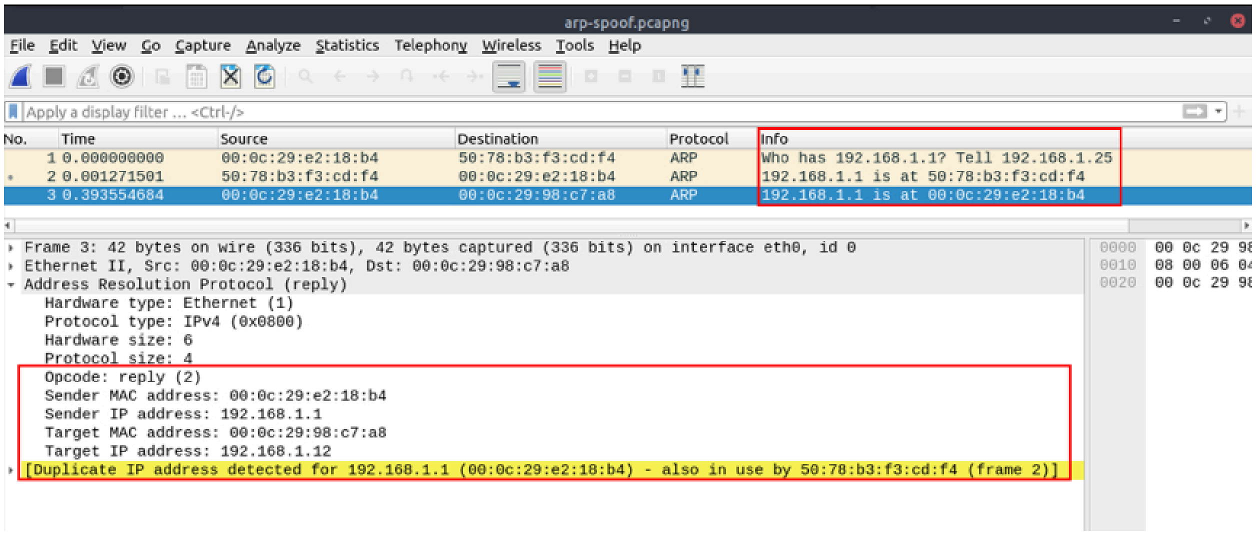
Lucas McMahon
7/29/24

Detecting ARP Spoofing

Today I am going to learn how to detect ARP Spoofing through Wireshark's packet analysis features. I will be using TryHackMe's learning room to walk through questions and answers. My goal is to be able to teach someone how to detect this attack by reading this writeup as well as enhance my own learning.

ARP poisoning is a type of network manipulation by sending malicious ARP packets to the default gateway. The goal is to alter the IP to MAC table and sniff traffic. It can prove to be a powerful tool in an attacker's arsenal but can also be easily detected given the right knowledge.

Suspicion should occur when two MAC addresses claim the same IP address. Luckily Wireshark provides a notice for when this occurs, here's what it looks like.


arp-spoof.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

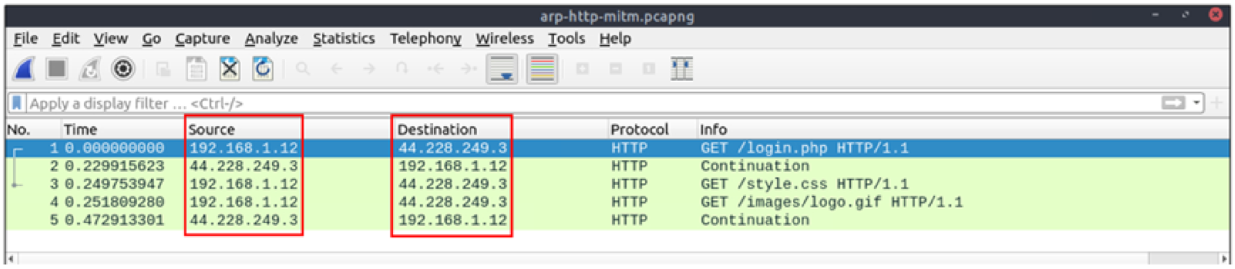
```
No.  Time         Source             Destination        Protocol  Info
  1 0.000000000   00:0c:29:e2:18:b4  50:78:b3:f3:cd:f4  ARP       Who has 192.168.1.1? Tell 192.168.1.25
  2 0.001271501   50:78:b3:f3:cd:f4  00:0c:29:e2:18:b4  ARP       192.168.1.1 is at 50:78:b3:f3:cd:f4
  3 0.393554684   00:0c:29:e2:18:b4  00:0c:29:98:c7:a8  ARP       192.168.1.1 is at 00:0c:29:e2:18:b4
```

```
▸ Frame 3: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface eth0, id 0
▸ Ethernet II, Src: 00:0c:29:e2:18:b4, Dst: 00:0c:29:98:c7:a8
▾ Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: 00:0c:29:e2:18:b4
    Sender IP address: 192.168.1.1
    Target MAC address: 00:0c:29:98:c7:a8
    Target IP address: 192.168.1.12
▸ [Duplicate IP address detected for 192.168.1.1 (00:0c:29:e2:18:b4) - also in use by 50:78:b3:f3:cd:f4 (frame 2)]
```

```
0000  00 0c 29 9
0010  08 00 06 0
0020  00 0c 29 9
```

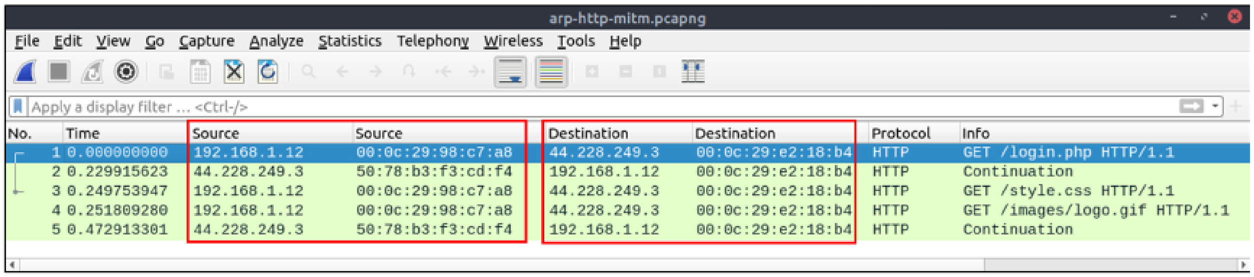
After seeing this we can still not be totally clear an attack is occuring, misconfigurations do occur. But just in case, it is best practice to look further. The first consideration should be finding out which one of the MAC addresses could be fraudulent. There is one ending in "b4" and another in "f4".

To continue the investigation, I'm going to change over to view the HTTP traffic file.


arp-http-mitm.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Info
1	0.000000000	192.168.1.12	44.228.249.3	HTTP	GET /login.php HTTP/1.1
2	0.229915623	44.228.249.3	192.168.1.12	HTTP	Continuation
3	0.249753947	192.168.1.12	44.228.249.3	HTTP	GET /style.css HTTP/1.1
4	0.251809280	192.168.1.12	44.228.249.3	HTTP	GET /images/logo.gif HTTP/1.1
5	0.472913301	44.228.249.3	192.168.1.12	HTTP	Continuation

At a first glance, it looks like normal HTTP traffic. By right clicking on any of the columns, we can go to preferences and add a column to view the MAC addresses associated with the traffic.


arp-http-mitm.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

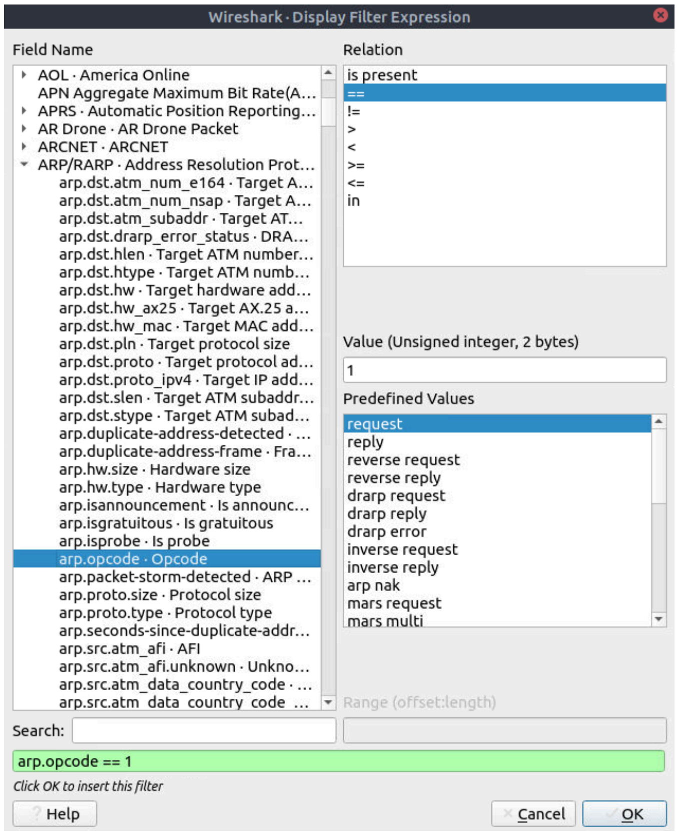
No.	Time	Source	Source	Destination	Destination	Protocol	Info
1	0.000000000	192.168.1.12	00:0c:29:98:c7:a8	44.228.249.3	00:0c:29:e2:18:b4	HTTP	GET /login.php HTTP/1.1
2	0.229915623	44.228.249.3	50:78:b3:f3:cd:f4	192.168.1.12	00:0c:29:e2:18:b4	HTTP	Continuation
3	0.249753947	192.168.1.12	00:0c:29:98:c7:a8	44.228.249.3	00:0c:29:e2:18:b4	HTTP	GET /style.css HTTP/1.1
4	0.251809280	192.168.1.12	00:0c:29:98:c7:a8	44.228.249.3	00:0c:29:e2:18:b4	HTTP	GET /images/logo.gif HTTP/1.1
5	0.472913301	44.228.249.3	50:78:b3:f3:cd:f4	192.168.1.12	00:0c:29:e2:18:b4	HTTP	Continuation

A clear anomaly! All the traffic associated with multiple different IP addresses is all being sent the same MAC address ending in “b4”, the one we had suspicion for earlier. This is an example of ARP spoofing or a “man in the middle” attack. Someone is spoofing IP addresses to intercept traffic.

From a defensive standpoint, it would be up to another platform to defend against this attack, Wireshark is only used for analysis. So i’m going to continue onto more examples to detect anomalies like this.

The first question asks how many ARP packets the attacker sent. Now that we know the attackers MAC address, all that's needed to be done is alter the display filters so we can find what’s needed.

Wireshark has a function that provides expressions needed to filter traffic, which can be found in the analyze tab at the top of the window. After scrolling through the tab, we can find the ARP section and get the expressions to filter for ARP requests.



Wireshark · Display Filter Expression

Field Name

```
AOL · America Online
APN Aggregate Maximum Bit Rate(A…
APRS · Automatic Position Reporting…
AR Drone · AR Drone Packet
ARCNET · ARCNET
ARP/RARP · Address Resolution Prot…
    arp.dst.atm_num_e164 · Target A…
    arp.dst.atm_num_nsap · Target A…
    arp.dst.atm_subaddr · Target AT…
    arp.dst.drarp_error_status · DRA…
    arp.dst.hlen · Target ATM number…
    arp.dst.htype · Target ATM numb…
    arp.dst.hw · Target hardware add…
    arp.dst.hw_ax25 · Target AX.25 a…
    arp.dst.hw_mac · Target MAC add…
    arp.dst.pln · Target protocol size
    arp.dst.proto · Target protocol ad…
    arp.dst.proto_ipv4 · Target IP add…
    arp.dst.slen · Target ATM subaddr…
    arp.dst.stype · Target ATM subad…
    arp.duplicate-address-detected · …
    arp.duplicate-address-frame · Fra…
    arp.hw.size · Hardware size
    arp.hw.type · Hardware type
    arp.isannouncement · Is announc…
    arp.isgratuitous · Is gratuitous
    arp.isprobe · Is probe
    arp.opcode · Opcode
    arp.packet-storm-detected · ARP …
    arp.proto.size · Protocol size
    arp.proto.type · Protocol type
    arp.seconds-since-duplicate-addr…
    arp.src.atm_afi · AFI
    arp.src.atm_afi.unknown · Unkno…
    arp.src.atm_data_country_code · …
    arp.src.atm_data_country_code …
```

Relation

```
is present
==
!=
>
<
>=
<=
in
```

Value (Unsigned integer, 2 bytes)

```
1
```

Predefined Values

```
request
reply
reverse request
reverse reply
drarp request
drarp reply
drarp error
inverse request
inverse reply
arp nak
mars request
mars multi
```

Range (offset:length)

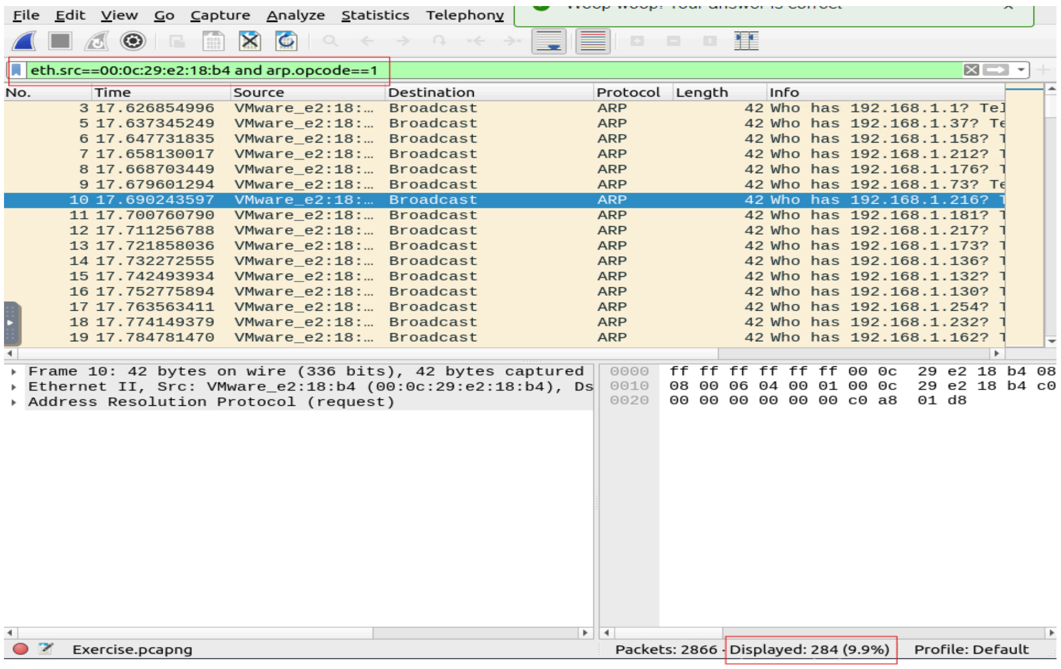
Search:

```
arp.opcode == 1
```

Click OK to insert this filter

Help Cancel OK

Using that and the expression to filter ethernet traffic, “eth.src== “, we can find the answer to our questions.



File Edit View Go Capture Analyze Statistics Telephony

```
eth.src==00:0c:29:e2:18:b4 and arp.opcode==1
```

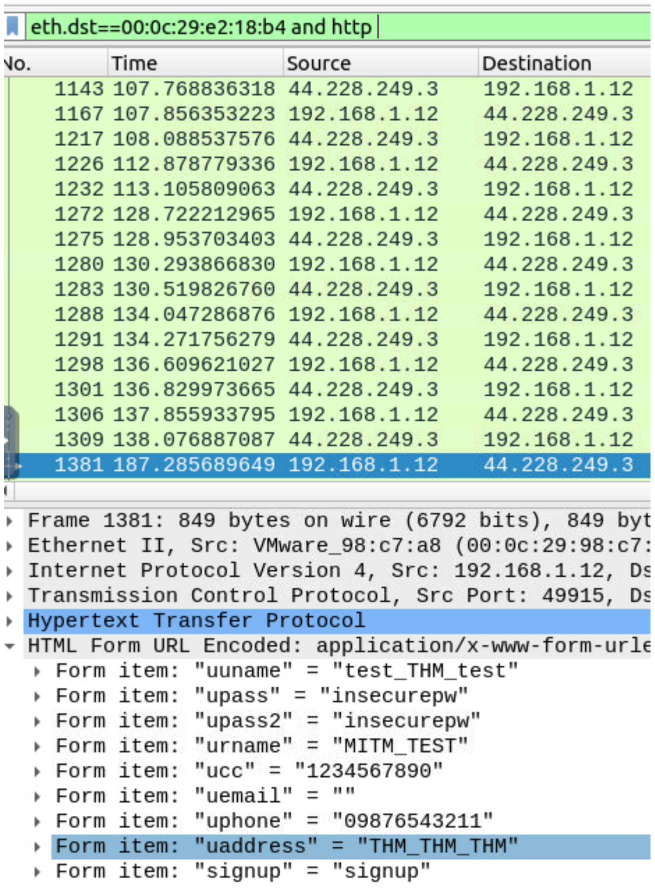
```
No.   Time          Source          Destination   Protocol  Length  Info
    3 17.626854996  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.1? Tel
    5 17.637345249  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.37? Te
    6 17.647731835  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.158? T
    7 17.658130017  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.212? T
    8 17.668703449  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.176? T
    9 17.679601294  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.73? Te
   10 17.690243597  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.216? T
   11 17.700760790  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.181? T
   12 17.711256788  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.217? T
   13 17.721858036  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.173? T
   14 17.732272555  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.136? T
   15 17.742493934  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.132? T
   16 17.752775894  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.130? T
   17 17.763563411  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.254? T
   18 17.774149379  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.232? T
   19 17.784781470  VMware_e2:18:…  Broadcast     ARP           42 Who has 192.168.1.162? T
```

```
▸ Frame 10: 42 bytes on wire (336 bits), 42 bytes captured
▸ Ethernet II, Src: VMware_e2:18:b4 (00:0c:29:e2:18:b4), Ds
▸ Address Resolution Protocol (request)
```

```
0000  ff ff ff ff ff ff 00 0c  29 e2 18 b4 08
0010  08 00 06 04 00 01 00 0c  29 e2 18 b4 c0
0020  00 00 00 00 00 00 c0 a8  01 d8
```

```
Exercise.pcapng     Packets: 2866 · Displayed: 284 (9.9%)     Profile: Default
```

The last question asks “How many usernames and passwords were sniffed by the attacker?” To find this, I'm going to start by filtering all the HTTP traffic sent to the attacker. I manually looked through the traffic messages to find an instance of a username and password being communicated.

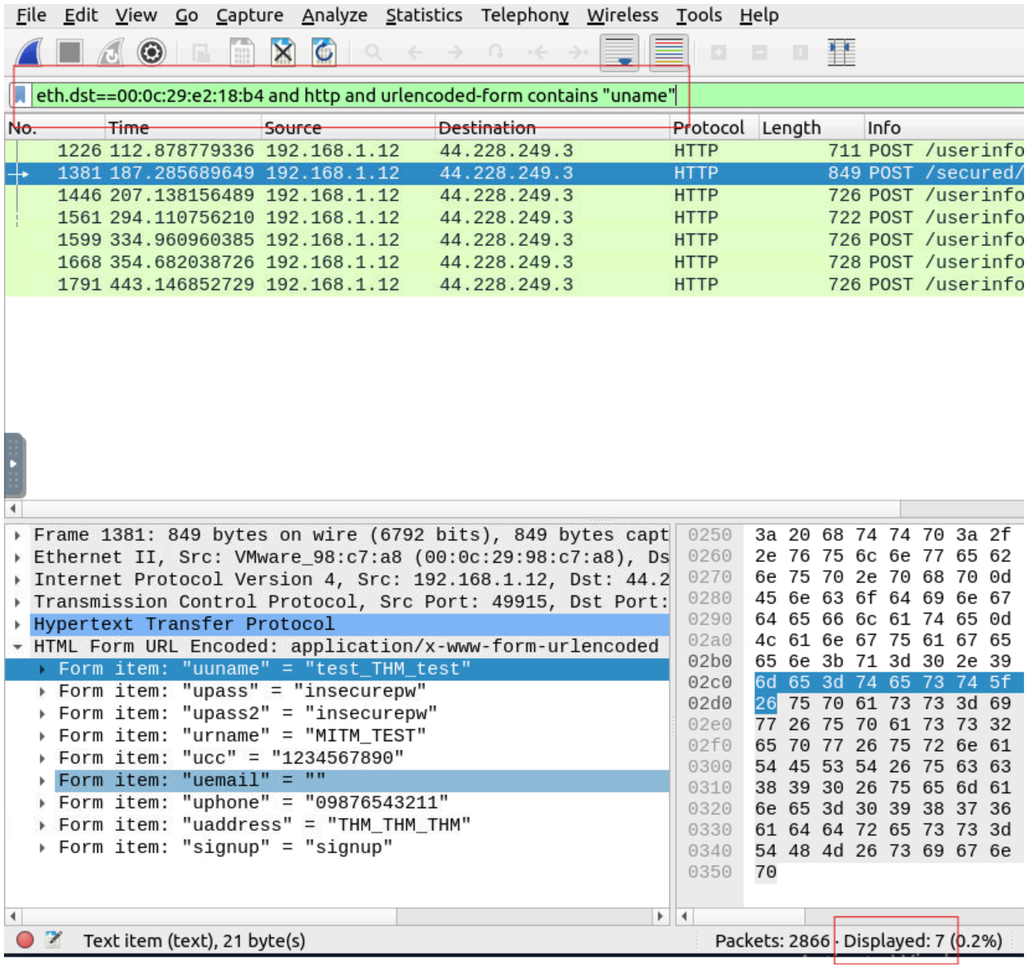


```
eth.dst==00:0c:29:e2:18:b4 and http

No.  Time          Source         Destination
1143 107.768836318 44.228.249.3   192.168.1.12
1167 107.856353223 192.168.1.12   44.228.249.3
1217 108.088537576 44.228.249.3   192.168.1.12
1226 112.878779336 192.168.1.12   44.228.249.3
1232 113.105809063 44.228.249.3   192.168.1.12
1272 128.722212965 192.168.1.12   44.228.249.3
1275 128.953703403 44.228.249.3   192.168.1.12
1280 130.293866830 192.168.1.12   44.228.249.3
1283 130.519826760 44.228.249.3   192.168.1.12
1288 134.047286876 192.168.1.12   44.228.249.3
1291 134.271756279 44.228.249.3   192.168.1.12
1298 136.609621027 192.168.1.12   44.228.249.3
1301 136.829973665 44.228.249.3   192.168.1.12
1306 137.855933795 192.168.1.12   44.228.249.3
1309 138.076887087 44.228.249.3   192.168.1.12
1381 187.285689649 192.168.1.12   44.228.249.3

Frame 1381: 849 bytes on wire (6792 bits), 849 byt
Ethernet II, Src: VMware_98:c7:a8 (00:0c:29:98:c7:
Internet Protocol Version 4, Src: 192.168.1.12, Ds
Transmission Control Protocol, Src Port: 49915, Ds
Hypertext Transfer Protocol
HTML Form URL Encoded: application/x-www-form-urle
  Form item: "uuname" = "test_THM_test"
  Form item: "upass" = "insecurepw"
  Form item: "upass2" = "insecurepw"
  Form item: "urname" = "MITM_TEST"
  Form item: "ucc" = "1234567890"
  Form item: "uemail" = ""
  Form item: "uphone" = "09876543211"
  Form item: "uaddress" = "THM_THM_THM"
  Form item: "signup" = "signup"
```

After that, I found what I was looking for, I added the filter "*urlencoded-form contains "uname"*" along to what I already had and got my answer.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
eth.dst==00:0c:29:e2:18:b4 and http and urlencoded-form contains "uname"
```

```
No.   Time           Source         Destination    Protocol Length Info
 1226 112.878779336 192.168.1.12   44.228.249.3   HTTP        711 POST /userinfo
 1381 187.285689649 192.168.1.12   44.228.249.3   HTTP        849 POST /secured/
 1446 207.138156489 192.168.1.12   44.228.249.3   HTTP        726 POST /userinfo
 1561 294.110756210 192.168.1.12   44.228.249.3   HTTP        722 POST /userinfo
 1599 334.960960385 192.168.1.12   44.228.249.3   HTTP        726 POST /userinfo
 1668 354.682038726 192.168.1.12   44.228.249.3   HTTP        728 POST /userinfo
 1791 443.146852729 192.168.1.12   44.228.249.3   HTTP        726 POST /userinfo
```

```
Frame 1381: 849 bytes on wire (6792 bits), 849 bytes capt
Ethernet II, Src: VMware_98:c7:a8 (00:0c:29:98:c7:a8), Ds
Internet Protocol Version 4, Src: 192.168.1.12, Dst: 44.2
Transmission Control Protocol, Src Port: 49915, Dst Port:
Hypertext Transfer Protocol
HTML Form URL Encoded: application/x-www-form-urlencoded
  Form item: "uuname" = "test_THM_test"
  Form item: "upass" = "insecurepw"
  Form item: "upass2" = "insecurepw"
  Form item: "urname" = "MITM_TEST"
  Form item: "ucc" = "1234567890"
  Form item: "uemail" = ""
  Form item: "uphone" = "09876543211"
  Form item: "uaddress" = "THM_THM_THM"
  Form item: "signup" = "signup"
```

```
0250  3a 20 68 74 74 70 3a 2f
0260  2e 76 75 6c 6e 77 65 62
0270  6e 75 70 2e 70 68 70 0d
0280  45 6e 63 6f 64 69 6e 67
0290  64 65 66 6c 61 74 65 0d
02a0  4c 61 6e 67 75 61 67 65
02b0  65 6e 3b 71 3d 30 2e 39
02c0  6d 65 3d 74 65 73 74 5f
02d0  26 75 70 61 73 73 3d 69
02e0  77 26 75 70 61 73 73 32
02f0  65 70 77 26 75 72 6e 61
0300  54 45 53 54 26 75 63 63
0310  38 39 30 26 75 65 6d 61
0320  6e 65 3d 30 39 38 37 36
0330  61 64 64 72 65 73 73 3d
0340  54 48 4d 26 73 69 67 6e
0350  70
```

Text item (text), 21 byte(s)   Packets: 2866 · Displayed: 7 (0.2%)