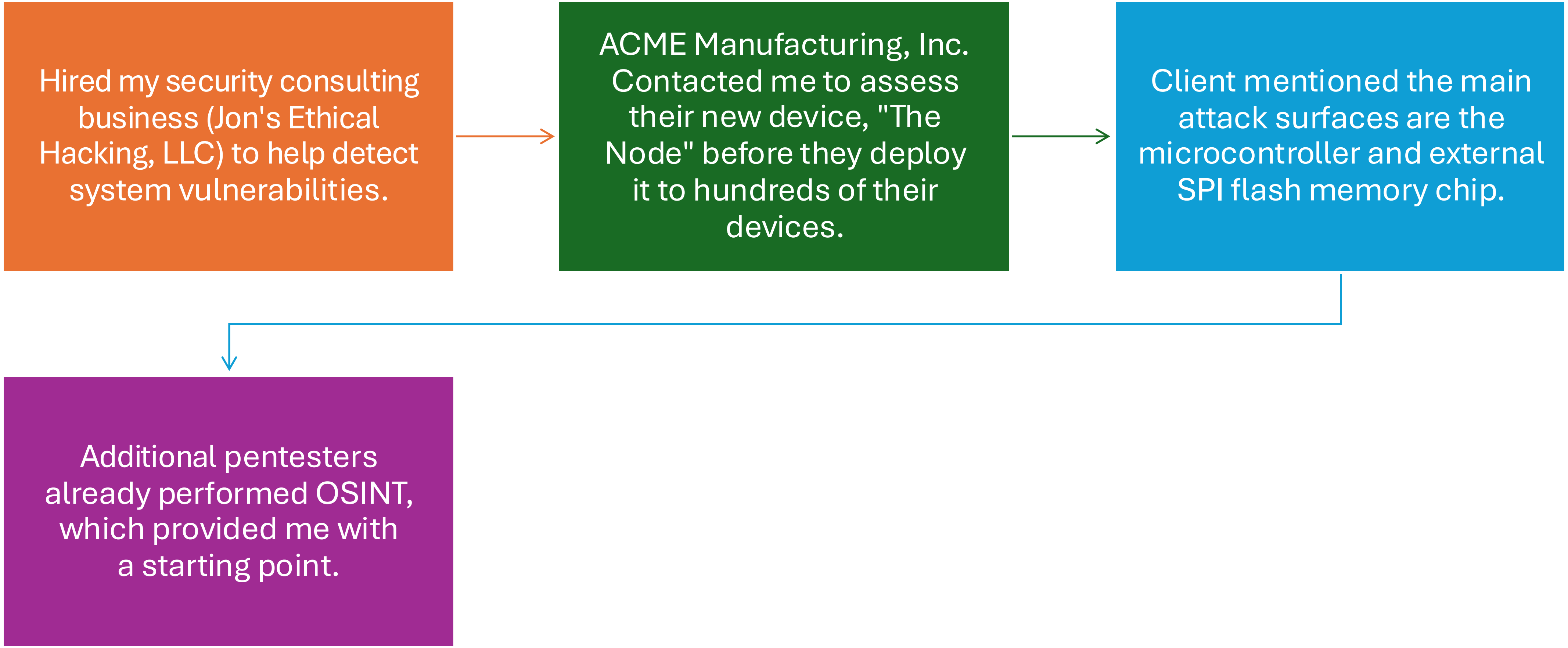
COSC 6840 – Final Project

Ethical Hacking Theory & Practice

Jonathan Menzel

Project Overview



Hired my security consulting business (Jon's Ethical Hacking, LLC) to help detect system vulnerabilities.

ACME Manufacturing, Inc. Contacted me to assess their new device, "The Node" before they deploy it to hundreds of their devices.

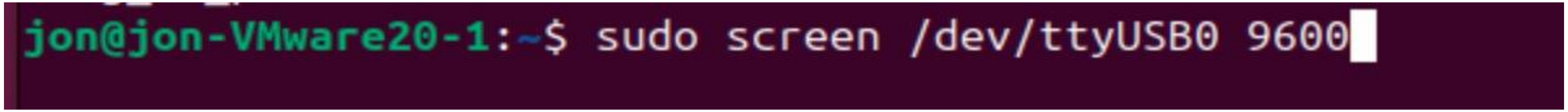
Client mentioned the main attack surfaces are the microcontroller and external SPI flash memory chip.

Additional pentesters already performed OSINT, which provided me with a starting point.

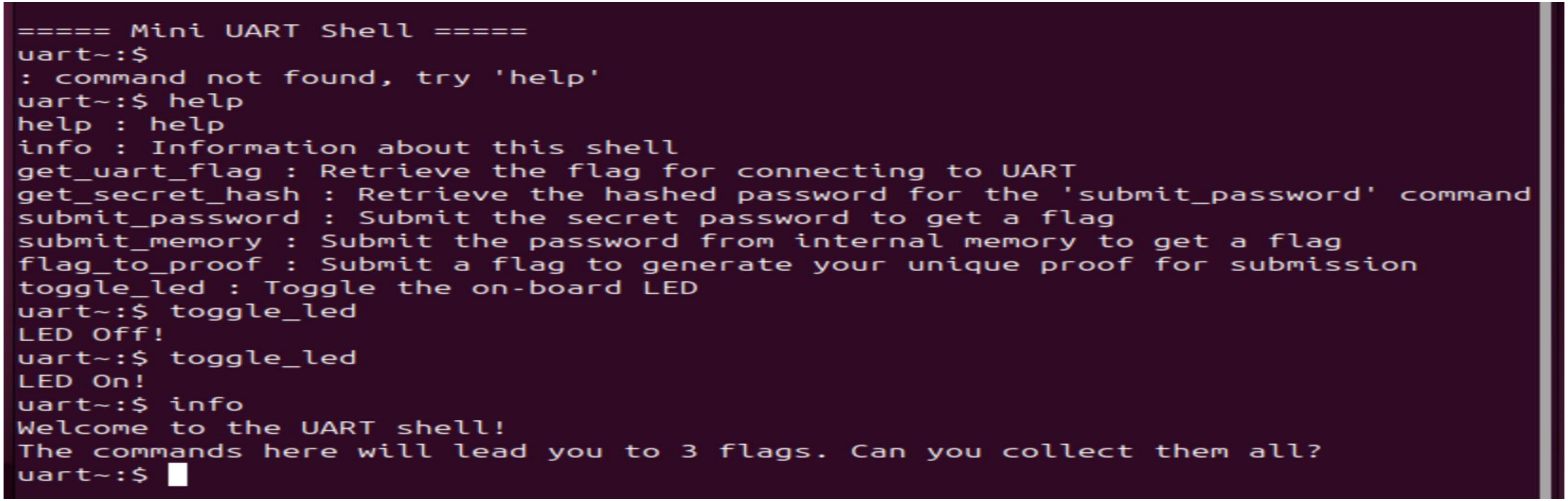
Methodology and Tools

- Connected to the Node using a USB-to-Serial adapter on the USART2 RX, TX, and GND pins
- Used screen at 9600 baud to establish the UART connection
- Accessed the UART shell and tested the commands, including the flag submission command
- Kali Linux/John the Ripper to crack the password hash found
- Used the cracked passwords in the UART shell to retrieve the password flag and collected all UART-based flags – not able to find Secret Stream Flag

Connected Shell



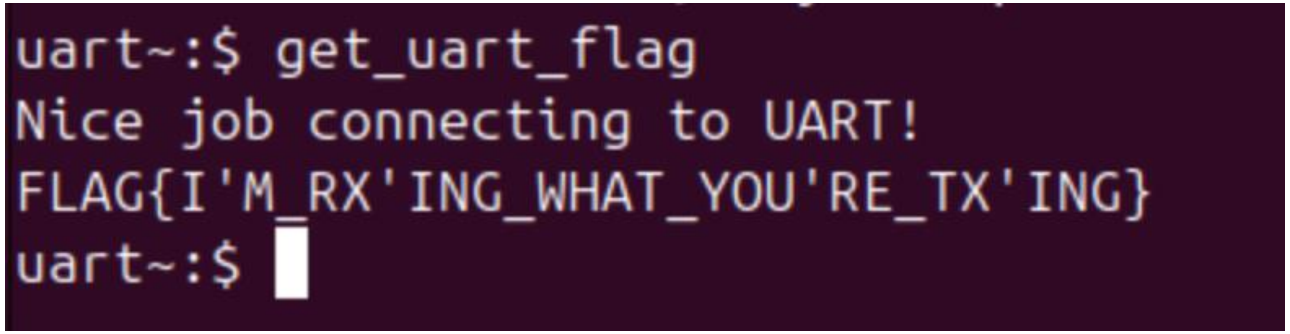
```
jon@jon-VMware20-1:~$ sudo screen /dev/ttyUSB0 9600
```



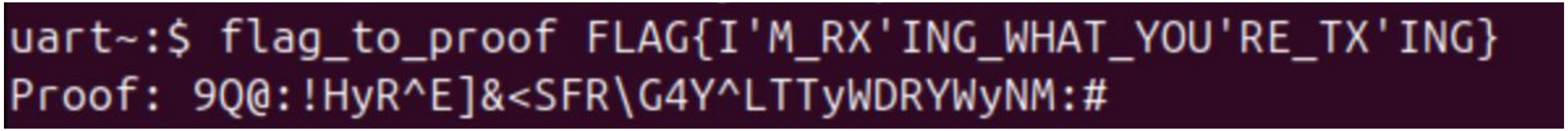
```
===== Mini UART Shell =====
uart~:$
: command not found, try 'help'
uart~:$ help
help : help
info : Information about this shell
get_uart_flag : Retrieve the flag for connecting to UART
get_secret_hash : Retrieve the hashed password for the 'submit_password' command
submit_password : Submit the secret password to get a flag
submit_memory : Submit the password from internal memory to get a flag
flag_to_proof : Submit a flag to generate your unique proof for submission
toggle_led : Toggle the on-board LED
uart~:$ toggle_led
LED Off!
uart~:$ toggle_led
LED On!
uart~:$ info
Welcome to the UART shell!
The commands here will lead you to 3 flags. Can you collect them all?
uart~:$
```

Connected through USB-to-Serial at 9600 baud and accessed the debug shell

UART Flag

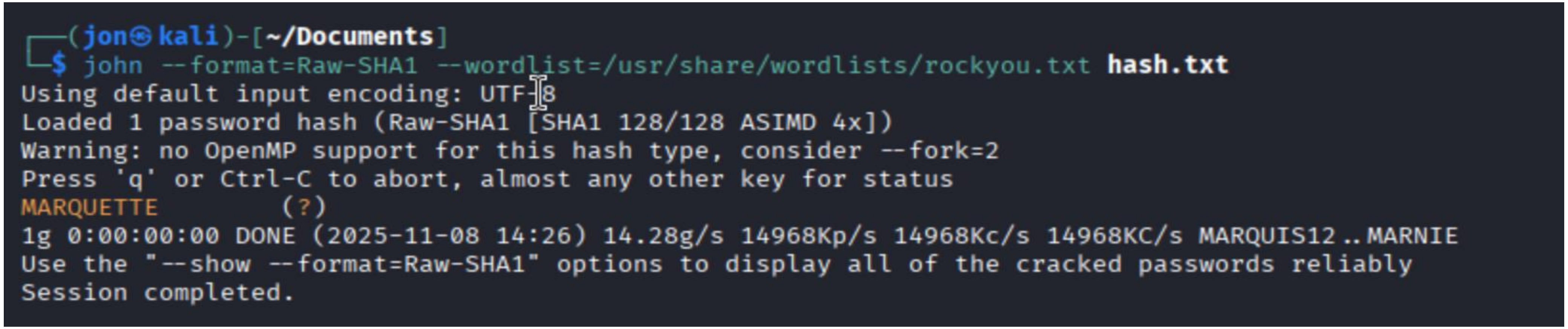


```
uart~:$ get_uart_flag
Nice job connecting to UART!
FLAG{I'M_RX'ING_WHAT_YOU'RE_TX'ING}
uart~:$ 
```

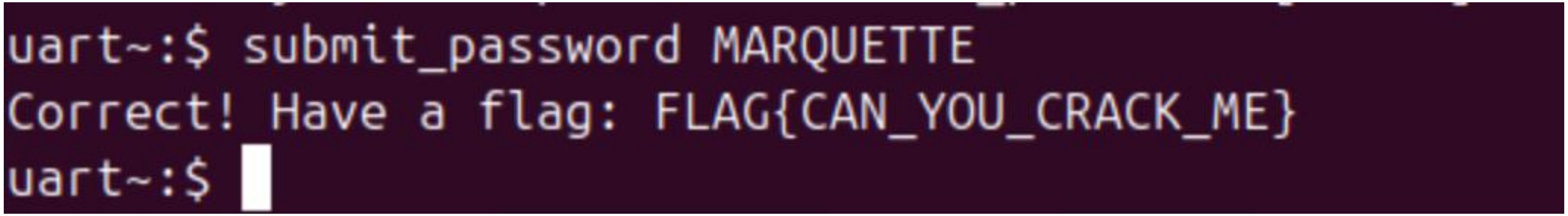


```
uart~:$ flag_to_proof FLAG{I'M_RX'ING_WHAT_YOU'RE_TX'ING}
Proof: 9Q@:!HyR^E]&<SFR\G4Y^LTTyWDRYWyNM:#
```

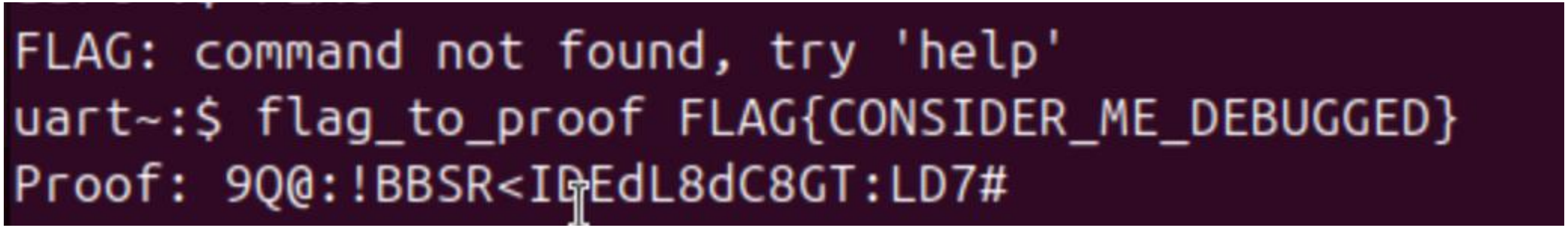
Password Cracking Flag



```
┌──(jon㉿kali)-[~/Documents]
└─$ john --format=Raw-SHA1 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-SHA1 [SHA1 128/128 ASIMD 4x])
Warning: no OpenMP support for this hash type, consider --fork=2
Press 'q' or Ctrl-C to abort, almost any other key for status
MARQUETTE        (?)
1g 0:00:00:00 DONE (2025-11-08 14:26) 14.28g/s 14968Kp/s 14968Kc/s 14968KC/s MARQUIS12..MARNIE
Use the "--show --format=Raw-SHA1" options to display all of the cracked passwords reliably
Session completed.
```

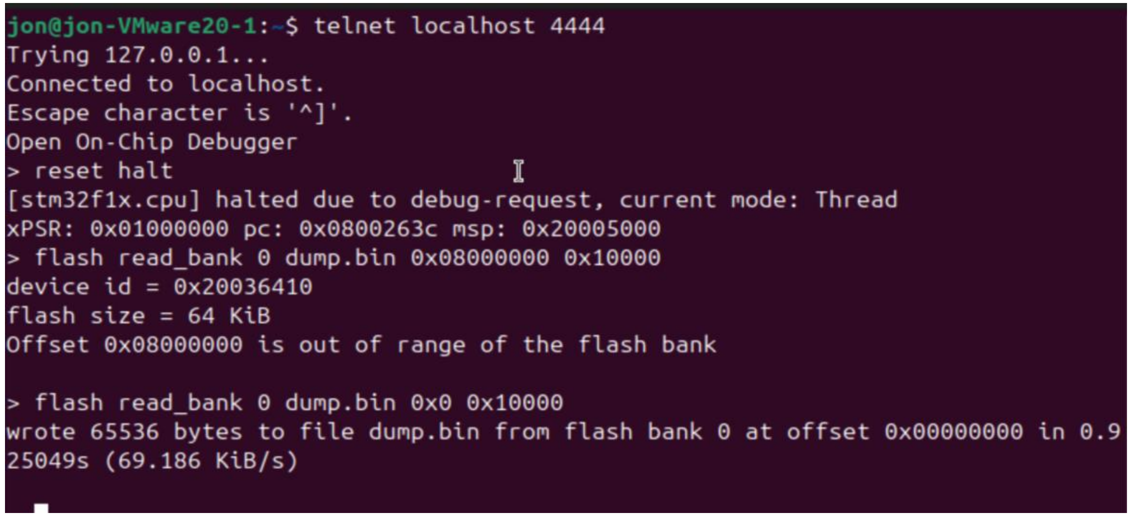


```
uart~:$ submit_password MARQUETTE
Correct! Have a flag: FLAG{CAN_YOU_CRACK_ME}
uart~:$
```

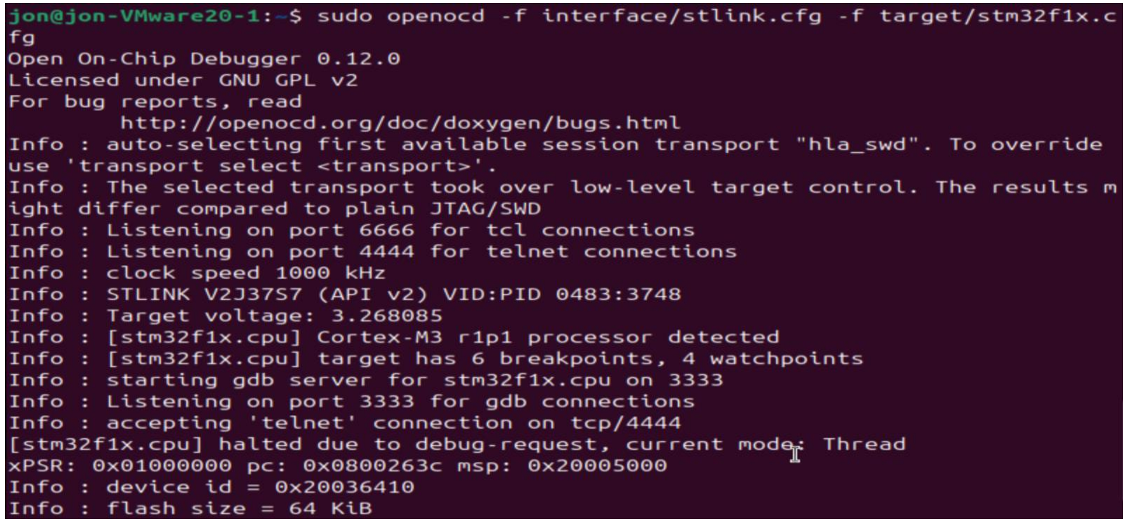


```
FLAG: command not found, try 'help'
uart~:$ flag_to_proof FLAG{CONSIDER_ME_DEBUGGED}
Proof: 9Q@:!BBSR<IDEdL8dC8GT:LD7#
```

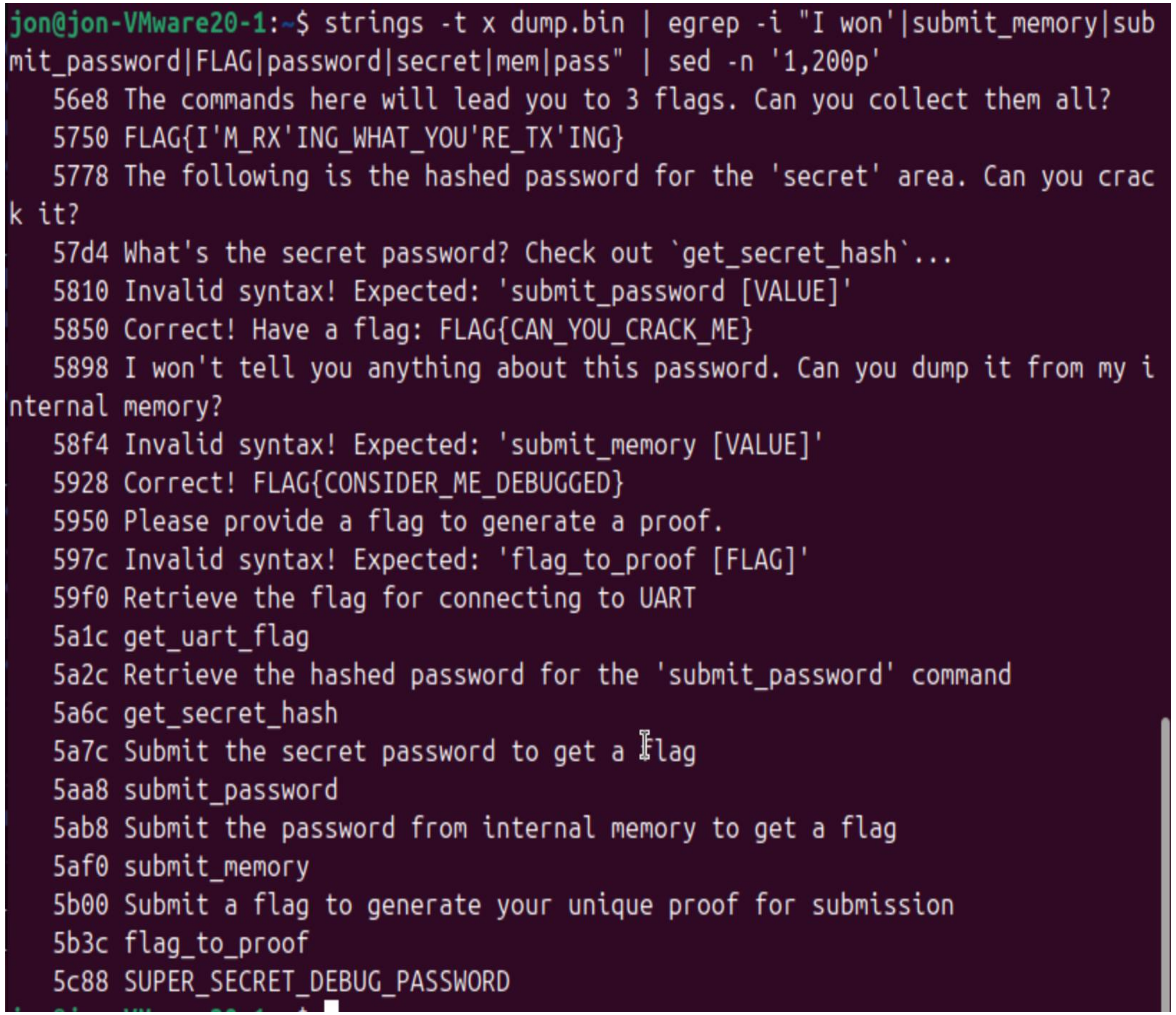
Internal Flash Memory Flag – PT.1



```
jon@jon-VMware20-1:~$ telnet localhost 4444
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Open On-Chip Debugger
> reset halt
[stm32f1x.cpu] halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0800263c msp: 0x20005000
> flash read_bank 0 dump.bin 0x08000000 0x10000
device id = 0x20036410
flash size = 64 KiB
Offset 0x08000000 is out of range of the flash bank

> flash read_bank 0 dump.bin 0x0 0x10000
wrote 65536 bytes to file dump.bin from flash bank 0 at offset 0x00000000 in 0.9
25049s (69.186 KiB/s)
```

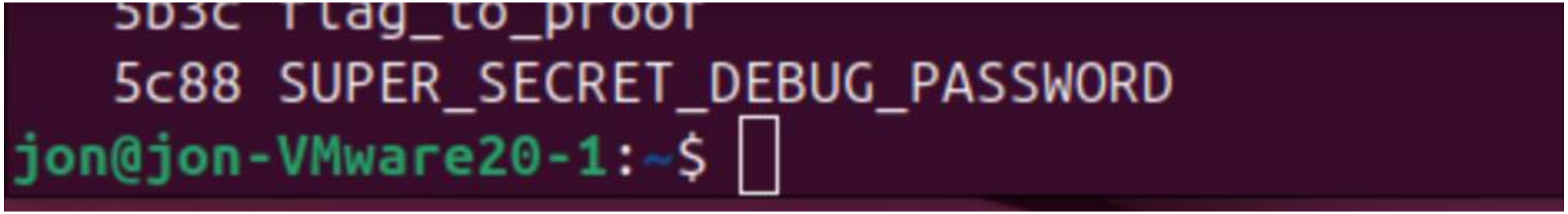


```
jon@jon-VMware20-1:~$ sudo openocd -f interface/stlink.cfg -f target/stm32f1x.c
fg
Open On-Chip Debugger 0.12.0
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
Info : auto-selecting first available session transport "hla_swd". To override
use 'transport select <transport>'.
Info : The selected transport took over low-level target control. The results m
ight differ compared to plain JTAG/SWD
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : clock speed 1000 kHz
Info : STLINK V2J37S7 (API v2) VID:PID 0483:3748
Info : Target voltage: 3.268085
Info : [stm32f1x.cpu] Cortex-M3 r1p1 processor detected
Info : [stm32f1x.cpu] target has 6 breakpoints, 4 watchpoints
Info : starting gdb server for stm32f1x.cpu on 3333
Info : Listening on port 3333 for gdb connections
Info : accepting 'telnet' connection on tcp/4444
[stm32f1x.cpu] halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x0800263c msp: 0x20005000
Info : device id = 0x20036410
Info : flash size = 64 KiB
```

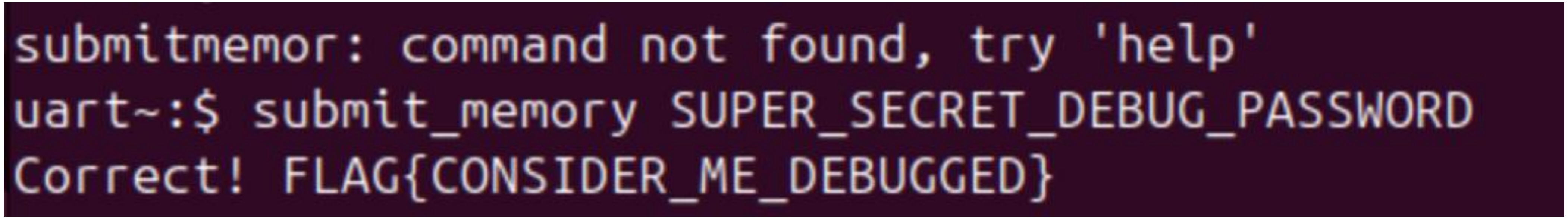


```
jon@jon-VMware20-1:~$ strings -t x dump.bin | egrep -i "I won'|submit_memory|sub
mit_password|FLAG|password|secret|mem|pass" | sed -n '1,200p'
   56e8 The commands here will lead you to 3 flags. Can you collect them all?
   5750 FLAG{I'M_RX'ING_WHAT_YOU'RE_TX'ING}
   5778 The following is the hashed password for the 'secret' area. Can you crac
k it?
   57d4 What's the secret password? Check out `get_secret_hash`...
   5810 Invalid syntax! Expected: 'submit_password [VALUE]'
   5850 Correct! Have a flag: FLAG{CAN_YOU_CRACK_ME}
   5898 I won't tell you anything about this password. Can you dump it from my i
nternal memory?
   58f4 Invalid syntax! Expected: 'submit_memory [VALUE]'
   5928 Correct! FLAG{CONSIDER_ME_DEBUGGED}
   5950 Please provide a flag to generate a proof.
   597c Invalid syntax! Expected: 'flag_to_proof [FLAG]'
   59f0 Retrieve the flag for connecting to UART
   5a1c get_uart_flag
   5a2c Retrieve the hashed password for the 'submit_password' command
   5a6c get_secret_hash
   5a7c Submit the secret password to get a flag
   5aa8 submit_password
   5ab8 Submit the password from internal memory to get a flag
   5af0 submit_memory
   5b00 Submit a flag to generate your unique proof for submission
   5b3c flag_to_proof
   5c88 SUPER_SECRET_DEBUG_PASSWORD
```

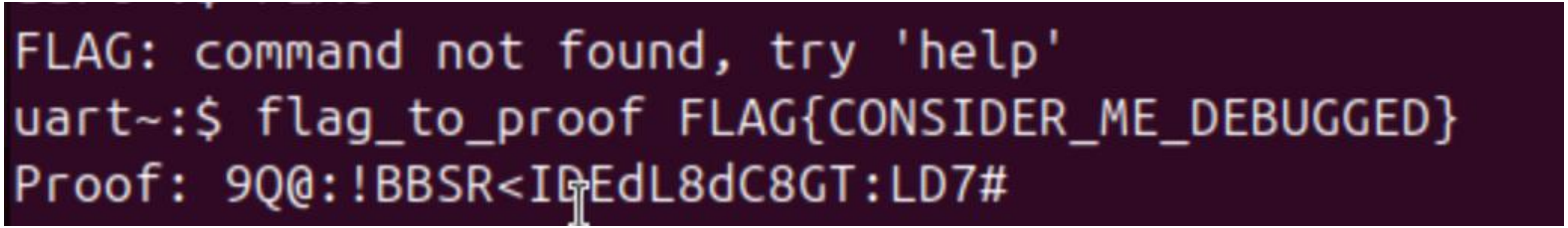
Internal Flash Memory Flag – PT.2



```
   5b3c flag_to_proof
   5c88 SUPER_SECRET_DEBUG_PASSWORD
jon@jon-VMware20-1:~$ 
```

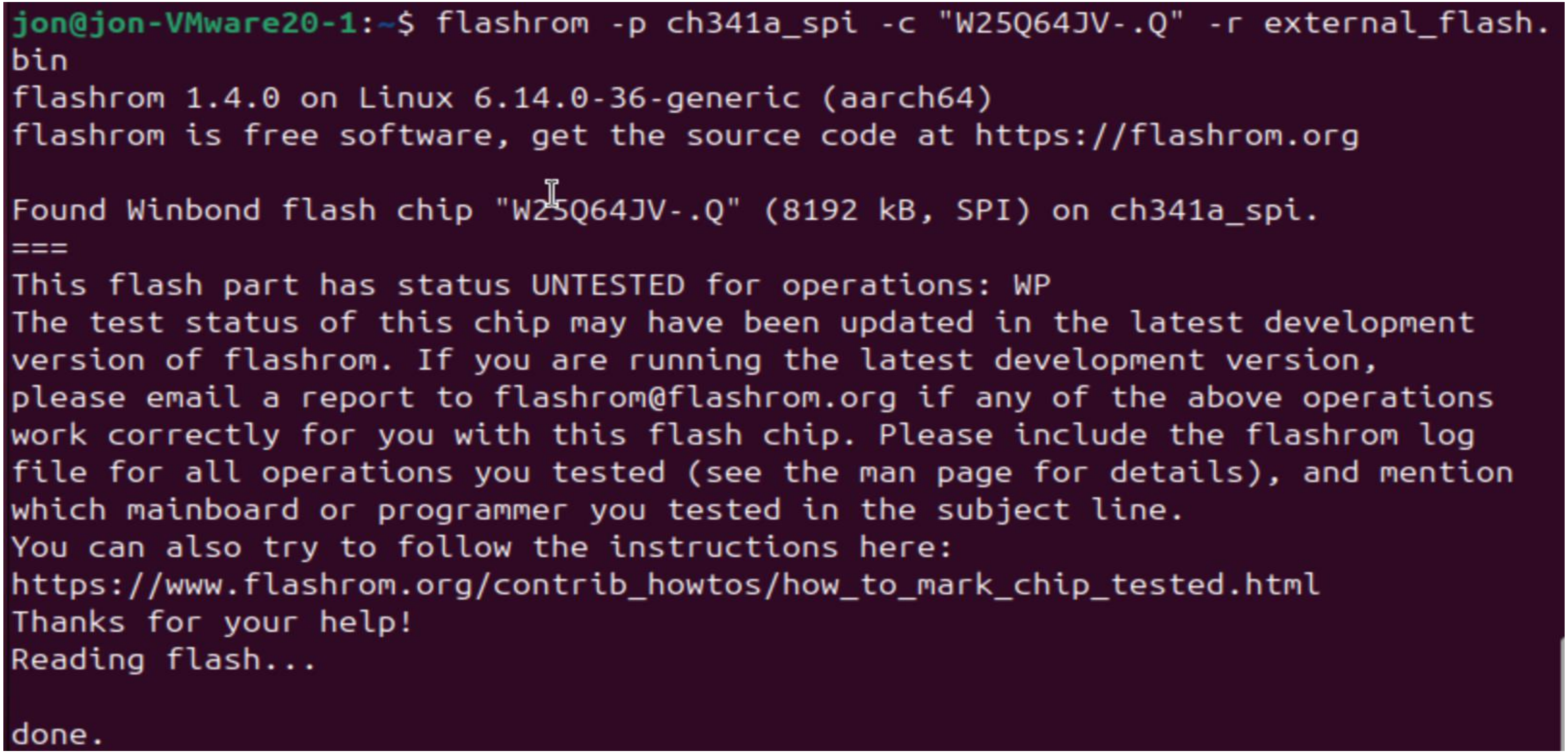


```
submitmemor: command not found, try 'help'
uart~:$ submit_memory SUPER_SECRET_DEBUG_PASSWORD
Correct! FLAG{CONSIDER_ME_DEBUGGED}
```



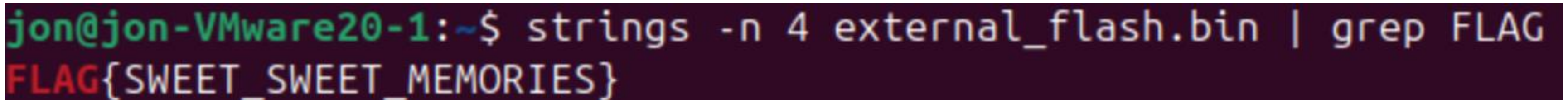
```
FLAG: command not found, try 'help'
uart~:$ flag_to_proof FLAG{CONSIDER_ME_DEBUGGED}
Proof: 9Q@:!BBSR<IDEdL8dC8GT:LD7#
```

External Flash – PT.1



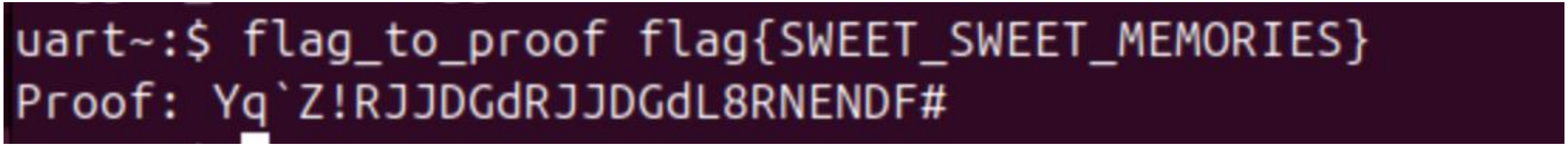
```
jon@jon-VMware20-1:~$ flashrom -p ch341a_spi -c "W25Q64JV-.Q" -r external_flash.
bin
flashrom 1.4.0 on Linux 6.14.0-36-generic (aarch64)
flashrom is free software, get the source code at https://flashrom.org

Found Winbond flash chip "W25Q64JV-.Q" (8192 kB, SPI) on ch341a_spi.
===
This flash part has status UNTESTED for operations: WP
The test status of this chip may have been updated in the latest development
version of flashrom. If you are running the latest development version,
please email a report to flashrom@flashrom.org if any of the above operations
work correctly for you with this flash chip. Please include the flashrom log
file for all operations you tested (see the man page for details), and mention
which mainboard or programmer you tested in the subject line.
You can also try to follow the instructions here:
https://www.flashrom.org/contrib_howtos/how_to_mark_chip_tested.html
Thanks for your help!
Reading flash...

done.
```

External Flash – PT.2



```
jon@jon-VMware20-1:~$ strings -n 4 external_flash.bin | grep FLAG
FLAG{SWEET_SWEET_MEMORIES}
```

Went back to UART shell and entered in flag…



```
uart~:$ flag_to_proof flag{SWEET_SWEET_MEMORIES}
Proof: Yq`Z!RJJDGdRJJDGdL8RNENDF#
```