Wireless Rogue Access Point Detection Using Raspberry Pi Honeypot

-Sushant Yadav(184101037)

-Shiva Verma(184101034)

ABSTRACT

In the modern world as the internet is growing at a large scale there is a great chance of breaching of information by any unauthenticated user. Secrecy of the information is the primary concern of today's network system. Wireless network system are more vulnerable than the wired networks. RAP can be easily set up in the wireless network by any unauthorized user, in this paper we will demonstrate how cost effectively and easily we can detect the RAP using the Raspberry Pi Honeypot , honeypot captures the activity of the attacker and helps in improving the security of the network.

INTRODUCTION

In any organization discernment of RAP is the primary concern to worry about because it can create information breaching in the organization and can brings security threat. As we all clearly know in today's scenario for any organization the most valuable aspect of it is it's information. Using the RAP any one can access the information and create a path in the network and steal the information and misuse it. So information security will be the most difficult task for the organization, maintaining a secure network requires lots of effort and money. RAP are the access points which are deployed in the network or organization without the knowledge of security administrator to steal information from the network and access the network in illegal way. RAP can be set in network by using improperly configured AP and through unauthorized AP. First one can be configured into RAP due to some minor mistakes in configuration because network administrator have lack of knowledge of the network security techniques or improper usage of them. Latter one can be deployed in system due to flexibility and scalability of the authorized user when they set up an AP without knowledge of security administrator. These AP can act as vulnerable devices of the system and intruders can easily get access of these APs and can sniff and monitor the sensitive data.

RELATED WORK

Lots of work have been already done for discovering RAP and various techniques with their pros and cons are present. Traditional RAP works based on the tools like NetStumbler etc. that takes too much time and also not much reliable and expensive one. Further it failed if attacker spoof the MAC and Service Set Identifier (SSID) of the authorized AP.

RASPBERRY PI HONEYPOT ARCHITECTURE

Architecture comprises of three phase along with a server that helps in keeping log the activity and helps in maintaining and improving the security of the network system by providing necessary updation information to filtering and IDS phase to further secure the network.

The main objective of this network is to gather information as much as possible and using that attackers information to stop the future attacks and removing any kind of loopholes.

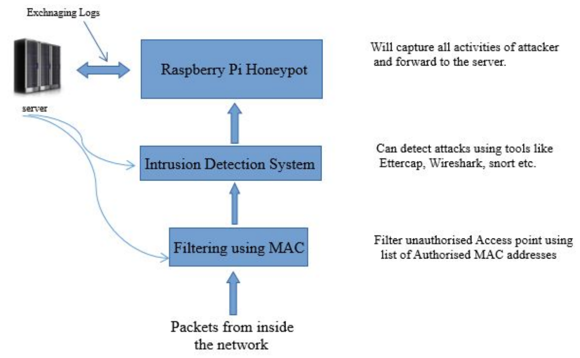

Exchnaging Logs
server
Raspberry Pi Honeypot
Intrusion Detection System
Filtering using MAC
Packets from inside the network
Will capture all activities of attacker and forward to the server.
Can detect attacks using tools like Ettercap, Wireshark, snort etc.
Filter unauthorised Access point using list of Authorised MAC addresses

fig.1: Architecture for the Approach

PROPOSED APPROACH

Since the administrator have a list of all authorized APs, can easily match the list with all the present AP and find whether some RAP is there or not but it is very easy to spoof the MAC address of some existing AP and then assigning SSID to RAP.

After passing the filtering phase it goes to the intrusion/anomaly detection phase where we consider things related to the packet and their characteristics as the administrator knows the assigned IP and MAC addresses of authorized users , can use the tools like Ettercap and wireshark to detect anomaly in the system. Ettercap check the IP and MAC address of the source of the packet and can filter out packets and catch the attacker but here also attacker can use MITM attacks like ARP spoofing ,DHCP spoofing ,DNS spoofing and can come in between authenticated user and default router.

After passing the two phases attacker will think he/she has pass the security of the organization and will search for the devices from where he can get access to sensitive data ,now as we have setted up a honeypot having IP (known to every authorized user in the network system) the attacker will try to attack at that IP address and will be caught by honeypot because any user tries to communicate with honeypot system will be considered as suspicious and all their activities will be recorded to further ensure the secrecy of the network from the errant users. Now we have many choices to do, like we can know what attacker is trying to do with the information or what are his/her intentions by just recording all the activities of the attackers after his/her intrusion in the system or we can just caught him when he tries to communicate with the honeypot and stop him there, without gathering the information regarding the reason behind the intrusion.

As there are many honeypot architecture already designed we can used any of them but keeping in mind the cost and implementation honeypot using raspberry pi will be best suited, it monitors the attacker's activity and record them which helps further to secure the system and making it more secure, raspberry tool also provide portability.

IMPLEMENTATION

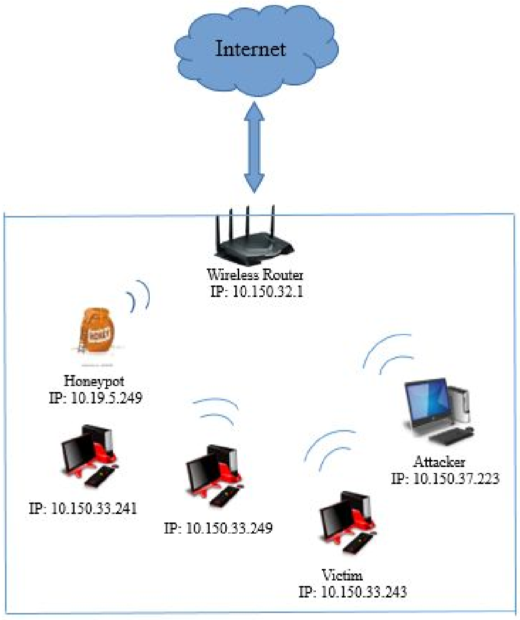


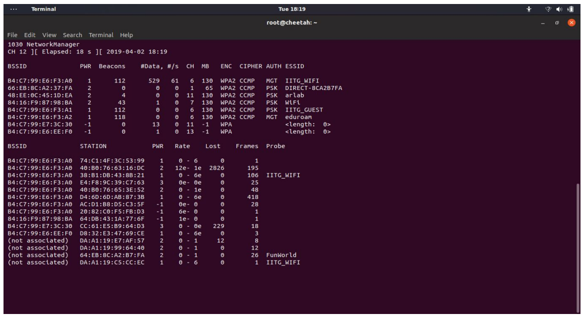
fig.2 Example network take for this paper

1.FILTERING

Network administrator can get all the MAC addresses of AP connected to that network using aircrack-ng package and administrator have a list of authorized AP's MAC Addresses. Any MAC address present in the network that doesn't in the administrator's list consider as the RAP. Administrator drop that APs.command used are :

$airmon-ng start wlp1s0

$airodump-ng prism0

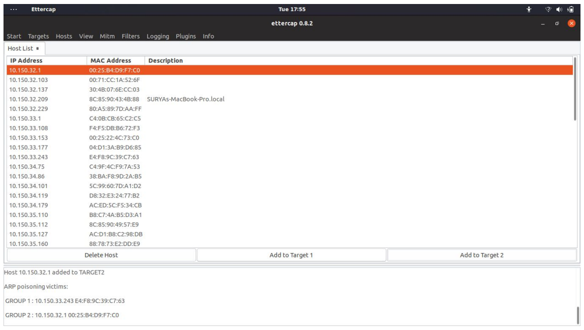


```
1030 NetworkManager
CH 12 ][ Elapsed: 18 s ][ 2019-04-02 18:19

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

B4:C7:99:E6:F3:A0    1      112      529   61   6  130  WPA2 CCMP   MGT  IITG_WIFI
66:EB:8C:A2:37:FA    2        0        0    0   1   65  WPA2 CCMP   PSK  DIRECT-8CA2B7FA
48:EE:0C:45:1D:EA    2        4        0    0  11  130  WPA2 CCMP   PSK  arlab
84:16:F9:87:98:BA    2       43        1    0   7  130  WPA2 CCMP   PSK  WiFi
B4:C7:99:E6:F3:A1    1      112        0    0   6  130  WPA2 CCMP   PSK  IITG_GUEST
B4:C7:99:E6:F3:A2    1      118        0    0   6  130  WPA2 CCMP   MGT  eduroam
B4:C7:99:E7:3C:30   -1        0       13    0  11  -1   WPA              <length:  0>
B4:C7:99:E6:EE:F0   -1        0        1    0  13  -1   WPA              <length:  0>

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

B4:C7:99:E6:F3:A0  74:C1:4F:3C:53:99    1    0 - 6      0        1
B4:C7:99:E6:F3:A0  40:B0:76:63:16:DC    2   12e- 1e  2826      195
B4:C7:99:E6:F3:A0  38:B1:DB:43:8B:21    1    0 - 6e     0      106  IITG_WIFI
B4:C7:99:E6:F3:A0  E4:F8:9C:39:C7:63    3    0e- 0e     0       25
B4:C7:99:E6:F3:A0  40:B0:76:65:3E:52    2    0 - 1e     0       48
B4:C7:99:E6:F3:A0  D4:6D:6D:AB:87:3B    1    0 - 6e     0      418
B4:C7:99:E6:F3:A0  AC:D1:B8:D5:C3:5F   -1    0e- 0      0       28
B4:C7:99:E6:F3:A0  20:82:C0:F5:FB:D3   -1    6e- 0      0        1
84:16:F9:87:98:BA  64:DB:43:1A:77:6F   -1    1e- 0      0        1
B4:C7:99:E7:3C:30  CC:61:E5:B9:64:D3    3    0 - 0e   229       18
B4:C7:99:E6:EE:F0  D8:32:E3:47:69:CE    1    0 - 6e     0        3
(not associated)   DA:A1:19:E7:AF:57    2    0 - 1     12        8
(not associated)   DA:A1:19:99:64:40    2    0 - 1      0       12
(not associated)   64:EB:8C:A2:B7:FA    2    0 - 1      0       26  FunWorld
(not associated)   DA:A1:19:C5:CC:EC    1    0 - 6      0        1  IITG_WIFI
```

fig.3 : Showing MAC addresses for all the Access Points present in the network

2.INTRUSION DETECTION

Packet moves to second phase after passing first one ,the incoming packet can be seen as either coming from authorized AP or from unauthorized AP. if it is coming from attacker , it will be easily filtered out using Ettercap , if ARP spoofing is not done .Ettercap shows the MAC and IP addresses of all AP's connected to the network as shown in fig.4

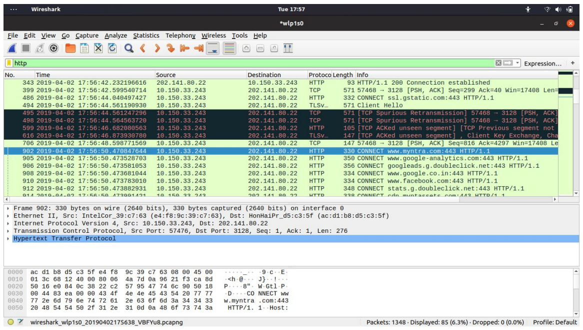


IP Address	MAC Address	Description
10.150.32.1	00:25:B4:D9:F7:C0	
10.150.32.103	00:71:CC:1A:52:6F	
10.150.32.137	30:4B:07:6E:CC:03	
10.150.32.209	8C:85:90:43:4B:88	SURYAs-MacBook-Pro.local
10.150.32.229	80:A5:89:7D:AA:FF	
10.150.33.1	C4:0B:CB:65:C2:C5	
10.150.33.108	F4:F5:DB:B6:72:F3	
10.150.33.153	00:25:22:4C:73:C0	
10.150.33.177	04:D1:3A:B9:D6:85	
10.150.33.243	E4:F8:9C:39:C7:63	
10.150.34.75	C4:9F:4C:F9:7A:53	
10.150.34.86	38:BA:F8:9D:2A:B5	
10.150.34.101	5C:99:60:7D:A1:D2	
10.150.34.119	D8:32:E3:24:77:B2	
10.150.34.179	AC:ED:5C:F5:34:CB	
10.150.35.110	B8:C7:4A:B5:D3:A1	
10.150.35.112	8C:85:90:49:57:E9	
10.150.35.127	AC:D1:B8:C2:98:DB	
10.150.35.160	88:78:73:E2:DD:E9	

```
Host 10.150.32.1 added to TARGET2

ARP poisoning victims:

 GROUP 1 : 10.150.33.243 E4:F8:9C:39:C7:63

 GROUP 2 : 10.150.32.1 00:25:B4:D9:F7:C0
```

fig.4 : showing the ARP poisioning attack using ettercap

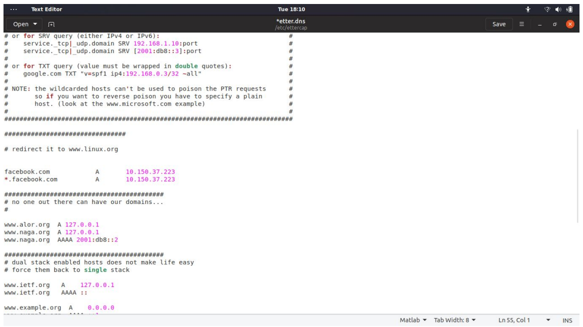
Using ettercap tool attacker can perform ARP poisioning attack and can keep an eye on the log of the authorized host and will get the sensitive information from the host and can perform other attack using that info. A scenario of the attackers sniffing shown below using the Wireshark tool.These attacks can be catch by using packet spacing technique and has already been developed thats why not covered in this paper.


Wireshark
Tue 17:57
*wlp1s0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
Expression…
+

```
No.  Time                           Source          Destination     Protoco Length Info
 343 2019-04-02 17:56:42.232196616  202.141.80.22   10.150.33.243   HTTP     93 HTTP/1.1 200 Connection established
 399 2019-04-02 17:56:42.599540714  10.150.33.243   202.141.80.22   TCP     571 57468 → 3128 [PSH, ACK] Seq=299 Ack=40 Win=17408 Len=
 486 2019-04-02 17:56:44.040497427  10.150.33.243   202.141.80.22   HTTP    332 CONNECT ssl.gstatic.com:443 HTTP/1.1
 494 2019-04-02 17:56:44.561190930  10.150.33.243   202.141.80.22   TLSv…   571 Client Hello
 495 2019-04-02 17:56:44.561247296  10.150.33.243   202.141.80.22   TCP     571 [TCP Spurious Retransmission] 57468 → 3128 [PSH, ACK]
 498 2019-04-02 17:56:44.564563720  10.150.33.243   202.141.80.22   TCP     571 [TCP Spurious Retransmission] 57468 → 3128 [PSH, ACK]
 599 2019-04-02 17:56:46.682080563  10.150.33.243   202.141.80.22   HTTP    105 [TCP ACKed unseen segment] [TCP Previous segment not
 616 2019-04-02 17:56:46.873930780  10.150.33.243   202.141.80.22   TLSv…   147 [TCP ACKed unseen segment] , Client Key Exchange, Cha
 706 2019-04-02 17:56:48.598771569  10.150.33.243   202.141.80.22   TCP     147 57468 → 3128 [PSH, ACK] Seq=816 Ack=4297 Win=17408 Le
 902 2019-04-02 17:56:50.470847644  10.150.33.243   202.141.80.22   HTTP    330 CONNECT www.myntra.com:443 HTTP/1.1
 905 2019-04-02 17:56:50.473528703  10.150.33.243   202.141.80.22   HTTP    350 CONNECT www.google-analytics.com:443 HTTP/1.1
 906 2019-04-02 17:56:50.473581053  10.150.33.243   202.141.80.22   HTTP    356 CONNECT googleads.g.doubleclick.net:443 HTTP/1.1
 908 2019-04-02 17:56:50.473681044  10.150.33.243   202.141.80.22   HTTP    334 CONNECT www.google.co.in:443 HTTP/1.1
 910 2019-04-02 17:56:50.473783010  10.150.33.243   202.141.80.22   HTTP    334 CONNECT www.facebook.com:443 HTTP/1.1
 912 2019-04-02 17:56:50.473882931  10.150.33.243   202.141.80.22   HTTP    348 CONNECT stats.g.doubleclick.net:443 HTTP/1.1
 914 2019-04-02 17:56:50.473991421  10.150.33.243   202.141.80.22   HTTP    338 CONNECT cdn.myntassets.com:443 HTTP/1.1

› Frame 902: 330 bytes on wire (2640 bits), 330 bytes captured (2640 bits) on interface 0
› Ethernet II, Src: IntelCor_39:c7:63 (e4:f8:9c:39:c7:63), Dst: HonHaiPr_d5:c3:5f (ac:d1:b8:d5:c3:5f)
› Internet Protocol Version 4, Src: 10.150.33.243, Dst: 202.141.80.22
› Transmission Control Protocol, Src Port: 57476, Dst Port: 3128, Seq: 1, Ack: 1, Len: 276
› Hypertext Transfer Protocol

0000  ac d1 b8 d5 c3 5f e4 f8  9c 39 c7 63 08 00 45 00   ·····_·· ·9·c··E·
0010  01 3c 68 12 40 00 80 06  4a 7d 0a 96 21 f3 ca 8d   ·<h·@··· J}··!···
0020  50 16 e0 84 0c 38 22 c2  57 95 47 74 6c 90 50 18   P····8"· W·Gtl·P·
0030  00 44 83 ea 00 00 43 4f  4e 4e 45 43 54 20 77 77   ·D····CO NNECT ww
0040  77 2e 6d 79 6e 74 72 61  2e 63 6f 6d 3a 34 34 33   w.myntra .com:443
0050  20 48 54 54 50 2f 31 2e  31 0d 0a 48 6f 73 74 3a    HTTP/1. 1··Host:

wireshark_wlp1s0_20190402175638_VBFYu8.pcapng    Packets: 1348 · Displayed: 85 (6.3%) · Dropped: 0 (0.0%)    Profile: Default
```

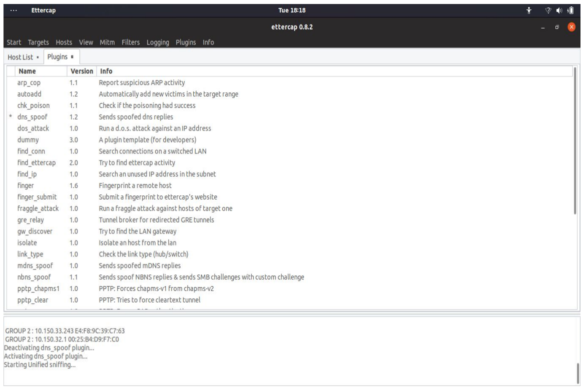
fig.5 : showing that attacker can see the websites that user is viewing (Man in the Middle) using wireshark.

Attacker can also perform DNS spoofing attack by using ettercap tool , first it will change the etter.dns file shown in fig.6 and provide its own website ip address for the website he/she want to spoof like in our case we have spoofed facebook.com to ip address 10.50.37.223.



```
# or for SRV query (either IPv4 or IPv6):                             #
#    service._tcp|_udp.domain SRV 192.168.1.10:port                   #
#    service._tcp|_udp.domain SRV [2001:db8::3]:port                  #
#                                                                     #
# or for TXT query (value must be wrapped in double quotes):          #
#    google.com TXT "v=spf1 ip4:192.168.0.3/32 ~all"                  #
#                                                                     #
# NOTE: the wildcarded hosts can't be used to poison the PTR requests #
#       so if you want to reverse poison you have to specify a plain  #
#       host. (look at the www.microsoft.com example)                 #
#                                                                     #
#######################################################################

################################

# redirect it to www.linux.org


facebook.com            A       10.150.37.223
*.facebook.com          A       10.150.37.223

##########################################
# no one out there can have our domains...
#

www.alor.org  A 127.0.0.1
www.naga.org  A 127.0.0.1
www.naga.org  AAAA 2001:db8::2

##########################################
# dual stack enabled hosts does not make life easy
# force them back to single stack

www.ietf.org   A    127.0.0.1
www.ietf.org   AAAA ::

www.example.org  A    0.0.0.0
```

fig.6 : showing the content of the file etter.dns after changing

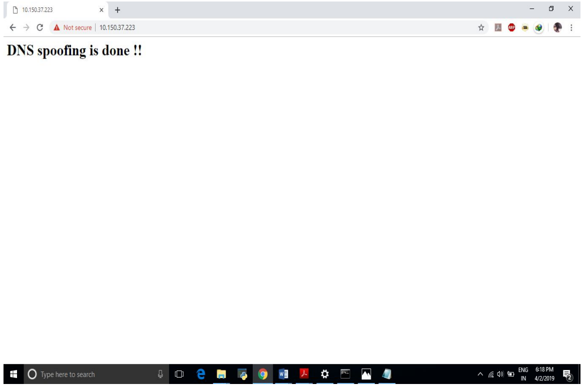


	Name	Version	Info
	arp_cop	1.1	Report suspicious ARP activity
	autoadd	1.2	Automatically add new victims in the target range
	chk_poison	1.1	Check if the poisoning had success
*	dns_spoof	1.2	Sends spoofed dns replies
	dos_attack	1.0	Run a d.o.s. attack against an IP address
	dummy	3.0	A plugin template (for developers)
	find_conn	1.0	Search connections on a switched LAN
	find_ettercap	2.0	Try to find ettercap activity
	find_ip	1.0	Search an unused IP address in the subnet
	finger	1.6	Fingerprint a remote host
	finger_submit	1.0	Submit a fingerprint to ettercap's website
	fraggle_attack	1.0	Run a fraggle attack against hosts of target one
	gre_relay	1.0	Tunnel broker for redirected GRE tunnels
	gw_discover	1.0	Try to find the LAN gateway
	isolate	1.0	Isolate an host from the lan
	link_type	1.0	Check the link type (hub/switch)
	mdns_spoof	1.0	Sends spoofed mDNS replies
	nbns_spoof	1.1	Sends spoof NBNS replies & sends SMB challenges with custom challenge
	pptp_chapms1	1.0	PPTP: Forces chapms-v1 from chapms-v2
	pptp_clear	1.0	PPTP: Tries to force cleartext tunnel

```
GROUP 2 : 10.150.33.243 E4:F8:9C:39:C7:63
GROUP 2 : 10.150.32.1 00:25:B4:D9:F7:C0
Deactivating dns_spoof plugin...
Activating dns_spoof plugin...
Starting Unified sniffing...
```

fig.7: show the ettercap plugins (dns_spoof is set)

And then set the plugins to be dns_spoof shown in fig.7 and then start sniffing .Now fig.8 shown that the pst scene of this attack on the victim's machine.

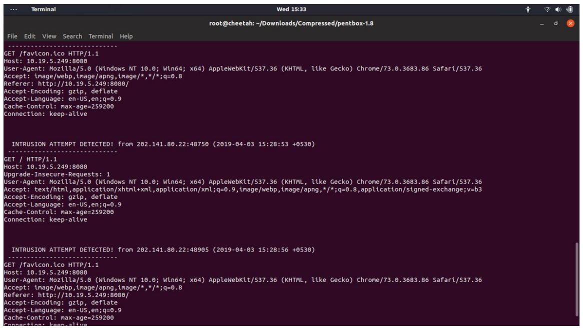


DNS spoofing is done !!

fig.8: victim is being spoofed to attackers ip

3.Raspberry Pi HoneyPot

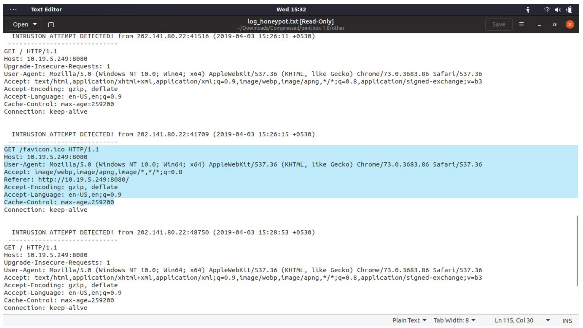
Honeypot is created using pentbox tool ,pentbox is a low interaction Honeypot system, it catch the attacker and displays the attackers log. It provide the information to which browser through attacker is accessing and all the information such as time, encoding and which OS is used by attacker. Log file maintained by the honeypot system is sended to the server that contains all the info of the attackers activity, that helps in future prospect to secure the system from further attacks and also in improving the security of the network.


Terminal
Wed 15:33
root@cheetah: ~/Downloads/Compressed/pentbox-1.8
File Edit View Search Terminal Help

```
 ----------------------------
GET /favicon.ico HTTP/1.1
Host: 10.19.5.249:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: http://10.19.5.249:8080/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cache-Control: max-age=259200
Connection: keep-alive



  INTRUSION ATTEMPT DETECTED! from 202.141.80.22:48750 (2019-04-03 15:28:53 +0530)
 ----------------------------
GET / HTTP/1.1
Host: 10.19.5.249:8080
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cache-Control: max-age=259200
Connection: keep-alive



  INTRUSION ATTEMPT DETECTED! from 202.141.80.22:48905 (2019-04-03 15:28:56 +0530)
 ----------------------------
GET /favicon.ico HTTP/1.1
Host: 10.19.5.249:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: http://10.19.5.249:8080/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cache-Control: max-age=259200
Connection: keep-alive
```

fig.9 : pentbox honeypot deployment and catching of intruder

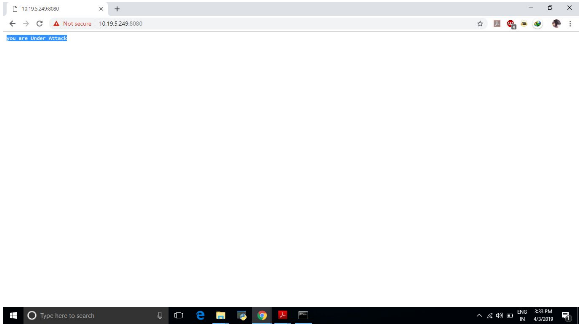
Intruder's all activities will be captured in log file that will be helpful to avoid future attack and relative updates will be done in lower phases to overall make the network resistant to any kind of intrusion.


Text Editor
Wed 15:32
Open
log_honeypot.txt [Read-Only]
~/Downloads/Compressed/pentbox-1.8/other
Save
Plain Text
Tab Width: 8
Ln 115, Col 30
INS

```
  INTRUSION ATTEMPT DETECTED! from 202.141.80.22:41516 (2019-04-03 15:26:11 +0530)
 ----------------------------
GET / HTTP/1.1
Host: 10.19.5.249:8080
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cache-Control: max-age=259200
Connection: keep-alive


  INTRUSION ATTEMPT DETECTED! from 202.141.80.22:41709 (2019-04-03 15:26:15 +0530)
 ----------------------------
GET /favicon.ico HTTP/1.1
Host: 10.19.5.249:8080
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: http://10.19.5.249:8080/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cache-Control: max-age=259200
Connection: keep-alive


  INTRUSION ATTEMPT DETECTED! from 202.141.80.22:48750 (2019-04-03 15:28:53 +0530)
 ----------------------------
GET / HTTP/1.1
Host: 10.19.5.249:8080
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/73.0.3683.86 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Cache-Control: max-age=259200
Connection: keep-alive
```

fig.10: log file containing information of intruder

fig.11 shows the screen of attacker after it is caught by the Honeypot .



you are Under Attack

fig.11: shows the intruder has been catched

CONCLUSION

In recent years wireless devices and wireless networks usage increased at a large scale so they require generally wireless connections to communicate, so for authenticated data exchange among the authorized user secrecy of the network should be high and mechanisms to detect and prevent the configuration of RAP should also be there. This paper demonstrates the raspberry pi honeypot architecture, which is cost effective and easy to implement compare to other techniques, this also provide the facility to further enhance the security of the network and captures the attackers and their activities.It strengthen the overall performance of the network by discovering the more attacks in the second phase and minimize the loop holes, secure the network architecture.

REFRENCE

1. Surendra Mahajan, Akshay Mhasku Adagale,Chetna Sahare, (2016),Intrusion Detection System Using Raspberry PI Honeypot in Network Security , ijesc

2. Neha Agrawal,Shashikala Tapaswi(2015), Wireless Rogue Access Point Detection Using Shadow Honeynet,springer

3. P. Diebold, A. Hess, G. Schäfer(2005) , A Honeypot Architecture for Detecting and Analyzing Unknown Network Attacks