CSE3140 — Lab 5

Mike Medved, Nithila Annadurai

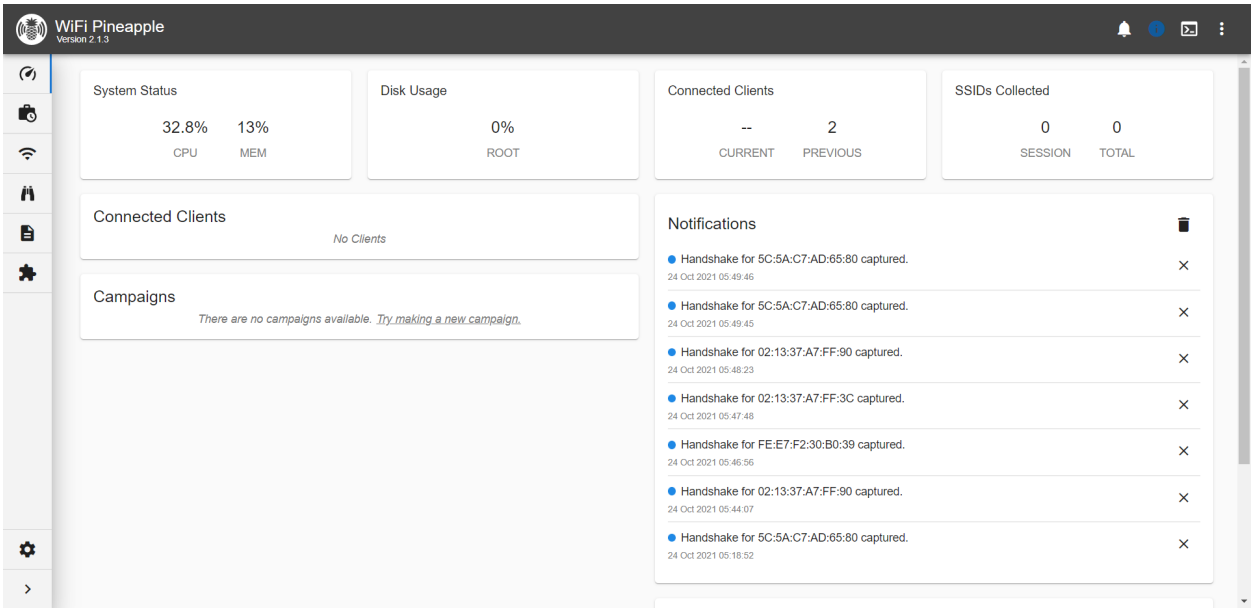
November 19th, 2022

Deliverables

Part 1

Below are the various fields displayed on the main dashboard of the Pineapple we used to conduct the lab, followed by a screenshot of the main dashboard itself.

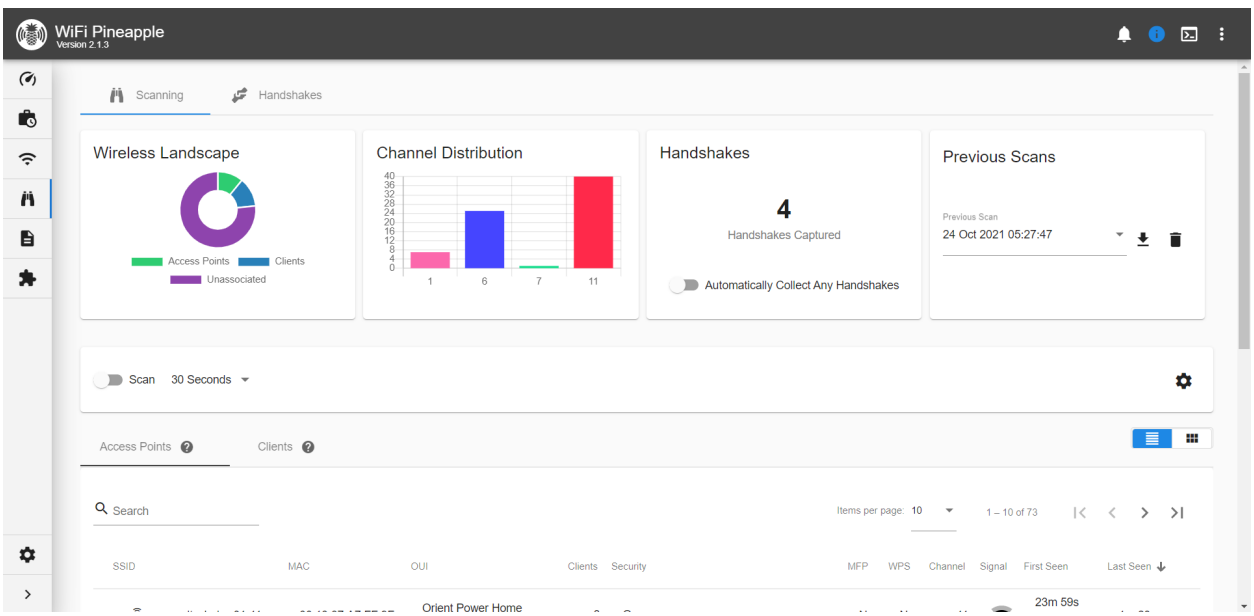
Setting	**Explanation**
System Status	Displays the current CPU and RAM usage percentages
Disk Usage	Displays the current disk usage percentage
Connected Clients	Displays each client that is currently, and was previously associated with the Pineapple
SSIDs Collected	Displays all SSIDs collected by the Pineapple since bootup
Campaigns	Displays a list of current campaigns, including their status, types, and names
Notifications	Displays all status notifications collected by the Pineapple regarding it's operation(s)

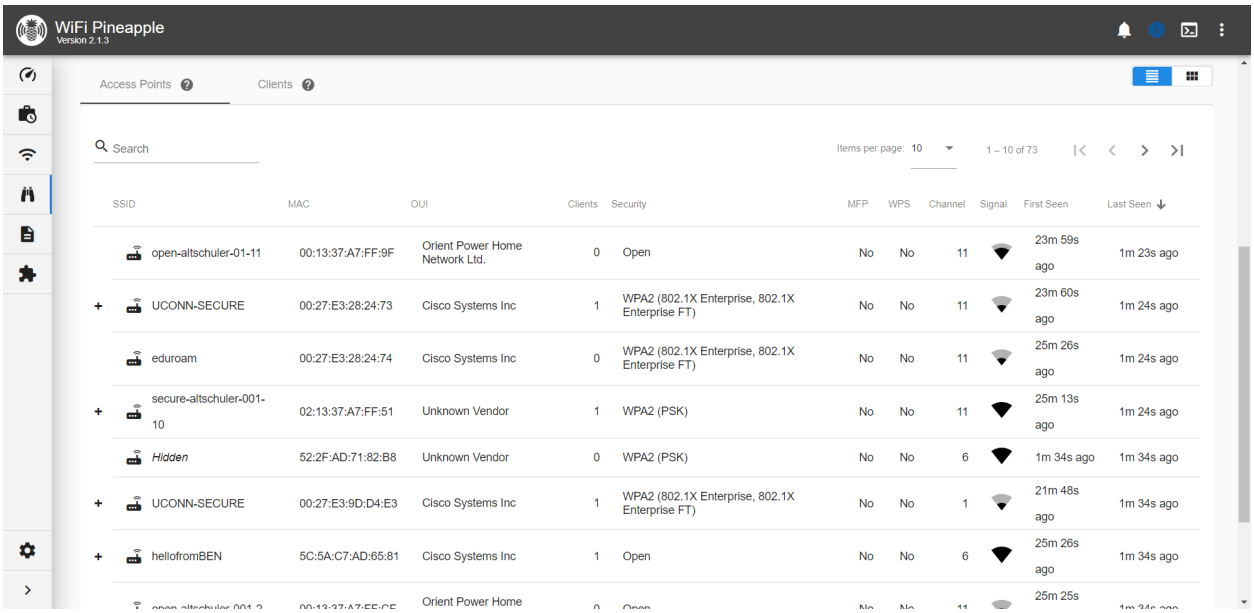


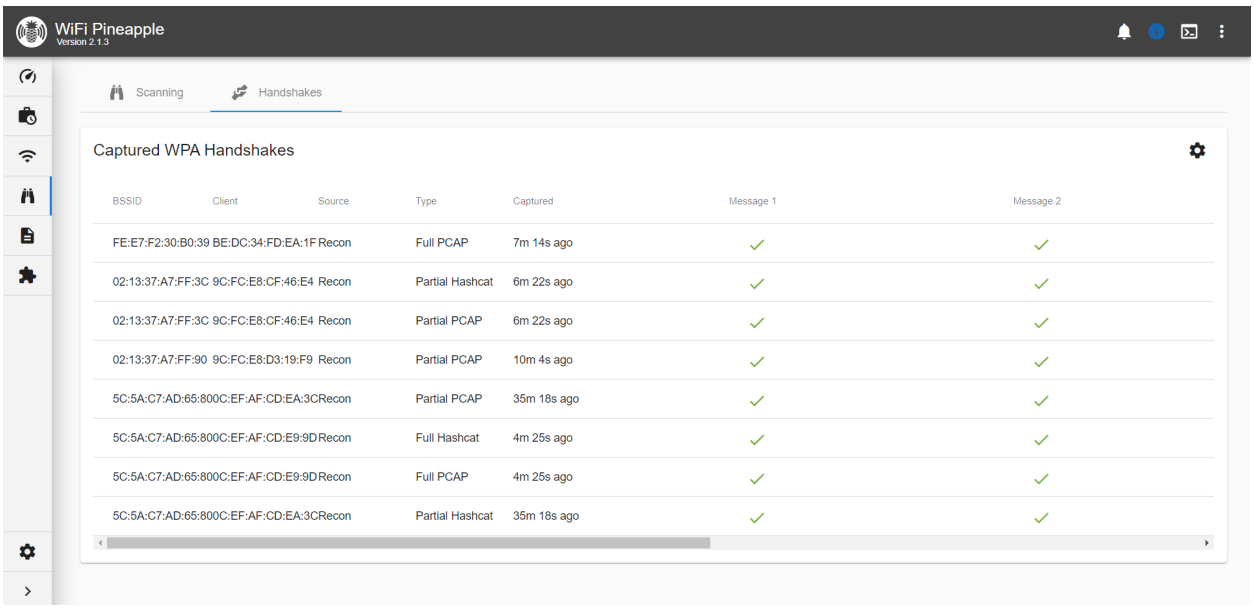
Part 2

Below are the various fields found on the Recon dashboard of the Pineapple we used to conduct the lab, followed by screenshots of the dashboard.

Page	Field	Explanation
Scanning	Wireless Landscape	Displays a pie-chart containing the number APs, clients, and unassociated clients
	Channel Distribution	Displays channel frequencies picked up by the Pineapple's antennas
	Handshakes	Displays the number of handshakes captured during a Recon scan
	Previous Scans	Displays the date and information regarding previous Recon scans
	Access Points	Displays information regarding detected APs (SSID, MAC, OUI, clients, security, MFP, WPS, channel, signal, and first + last seen times)
	Clients	Displays information regarding all detected clients during scans (IP, MAC, time of connection)
Handshakes	Page Information	Displays information regarding captured WPA handshakes (BSSID, client, source, type, time since capture, message1, message2)




WiFi Pineapple
Version 2.1.3
Access Points
Clients
Search
Items per page: 10
1 – 10 of 73
SSID MAC OUI Clients Security MFP WPS Channel Signal First Seen Last Seen
open-altschuler-01-11 00:13:37:A7:FF:9F Orient Power Home Network Ltd. 0 Open No No 11 23m 59s ago 1m 23s ago
UCONN-SECURE 00:27:E3:28:24:73 Cisco Systems Inc 1 WPA2 (802.1X Enterprise, 802.1X Enterprise FT) No No 11 23m 60s ago 1m 24s ago
eduroam 00:27:E3:28:24:74 Cisco Systems Inc 0 WPA2 (802.1X Enterprise, 802.1X Enterprise FT) No No 11 25m 26s ago 1m 24s ago
secure-altschuler-001-10 02:13:37:A7:FF:51 Unknown Vendor 1 WPA2 (PSK) No No 11 25m 13s ago 1m 24s ago
Hidden 52:2F:AD:71:82:B8 Unknown Vendor 0 WPA2 (PSK) No No 6 1m 34s ago 1m 34s ago
UCONN-SECURE 00:27:E3:9D:D4:E3 Cisco Systems Inc 1 WPA2 (802.1X Enterprise, 802.1X Enterprise FT) No No 1 21m 48s ago 1m 34s ago
hellofromBEN 5C:5A:C7:AD:65:81 Cisco Systems Inc 1 Open No No 6 25m 26s ago 1m 34s ago
Orient Power Home
25m 25s


WiFi Pineapple
Version 2.1.3
Scanning
Handshakes
Captured WPA Handshakes
BSSID Client Source Type Captured Message 1 Message 2
FE:E7:F2:30:B0:39 BE:DC:34:FD:EA:1F Recon Full PCAP 7m 14s ago
02:13:37:A7:FF:3C 9C:FC:E8:CF:46:E4 Recon Partial Hashcat 6m 22s ago
02:13:37:A7:FF:3C 9C:FC:E8:CF:46:E4 Recon Partial PCAP 6m 22s ago
02:13:37:A7:FF:90 9C:FC:E8:D3:19:F9 Recon Partial PCAP 10m 4s ago
5C:5A:C7:AD:65:80 0C:EF:AF:CD:EA:3C Recon Partial PCAP 35m 18s ago
5C:5A:C7:AD:65:80 0C:EF:AF:CD:E9:9D Recon Full Hashcat 4m 25s ago
5C:5A:C7:AD:65:80 0C:EF:AF:CD:E9:9D Recon Full PCAP 4m 25s ago
5C:5A:C7:AD:65:80 0C:EF:AF:CD:EA:3C Recon Partial Hashcat 35m 18s ago

Part 3

For this question, we were able to see the access point created by personal hotspot on one of our phones:

Mike's iPhone	88:A4:79:CE:1C:B3	Unknown Vendor	0	WPA3 Mixed (PSK, SAE)	Optional	No	6		19m 45s ago	18m 28s ago

From the Recon's scanning tab, we were unable to see the users connected to this AP, nor any other access point. This is by design, as the Pineapple should not be able to see the clients associated to unsecured APs, in which case the hotspot was using WPA3 security. The "cse3140" network was also visible during our scans, and we could see that it was using WPA2 security.

We also managed to capture some handshakes for devices connecting to our vulnerable network, as shown below:

BSSID	Client	Source	Type	Captured	Message 1	Message 2
FE:E7:F2:30:B0:39	BE:DC:34:FD:EA:1F	Recon	Full PCAP	7m 14s ago	✓	✓
02:13:37:A7:FF:3C	9C:FC:E8:CF:46:E4	Recon	Partial Hashcat	6m 22s ago	✓	✓
02:13:37:A7:FF:3C	9C:FC:E8:CF:46:E4	Recon	Partial PCAP	6m 22s ago	✓	✓
02:13:37:A7:FF:90	9C:FC:E8:D3:19:F9	Recon	Partial PCAP	10m 4s ago	✓	✓
5C:5A:C7:AD:65:80	0C:EF:AF:CD:EA:3C	Recon	Partial PCAP	35m 18s ago	✓	✓
5C:5A:C7:AD:65:80	0C:EF:AF:CD:E9:9D	Recon	Full Hashcat	4m 25s ago	✓	✓
5C:5A:C7:AD:65:80	0C:EF:AF:CD:E9:9D	Recon	Full PCAP	4m 25s ago	✓	✓
5C:5A:C7:AD:65:80	0C:EF:AF:CD:EA:3C	Recon	Partial Hashcat	35m 18s ago	✓	✓

Field	**Explanation**
BSSID	The BSSID of the client (as it refers to APs) is essentially it's MAC address
Client	The client fields refers to the client whose handshake has been captured
Source	The source fields refers to the source of the capture, in this case the Recon module
Type	The type refers to the type of handshake that has been captured, and is available
Captured	The captured field refers to the time since the handshake has been captured
Message1	This boolean field refers to whether or not the Message1 from the handshake was captured
Message2	This boolean field, as with the last one, yields whether or not Message 2 was captured

Part 4

For devices connected to our vulnerable network, we were able to capture unprotected HTTP traffic to the banking website, as shown below:

```
    172.16.50.27.52569 > 172.16.48.80.80: Flags [.], cksum 0x47e6 (correct), seq 341, ack 781, win 2038, options [nop,nop,TS val 2475723359 ecr 2592657125], length 0
05:01:07.761051 IP (tos 0x2,ECT(0), ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 492)
    172.16.42.165.52568 > 172.16.48.80.80: Flags [P.], cksum 0x794d (correct), seq 730:1170, ack 1481, win 2048, options [nop,nop,TS val 2697592595 ecr 2592657115], length 440: HTTP, length: 440
        GET /static/images/staticNames/Spring_Fog.jpg HTTP/1.1
        Host: 172.16.48.80
        Connection: keep-alive
        Accept: image/webp,image/avif,video/*;q=0.8,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
        User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 16_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Mobile/15E148 Safari/604.1
        Accept-Language: en-US,en;q=0.9
        Referer: http://172.16.48.80/
        Accept-Encoding: gzip, deflate

05:01:07.761051 IP (tos 0x2,ECT(0), ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 492)
    172.16.42.165.52568 > 172.16.48.80.80: Flags [P.], cksum 0x794d (correct), seq 730:1170, ack 1481, win 2048, options [nop,nop,TS val 2697592595 ecr 2592657115], length 440: HTTP, length: 440
        GET /static/images/staticNames/Spring_Fog.jpg HTTP/1.1
        Host: 172.16.48.80
        Connection: keep-alive
        Accept: image/webp,image/avif,video/*;q=0.8,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
        User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 16_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Mobile/15E148 Safari/604.1
        Accept-Language: en-US,en;q=0.9
        Referer: http://172.16.48.80/
        Accept-Encoding: gzip, deflate

05:01:07.761325 IP (tos 0x2,ECT(0), ttl 63, id 0, offset 0, flags [DF], proto TCP (6), length 492)
    172.16.50.27.52568 > 172.16.48.80.80: Flags [P.], cksum 0x71d7 (correct), seq 730:1170, ack 1481, win 2048, options [nop,nop,TS val 2697592595 ecr 2592657115], length 440: HTTP, length: 440
        GET /static/images/staticNames/Spring_Fog.jpg HTTP/1.1
        Host: 172.16.48.80
        Connection: keep-alive
        Accept: image/webp,image/avif,video/*;q=0.8,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
        User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 16_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Mobile/15E148 Safari/604.1
        Accept-Language: en-US,en;q=0.9
        Referer: http://172.16.48.80/
        Accept-Encoding: gzip, deflate

05:01:07.761052 IP (tos 0x2,ECT(0), ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 490)
    172.16.42.165.52569 > 172.16.48.80.80: Flags [P.], cksum 0xcc5d (correct), seq 341:779, ack 781, win 2048, options [nop,nop,TS val 2475723372 ecr 2592657125], length 438: HTTP, length: 438
        GET /static/images/staticNames/husky_qa.jpg HTTP/1.1
        Host: 172.16.48.80
        Connection: keep-alive
        Accept: image/webp,image/avif,video/*;q=0.8,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5
        User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 16_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Mobile/15E148 Safari/604.1
        Accept-Language: en-US,en;q=0.9
        Referer: http://172.16.48.80/
        Accept-Encoding: gzip, deflate

05:01:07.761052 IP (tos 0x2,ECT(0), ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 490)
    172.16.42.165.52569 > 172.16.48.80.80: Flags [P.], cksum 0xcc5d (correct), seq 341:779, ack 781, win 2048, options [nop,nop,TS val 2475723372 ecr 2592657125], length 438: HTTP, length: 438
        GET /static/images/staticNames/husky_qa.jpg HTTP/1.1
        Host: 172.16.48.80
```

Part 5

Similarly, we were also able to observe unprotected HTTP traffic moving along our secure network in much the same way, as shown below:

```
    172.16.42.212.52696 > 172.16.48.80.80: Flags [.], cksum 0xd1bb (correct), seq 1170, ack 1113429, win 2048, options [nop,nop,TS val 4231389400 ecr 2592858488], length 0
05:04:29.195956 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 52)
    172.16.42.212.52696 > 172.16.48.80.80: Flags [.], cksum 0xd1bb (correct), seq 1170, ack 1113429, win 2048, options [nop,nop,TS val 4231389400 ecr 2592858488], length 0
05:04:29.196179 IP (tos 0x0, ttl 63, id 0, offset 0, flags [DF], proto TCP (6), length 52)
    172.16.50.27.52696 > 172.16.48.80.80: Flags [.], cksum 0xca74 (correct), seq 1170, ack 1113429, win 2048, options [nop,nop,TS val 4231389400 ecr 2592858488], length 0
05:04:29.267155 IP (tos 0x2,ECT(0), ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 416)
    172.16.42.212.52696 > 172.16.48.80.80: Flags [P.], cksum 0x733d (correct), seq 1170:1534, ack 1113429, win 2048, options [nop,nop,TS val 4231389472 ecr 2592858488], length 364: HTTP, length: 364
        GET /static/images/staticNames/Icon/johnathan.ico HTTP/1.1
        Host: 172.16.48.80
        Connection: keep-alive
        Accept: */*
        User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 16_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Mobile/15E148 Safari/604.1
        Accept-Language: en-US,en;q=0.9
        Referer: http://172.16.48.80/
        Accept-Encoding: gzip, deflate

05:04:29.267155 IP (tos 0x2,ECT(0), ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 416)
    172.16.42.212.52696 > 172.16.48.80.80: Flags [P.], cksum 0x733d (correct), seq 1170:1534, ack 1113429, win 2048, options [nop,nop,TS val 4231389472 ecr 2592858488], length 364: HTTP, length: 364
        GET /static/images/staticNames/Icon/johnathan.ico HTTP/1.1
        Host: 172.16.48.80
        Connection: keep-alive
        Accept: */*
        User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 16_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Mobile/15E148 Safari/604.1
        Accept-Language: en-US,en;q=0.9
        Referer: http://172.16.48.80/
        Accept-Encoding: gzip, deflate

05:04:29.267440 IP (tos 0x2,ECT(0), ttl 63, id 0, offset 0, flags [DF], proto TCP (6), length 416)
    172.16.50.27.52696 > 172.16.48.80.80: Flags [P.], cksum 0x6bf6 (correct), seq 1170:1534, ack 1113429, win 2048, options [nop,nop,TS val 4231389472 ecr 2592858488], length 364: HTTP, length: 364
        GET /static/images/staticNames/Icon/johnathan.ico HTTP/1.1
        Host: 172.16.48.80
        Connection: keep-alive
        Accept: */*
        User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 16_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Mobile/15E148 Safari/604.1
        Accept-Language: en-US,en;q=0.9
        Referer: http://172.16.48.80/
        Accept-Encoding: gzip, deflate

05:04:29.269058 IP (tos 0x0, ttl 64, id 13255, offset 0, flags [DF], proto TCP (6), length 52)
    172.16.48.80.80 > 172.16.50.27.52696: Flags [.], cksum 0xce39 (correct), seq 1113429, ack 1534, win 500, options [nop,nop,TS val 2592858635 ecr 4231389472], length 0
05:04:29.269249 IP (tos 0x0, ttl 63, id 13255, offset 0, flags [DF], proto TCP (6), length 52)
    172.16.48.80.80 > 172.16.42.212.52696: Flags [.], cksum 0xd580 (correct), seq 1113429, ack 1534, win 500, options [nop,nop,TS val 2592858635 ecr 4231389472], length 0
05:04:29.269282 IP (tos 0x0, ttl 63, id 13255, offset 0, flags [DF], proto TCP (6), length 52)
    172.16.48.80.80 > 172.16.42.212.52696: Flags [.], cksum 0xd580 (correct), seq 1113429, ack 1534, win 500, options [nop,nop,TS val 2592858635 ecr 4231389472], length 0
05:04:29.274408 IP (tos 0x2,ECT(0), ttl 64, id 13256, offset 0, flags [DF], proto TCP (6), length 1290)
    172.16.48.80.80 > 172.16.50.27.52696: Flags [.], cksum 0xcd09 (correct), seq 1113429:1114667, ack 1534, win 501, options [nop,nop,TS val 2592858635 ecr 4231389472], length 1238: HTTP, length: 1238
        HTTP/1.1 200 OK
        Date: Mon, 14 Nov 2022 12:07:22 GMT
        Server: Apache/2.4.41 (Ubuntu)
```

Notice however that the traffic looks pretty much the same as the previous capture from our vulnerable network. Principally, the security used to control who can get onto a network does not necessarily encrypt the traffic moving along that network. We can see this here, as the HTTP traffic is still visible in plaintext, and the only difference between the two captures is the IP addresses of the devices involved.

In order to be able to hide the contents of the HTTP traffic being sent over the network, we would need to use HTTPS, which is a secure version of HTTP that encrypts the traffic using SSL/TLS. This is the same protocol that is used to encrypt traffic on the internet.

Part 6

We were able to detect all wireless APs in the lab when running the scan for this part of the lab, the results are shown below:

	SSID	MAC	OUI	Clients	Security	MFP	WPS	Channel	Signal	First Seen	Last Seen
	open-altschuler-01-11	00:13:37:A7:FF:9F	Orient Power Home Network Ltd.	0	Open	No	No	11		23m 59s ago	1m 23s ago
+	UCONN-SECURE	00:27:E3:28:24:73	Cisco Systems Inc	1	WPA2 (802.1X Enterprise, 802.1X Enterprise FT)	No	No	11		23m 60s ago	1m 24s ago
	eduroam	00:27:E3:28:24:74	Cisco Systems Inc	0	WPA2 (802.1X Enterprise, 802.1X Enterprise FT)	No	No	11		25m 26s ago	1m 24s ago
+	secure-altschuler-001-10	02:13:37:A7:FF:51	Unknown Vendor	1	WPA2 (PSK)	No	No	11		25m 13s ago	1m 24s ago
	Hidden	52:2F:AD:71:82:B8	Unknown Vendor	0	WPA2 (PSK)	No	No	6		1m 34s ago	1m 34s ago
+	UCONN-SECURE	00:27:E3:9D:D4:E3	Cisco Systems Inc	1	WPA2 (802.1X Enterprise, 802.1X Enterprise FT)	No	No	1		21m 48s ago	1m 34s ago
+	hellofromBEN	5C:5A:C7:AD:65:81	Cisco Systems Inc	1	Open	No	No	6		25m 26s ago	1m 34s ago
	[illegible]	[illegible]	Orient Power Home	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]		25m 25s	[illegible]

We were also able to identify some APs that announce more than one network due to the channel they were broadcasting. For example, in the above screenshot, we can see UCONN-SECURE twice, one with channel 11, and one with channel 1. This is because the AP is broadcasting on both channels, and the Pineapple is able to detect both of them.

We can also see all of the different wireless security protocols being used by the APs in and around the lab. They were all either WPA2 (Enterprise). WPA2 (PSK), or Open. For some APs, we were able to see a client count, but it varied from 0 to 1, so we were not able to conclude how accurate this figure was.

Part 7 and Part 8

We were unable to complete these questions due to the number of active APs in the room interfering with each other.

Part 9

1. We were able to create the DNS records on the Pineapple to redirect requests to *bank.com* and *test.com* to the static IP address of our VM. This was accomplished by modifying the */etc/hosts* file as such:

```
127.0.0.1 localhost
172.16.51.49 bank.com
172.16.51.49 test.com

::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

2. We were unable to complete this part due to the Pineapple not relaying the DNS configuration to our test machines.