MARIST

School of Computer Science and Mathematics

Covert Message Channels and Clock Spoof DoS Attacks on IEEE Precision Time Protocol (PTPv2) with Timemaster

Casimer DeCusatis, Ph.D. & IBM D.E. Emeritus and Luke Jacobs, Marist College
casimer.decusatis@marist.edu

Paul Wojciak, D.E., Clay Kaiser, and Steve Guendert, IBM
wojciak@us.ibm.com

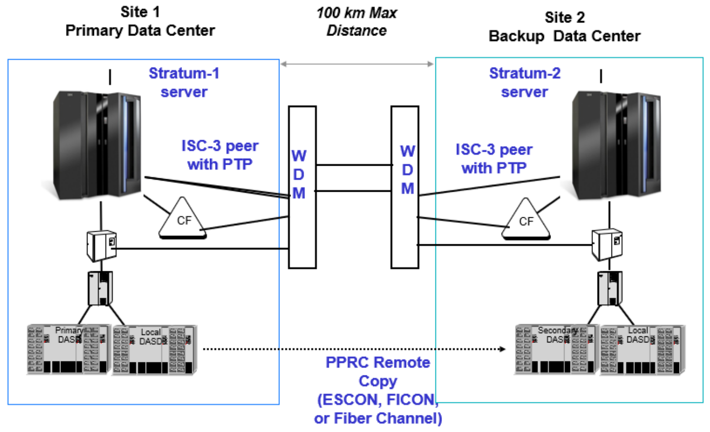
IBM

WSTS 2022, Denver, CO May 2022

Overview – What is PTP?

- The IEEE 1588 standard Precision Time Protocol standard (PTP) is a follow-on to the well known Network Time Protocol (NTP) which provides highly accurate (nanosecond or better) synchronized data center clock signals.

- Cyberattacks which destroy clock synchronization have devastating consequences.
  - Does not preserve order of transactions; critical issue for IBM Z Systems and Next Generation GDPS
  - Impacts event scheduling (backup/recovery with incorrect timestamps) including recovery time point/objective, causality violation
  - Induce time skips, temporal vortex, or complete loss of clock synchronization to all clients



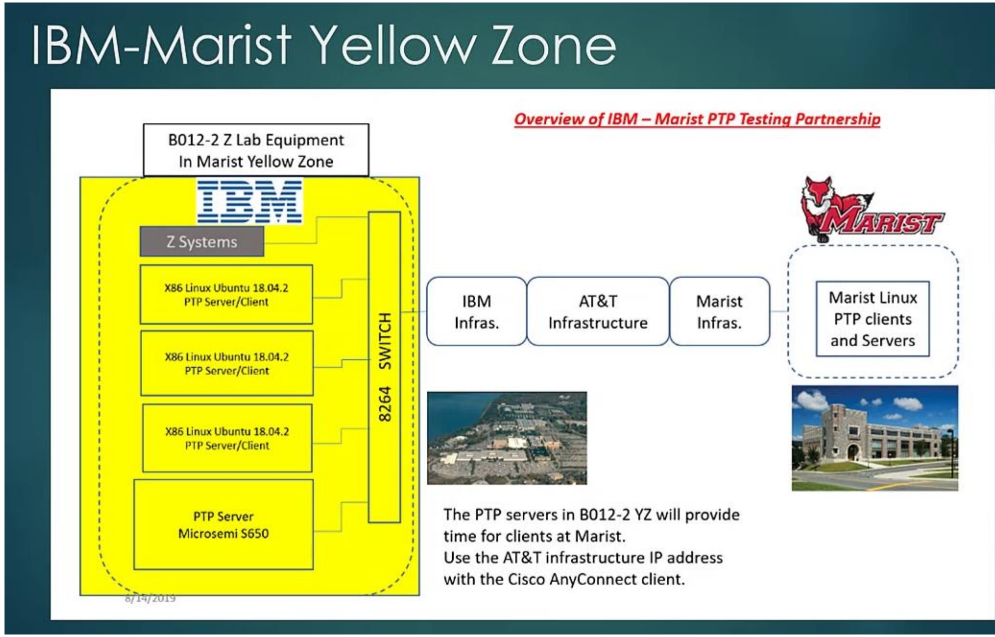
MARIST

School of Computer Science and Mathematics

The IBM-Marist Joint Study Program

- We built an experimental test bed through the IBM-Marist Joint Study. Results include: security vulnerabilities in PTP, and propose mitigation techniques for several attacks:
  - 1. DoS Attack using Announce packets
  - 2. Source Spoof Attack (masquerade attack impersonating system source clock)
  - 3. Atomic Source Takeover (spoof the PTP process with a fake atomic clock)
  - 4. Covert Channel MITM Attack
  - 5. Clock Frequency Manipulation Attack

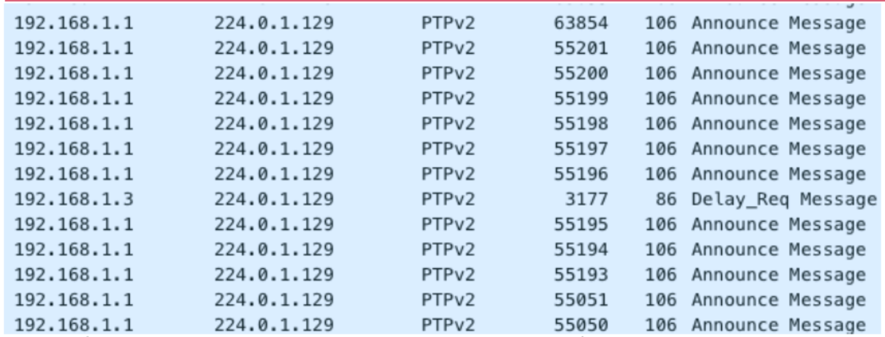
  - PTP Covert Channel for data exfiltration


IBM-Marist Yellow Zone
Overview of IBM – Marist PTP Testing Partnership
B012-2 Z Lab Equipment In Marist Yellow Zone
IBM
Z Systems
X86 Linux Ubuntu 18.04.2 PTP Server/Client
X86 Linux Ubuntu 18.04.2 PTP Server/Client
X86 Linux Ubuntu 18.04.2 PTP Server/Client
PTP Server Microsemi S650
8264 SWITCH
IBM Infras.
AT&T Infrastructure
Marist Infras.
Marist Linux PTP clients and Servers
MARIST
8/14/2019

The PTP servers in B012-2 YZ will provide time for clients at Marist.
Use the AT&T infrastructure IP address with the Cisco AnyConnect client.

1. Announce DoS – spam announce packets at the follower

Announce DoS



```
192.168.1.1     224.0.1.129     PTPv2     63854   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55201   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55200   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55199   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55198   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55197   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55196   106  Announce Message
192.168.1.3     224.0.1.129     PTPv2      3177    86  Delay_Req Message
192.168.1.1     224.0.1.129     PTPv2     55195   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55194   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55193   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55051   106  Announce Message
192.168.1.1     224.0.1.129     PTPv2     55050   106  Announce Message
```

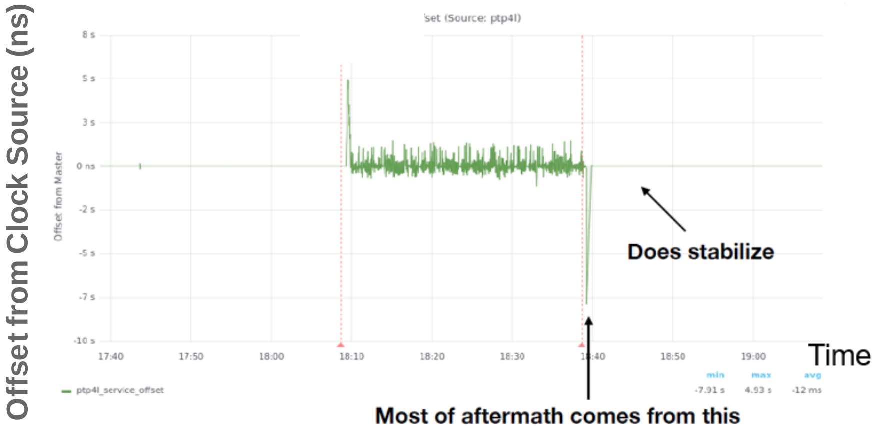
Spoofed IP

“Valid” Sequence IDs

No need to spoof sequence IDs

200-300 spam packets/second

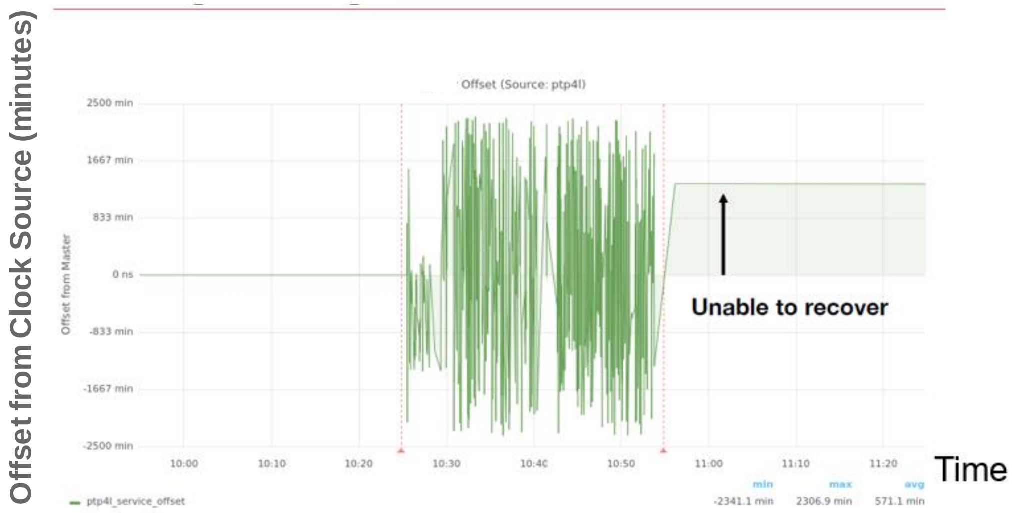
Announce DoS - Graph



Average Offset During Attack: 137.8 ms

Average Offset After Attack: -86.1 ms

2. Source Spoof – pretend to be the main clock source and send false data to the followers



30 minute attack can push the clock days or years out of sync

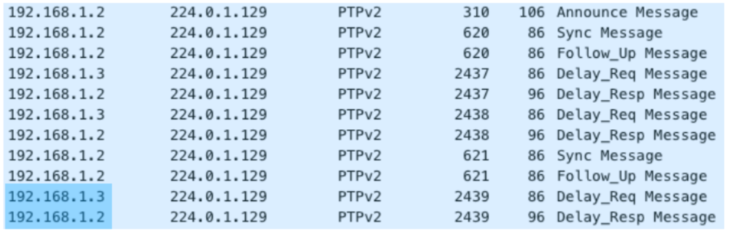
We do not need to know the IP address of the follower since multicast is supported; the multicast address (224.0.1.129) and port (320) always remain the same.

The clock ID of the follower is not required.

We only need to know the MAC address of the PTP enabled switch.

Although the follower recognizes that something is wrong (as reflected in the syslog and management console logs), it still accepts our spoofed SYNC packets.

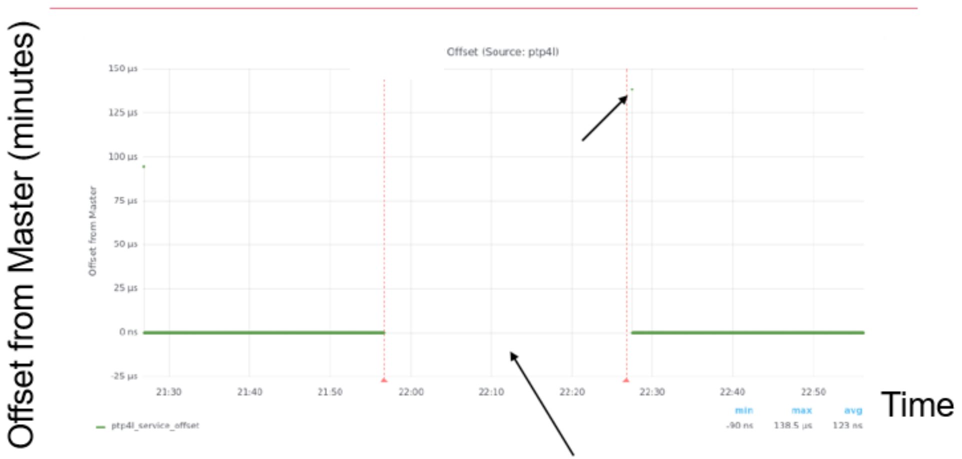
3. Atomic Source Takeover – fake the whole PTP process and pretend to be an atomic clock



```
192.168.1.2    224.0.1.129    PTPv2     310   106 Announce Message
192.168.1.2    224.0.1.129    PTPv2     620    86 Sync Message
192.168.1.2    224.0.1.129    PTPv2     620    86 Follow_Up Message
192.168.1.3    224.0.1.129    PTPv2    2437    86 Delay_Req Message
192.168.1.2    224.0.1.129    PTPv2    2437    96 Delay_Resp Message
192.168.1.3    224.0.1.129    PTPv2    2438    86 Delay_Req Message
192.168.1.2    224.0.1.129    PTPv2    2438    96 Delay_Resp Message
192.168.1.2    224.0.1.129    PTPv2     621    86 Sync Message
192.168.1.2    224.0.1.129    PTPv2     621    86 Follow_Up Message
192.168.1.3    224.0.1.129    PTPv2    2439    86 Delay_Req Message
192.168.1.2    224.0.1.129    PTPv2    2439    96 Delay_Resp Message
```

Follower communicating with fake source

Full sync sequence



Average Offset During Attack: N/A

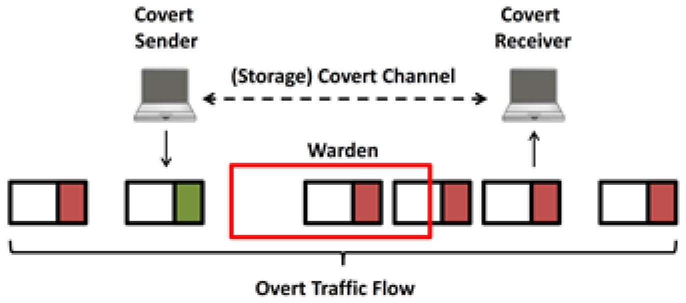
Acts like packets are being dropped

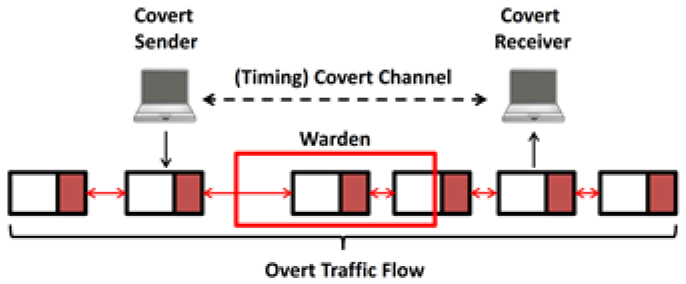
Average Offset After Attack: 148 ns

Possible Mitigation So Far…

- Attacks succeed because there is no authentication between the follower and the clock source
  - No session semantics; follower accepts any packets with a valid sequence number from a reasonable looking IP address
- Construct binding between the source clock ID and its IP address
  - Follower can derive the source clock address and verify the source
- Add a nonce to the source clock sequence numbers to uniquely verify each series of timing packets
- Establish digital identity for clock source (FPA with TAC, see IEEE Trans.)
- Consider NTS for PTP, Tesla, other emerging options

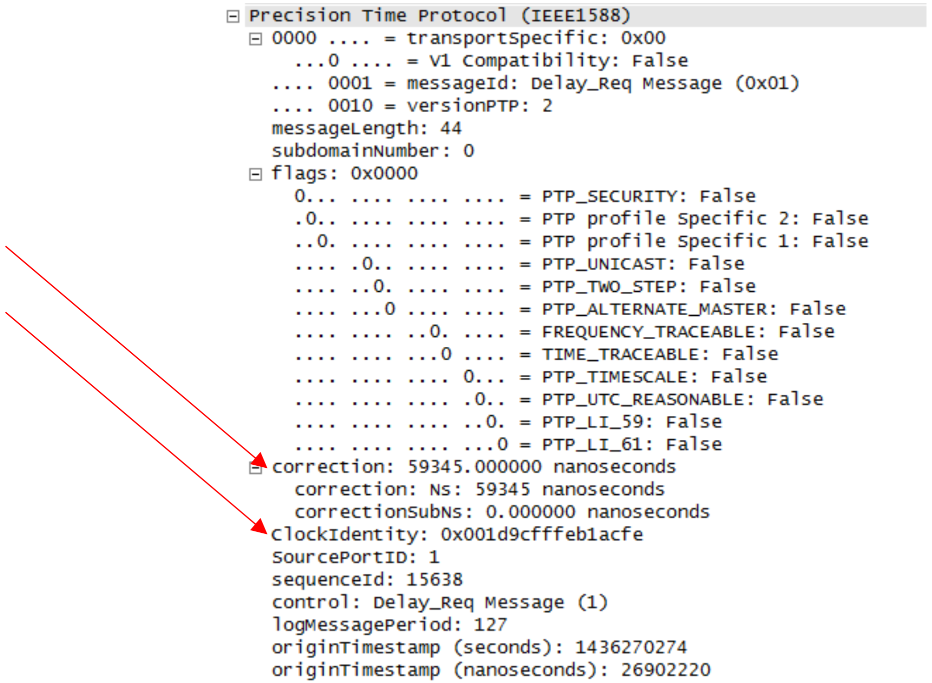
Covert Channels





- Covert channels transfer information between processes that are not normally allowed to communicate based on cybersecurity policy
- Ideally the communication is difficult to detect by other processes unless all meta-data fields are validated, and does not obviously impede normal operation
- Covert channels were not designed for communication, and therefore often exhibit low data rates, lack of redundancy/retransmission or error correction capability
- Often used for data exfiltration or to install/update malware
- Prior documented examples include DNS, NTP, and others (see N. Tsapakis, Virusbulletin.com, April 2019)

PTP Packet Headers as Covert Channels



```
⊟ Precision Time Protocol (IEEE1588)
  ⊟ 0000 .... = transportSpecific: 0x00
      ...0 .... = V1 Compatibility: False
    .... 0001 = messageId: Delay_Req Message (0x01)
    .... 0010 = versionPTP: 2
    messageLength: 44
    subdomainNumber: 0
  ⊟ flags: 0x0000
      0... .... .... .... = PTP_SECURITY: False
      .0.. .... .... .... = PTP profile Specific 2: False
      ..0. .... .... .... = PTP profile Specific 1: False
      .... .0.. .... .... = PTP_UNICAST: False
      .... ..0. .... .... = PTP_TWO_STEP: False
      .... ...0 .... .... = PTP_ALTERNATE_MASTER: False
      .... .... ..0. .... = FREQUENCY_TRACEABLE: False
      .... .... ...0 .... = TIME_TRACEABLE: False
      .... .... .... 0... = PTP_TIMESCALE: False
      .... .... .... .0.. = PTP_UTC_REASONABLE: False
      .... .... .... ..0. = PTP_LI_59: False
      .... .... .... ...0 = PTP_LI_61: False
  ⊟ correction: 59345.000000 nanoseconds
      correction: Ns: 59345 nanoseconds
      correctionSubNs: 0.000000 nanoseconds
    ClockIdentity: 0x001d9cfffeb1acfe
    SourcePortID: 1
    sequenceId: 15638
    control: Delay_Req Message (1)
    logMessagePeriod: 127
    originTimestamp (seconds): 1436270274
    originTimestamp (nanoseconds): 26902220
```

1. Sniff for incoming packets to determine the next sequence ID (only to avoid packet collision).
2. Construct spoofed packet.
    1. 8 bytes is inserted into the correction field during packet creation.
    2. 8 bytes can also optionally be inserted into the clock identity field.
3. Read hexadecimal data from a text file to simulate data exfiltration.
4. Send spoofed packet to source node.
5. Send packets in time intervals that mimic normal occurrences.

Undetectable for delay_request messages

MARIST
School of Computer Science and Mathematics

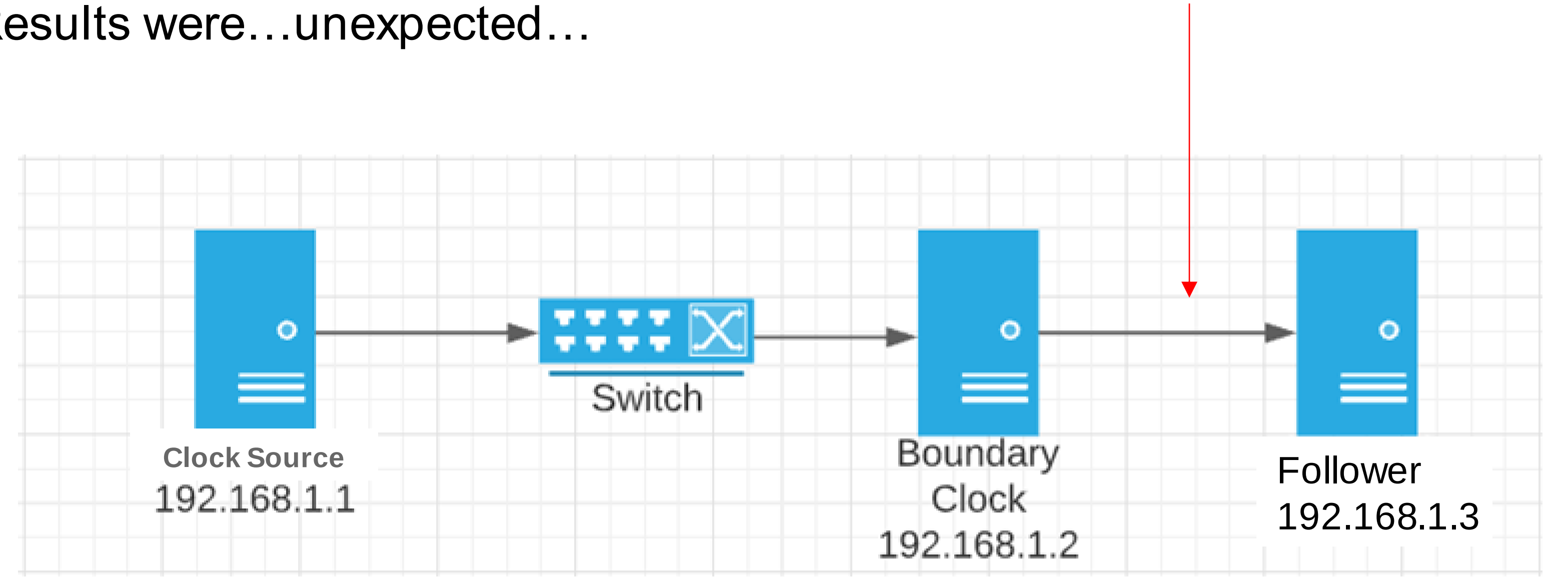
Covert Channel Effects on PTP

- **Node not running ptp**: The source node responds normally to the delay req messages with a delay response
- **Node running ptp with non-colliding sequence ids**: Same reaction as a node that's not running ptp; normal response, undetected *(this is part of the default IBM Enterprise Profile for PTP)*
- **Node running ptp with colliding sequence ids:** The data in the correction field is reflected by the raw delay value in ptp output. Source still sends delay response
- **Node running ptp with colliding sequence ids & tsproc_mode set to raw:** Large source offsets since the offset is now computed taking into account the raw value

Leads to 2 new attack vectors…

4. Correction Field MITM Attack

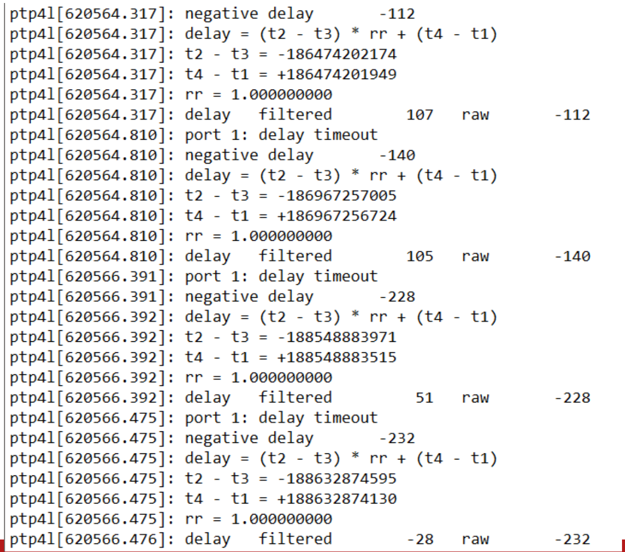
- Intercepting packets before they leave the boundary node, injecting large data into the correction field to cause large offset at the follower node
- Results were…unexpected…



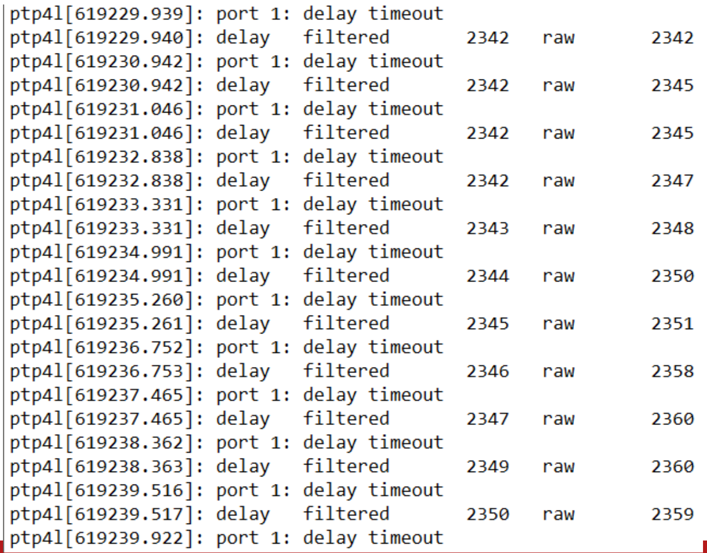
MITM Attack Results

- If correction field values are **too small**, we get negative delay messages but PHC2SYS still runs fine

- If correction field values are **too large**, we stop getting offset messages at all; impossible to graph offsets or tell how this attack impacts PTP4L



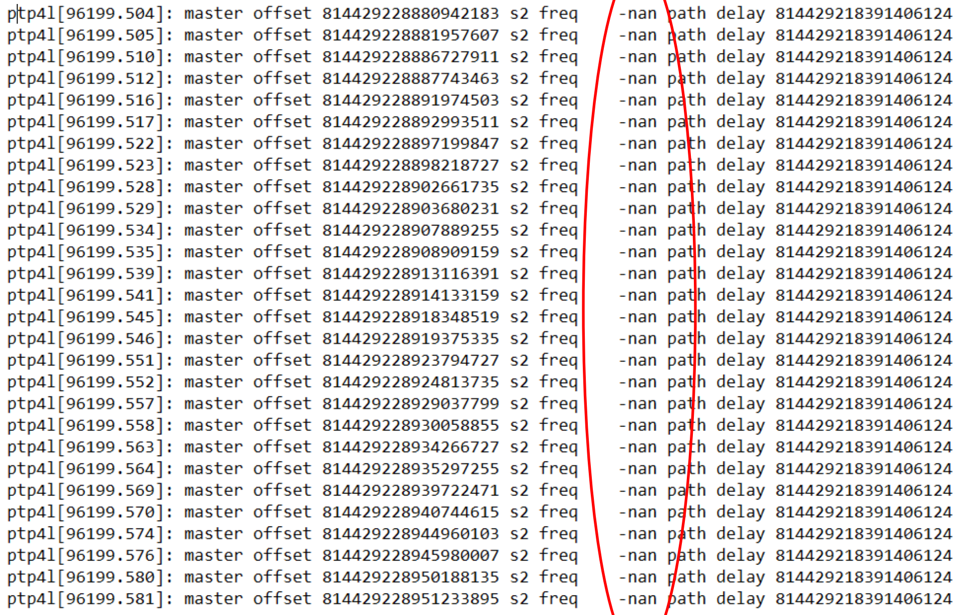
```
ptp4l[620564.317]: negative delay        -112
ptp4l[620564.317]: delay = (t2 - t3) * rr + (t4 - t1)
ptp4l[620564.317]: t2 - t3 = -186474202174
ptp4l[620564.317]: t4 - t1 = +186474201949
ptp4l[620564.317]: rr = 1.000000000
ptp4l[620564.317]: delay   filtered        107    raw        -112
ptp4l[620564.810]: port 1: delay timeout
ptp4l[620564.810]: negative delay        -140
ptp4l[620564.810]: delay = (t2 - t3) * rr + (t4 - t1)
ptp4l[620564.810]: t2 - t3 = -186967257005
ptp4l[620564.810]: t4 - t1 = +186967256724
ptp4l[620564.810]: rr = 1.000000000
ptp4l[620564.810]: delay   filtered        105    raw        -140
ptp4l[620566.391]: port 1: delay timeout
ptp4l[620566.391]: negative delay        -228
ptp4l[620566.392]: delay = (t2 - t3) * rr + (t4 - t1)
ptp4l[620566.392]: t2 - t3 = -188548883971
ptp4l[620566.392]: t4 - t1 = +188548883515
ptp4l[620566.392]: rr = 1.000000000
ptp4l[620566.392]: delay   filtered         51    raw        -228
ptp4l[620566.475]: port 1: delay timeout
ptp4l[620566.475]: negative delay        -232
ptp4l[620566.475]: delay = (t2 - t3) * rr + (t4 - t1)
ptp4l[620566.475]: t2 - t3 = -188632874595
ptp4l[620566.475]: t4 - t1 = +188632874130
ptp4l[620566.475]: rr = 1.000000000
ptp4l[620566.476]: delay   filtered        -28    raw        -232
```



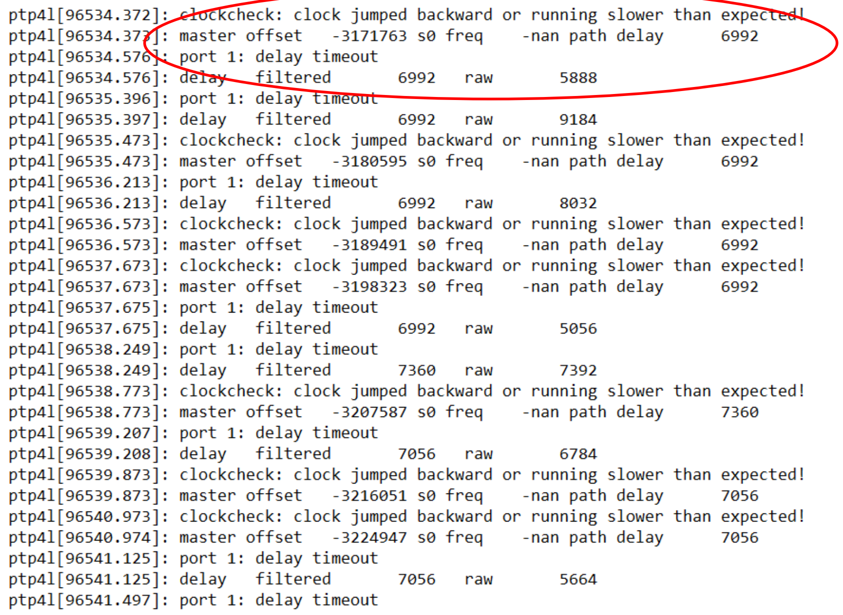
```
ptp4l[619229.939]: port 1: delay timeout
ptp4l[619229.940]: delay   filtered      2342    raw        2342
ptp4l[619230.942]: port 1: delay timeout
ptp4l[619230.942]: delay   filtered      2342    raw        2345
ptp4l[619231.046]: port 1: delay timeout
ptp4l[619231.046]: delay   filtered      2342    raw        2345
ptp4l[619232.838]: port 1: delay timeout
ptp4l[619232.838]: delay   filtered      2342    raw        2347
ptp4l[619233.331]: port 1: delay timeout
ptp4l[619233.331]: delay   filtered      2343    raw        2348
ptp4l[619234.991]: port 1: delay timeout
ptp4l[619234.991]: delay   filtered      2344    raw        2350
ptp4l[619235.260]: port 1: delay timeout
ptp4l[619235.261]: delay   filtered      2345    raw        2351
ptp4l[619236.752]: port 1: delay timeout
ptp4l[619236.753]: delay   filtered      2346    raw        2358
ptp4l[619237.465]: port 1: delay timeout
ptp4l[619237.465]: delay   filtered      2347    raw        2360
ptp4l[619238.362]: port 1: delay timeout
ptp4l[619238.363]: delay   filtered      2349    raw        2360
ptp4l[619239.516]: port 1: delay timeout
ptp4l[619239.517]: delay   filtered      2350    raw        2359
ptp4l[619239.922]: port 1: delay timeout
```

5. Clock Frequency Manipulation Attack

- Spoof packets with large amounts of data in correction field
- Clock frequency exceeds max value, unable to synchronize with source



```
ptp4l[96199.504]: master offset 814429228880942183 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.505]: master offset 814429228881957607 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.510]: master offset 814429228886727911 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.512]: master offset 814429228887743463 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.516]: master offset 814429228891974503 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.517]: master offset 814429228892993511 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.522]: master offset 814429228897199847 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.523]: master offset 814429228898218727 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.528]: master offset 814429228902661735 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.529]: master offset 814429228903680231 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.534]: master offset 814429228907889255 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.535]: master offset 814429228908909159 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.539]: master offset 814429228913116391 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.541]: master offset 814429228914133159 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.545]: master offset 814429228918348519 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.546]: master offset 814429228919375335 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.551]: master offset 814429228923794727 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.552]: master offset 814429228924813735 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.557]: master offset 814429228929037799 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.558]: master offset 814429228930058855 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.563]: master offset 814429228934266727 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.564]: master offset 814429228935297255 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.569]: master offset 814429228939722471 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.570]: master offset 814429228940744615 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.574]: master offset 814429228944960103 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.576]: master offset 814429228945980007 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.580]: master offset 814429228950188135 s2 freq    -nan path delay 814429218391406124
ptp4l[96199.581]: master offset 814429228951233895 s2 freq    -nan path delay 814429218391406124
```

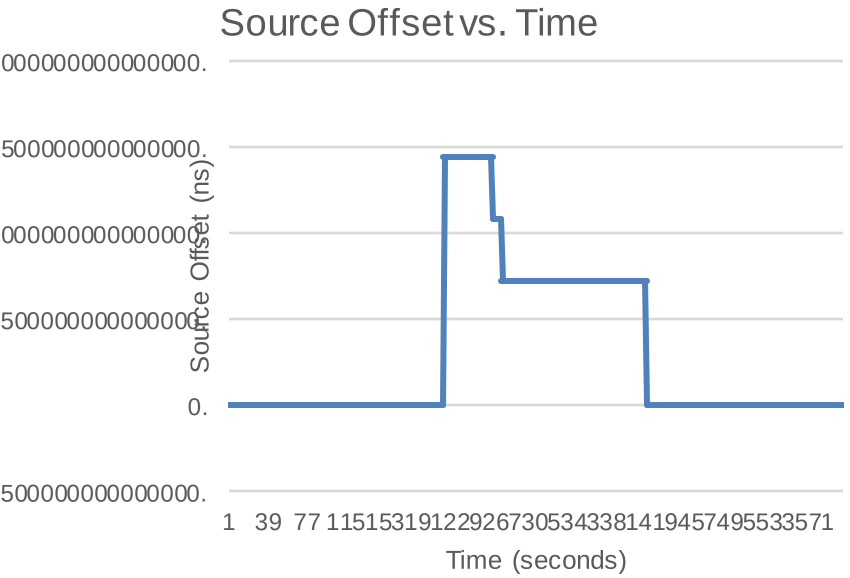


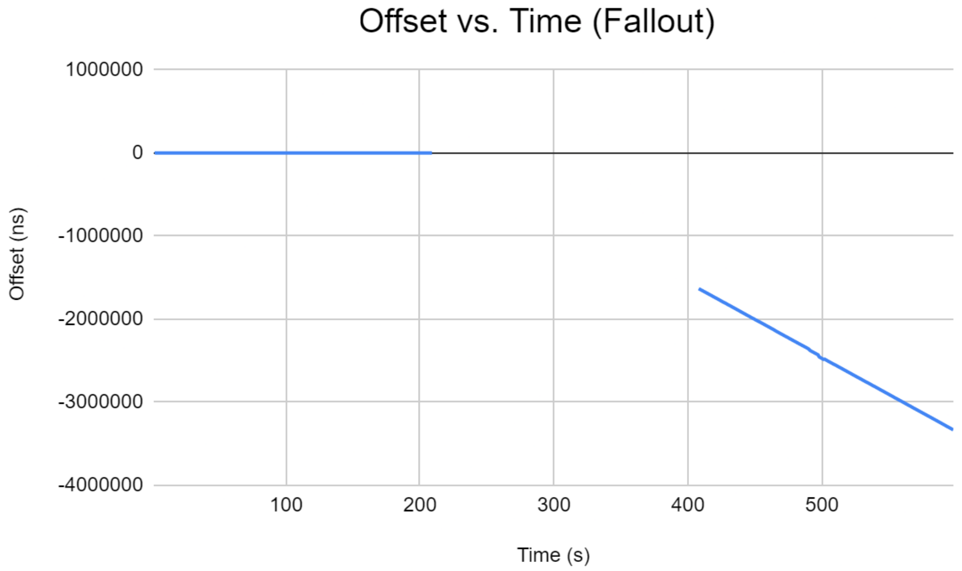
```
ptp4l[96534.372]: clockcheck: clock jumped backward or running slower than expected!
ptp4l[96534.373]: master offset   -3171763 s0 freq     -nan path delay      6992
ptp4l[96534.576]: port 1: delay timeout
ptp4l[96534.576]: delay   filtered       6992   raw       5888
ptp4l[96535.396]: port 1: delay timeout
ptp4l[96535.397]: delay   filtered       6992   raw       9184
ptp4l[96535.473]: clockcheck: clock jumped backward or running slower than expected!
ptp4l[96535.473]: master offset   -3180595 s0 freq     -nan path delay      6992
ptp4l[96536.213]: port 1: delay timeout
ptp4l[96536.213]: delay   filtered       6992   raw       8032
ptp4l[96536.573]: clockcheck: clock jumped backward or running slower than expected!
ptp4l[96536.573]: master offset   -3189491 s0 freq     -nan path delay      6992
ptp4l[96537.673]: clockcheck: clock jumped backward or running slower than expected!
ptp4l[96537.673]: master offset   -3198323 s0 freq     -nan path delay      6992
ptp4l[96537.675]: port 1: delay timeout
ptp4l[96537.675]: delay   filtered       6992   raw       5056
ptp4l[96538.249]: port 1: delay timeout
ptp4l[96538.249]: delay   filtered       7360   raw       7392
ptp4l[96538.773]: clockcheck: clock jumped backward or running slower than expected!
ptp4l[96538.773]: master offset   -3207587 s0 freq     -nan path delay      7360
ptp4l[96539.207]: port 1: delay timeout
ptp4l[96539.208]: delay   filtered       7056   raw       6784
ptp4l[96539.873]: clockcheck: clock jumped backward or running slower than expected!
ptp4l[96539.873]: master offset   -3216051 s0 freq     -nan path delay      7056
ptp4l[96540.973]: clockcheck: clock jumped backward or running slower than expected!
ptp4l[96540.974]: master offset   -3224947 s0 freq     -nan path delay      7056
ptp4l[96541.125]: port 1: delay timeout
ptp4l[96541.125]: delay   filtered       7056   raw       5664
ptp4l[96541.497]: port 1: delay timeout
```

MARIST
School of Computer Science and Mathematics

Clock Frequency Manipulation Attack Results

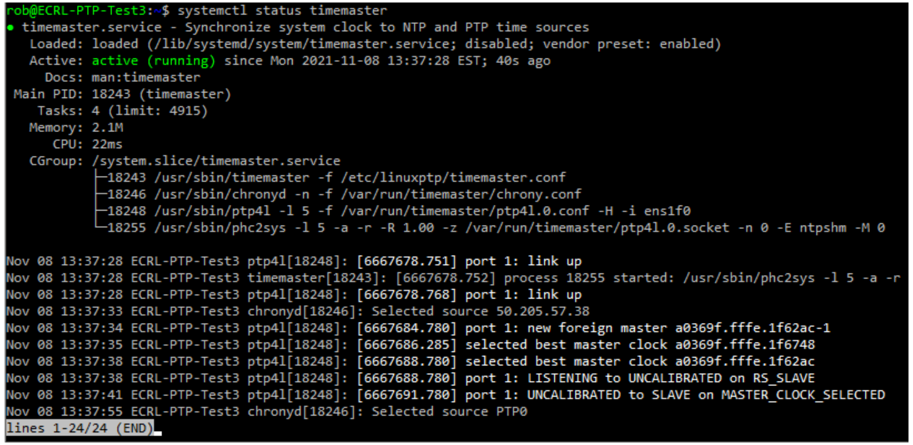
- Clock servo algorithm unable to synchronize back to the source
- Source offset continues to drift even after the attack has concluded





Timemaster

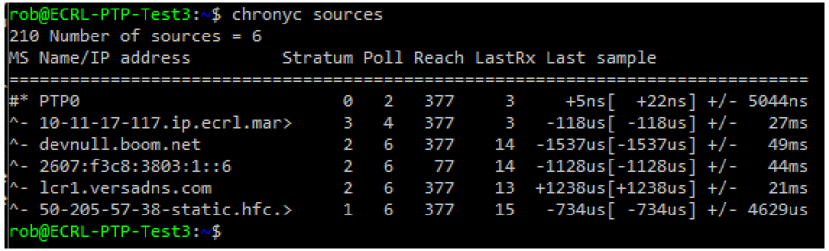
- Timemaster is a lightweight service that incorporates a defense in depth strategy for timing synchronization
- NTP and PTP run simultaneously, with NTP acting as a fallback in case the PTP network goes unresponsive or is attacked
- Uses NTP as a fallback timing service if the PTP network is compromised
- All Timemaster testing was done using chronyd, an ntp daemon, and the linuxptp package



```
rob@ECRL-PTP-Test3:~$ systemctl status timemaster
● timemaster.service - Synchronize system clock to NTP and PTP time sources
   Loaded: loaded (/lib/systemd/system/timemaster.service; disabled; vendor preset: enabled)
   Active: active (running) since Mon 2021-11-08 13:37:28 EST; 40s ago
     Docs: man:timemaster
 Main PID: 18243 (timemaster)
    Tasks: 4 (limit: 4915)
   Memory: 2.1M
      CPU: 22ms
   CGroup: /system.slice/timemaster.service
           ├─18243 /usr/sbin/timemaster -f /etc/linuxptp/timemaster.conf
           ├─18246 /usr/sbin/chronyd -n -f /var/run/timemaster/chrony.conf
           ├─18248 /usr/sbin/ptp4l -l 5 -f /var/run/timemaster/ptp4l.0.conf -H -i ens1f0
           └─18255 /usr/sbin/phc2sys -l 5 -a -r -R 1.00 -z /var/run/timemaster/ptp4l.0.socket -n 0 -E ntpshm -M 0

Nov 08 13:37:28 ECRL-PTP-Test3 ptp4l[18248]: [6667678.751] port 1: link up
Nov 08 13:37:28 ECRL-PTP-Test3 timemaster[18243]: [6667678.752] process 18255 started: /usr/sbin/phc2sys -l 5 -a -r
Nov 08 13:37:28 ECRL-PTP-Test3 ptp4l[18248]: [6667678.768] port 1: link up
Nov 08 13:37:33 ECRL-PTP-Test3 chronyd[18246]: Selected source 50.205.57.38
Nov 08 13:37:34 ECRL-PTP-Test3 ptp4l[18248]: [6667684.780] port 1: new foreign master a0369f.fffe.1f62ac-1
Nov 08 13:37:35 ECRL-PTP-Test3 ptp4l[18248]: [6667686.285] selected best master clock a0369f.fffe.1f6748
Nov 08 13:37:38 ECRL-PTP-Test3 ptp4l[18248]: [6667688.780] selected best master clock a0369f.fffe.1f62ac
Nov 08 13:37:38 ECRL-PTP-Test3 ptp4l[18248]: [6667688.780] port 1: LISTENING to UNCALIBRATED on RS_SLAVE
Nov 08 13:37:41 ECRL-PTP-Test3 ptp4l[18248]: [6667691.780] port 1: UNCALIBRATED to SLAVE on MASTER_CLOCK_SELECTED
Nov 08 13:37:55 ECRL-PTP-Test3 chronyd[18246]: Selected source PTP0
lines 1-24/24 (END)
```

Shows ECRL-PTP-Test1 getting selected by PTP as the source clock.
Chronyd output shows PTP getting selected as the most accurate time source.



```
rob@ECRL-PTP-Test3:~$ chronyc sources
210 Number of sources = 6
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
#* PTP0                          0   2   377     3     +5ns[  +22ns] +/- 5044ns
^- 10-11-17-117.ip.ecrl.mar>     3   4   377     3   -118us[ -118us] +/-   27ms
^- devnull.boom.net              2   6   377    14  -1537us[-1537us] +/-   49ms
^- 2607:f3c8:3803:1::6           2   6    77    14  -1128us[-1128us] +/-   44ms
^- lcr1.versadns.com             2   6   377    13  +1238us[+1238us] +/-   21ms
^- 50-205-57-38-static.hfc.>     1   6   377    15   -734us[ -734us] +/- 4629us
rob@ECRL-PTP-Test3:~$
```

- Chrony selects PTP as the optimal time source, which is shown by the “*” symbol. The “#” symbol identifies PTP as an internal time source. The last sample column shows the current offset, which is +/- 5ns.

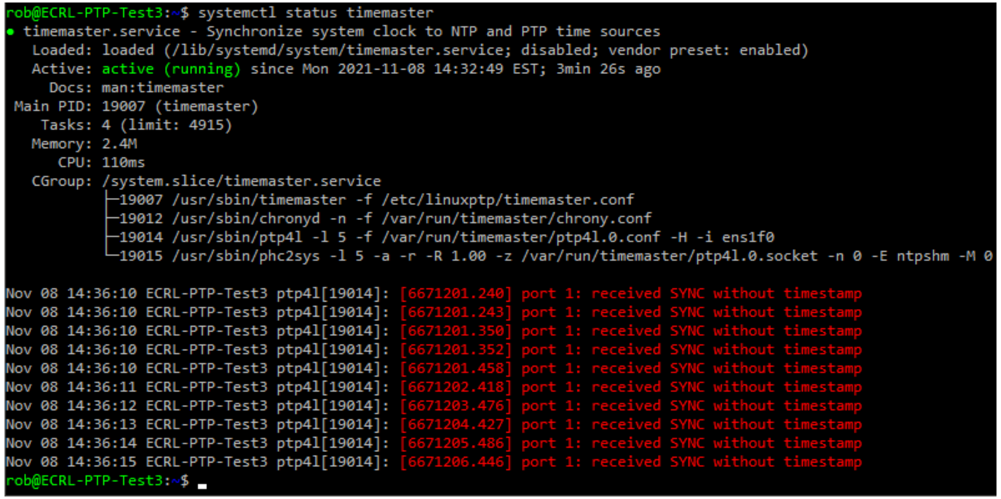
- ECRL-PTP-Test1 (10.11.17.117) is advertised as the next most accurate time source as an NTP server.

MARIST

School of Computer Science and Mathematics

Timemaster Drawbacks

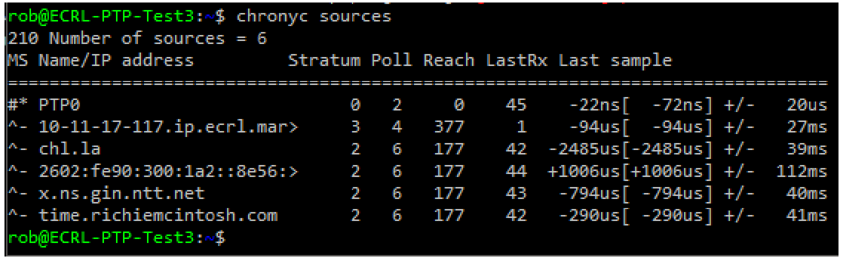
- Unable to defend against passive attacks that manipulate a PTP network
- Most notably susceptible to the Source Clock Takeover attack and Covert Channels vulnerability since they do not produce large offsets while active in the a network



```
rob@ECRL-PTP-Test3:~$ systemctl status timemaster
● timemaster.service - Synchronize system clock to NTP and PTP time sources
   Loaded: loaded (/lib/systemd/system/timemaster.service; disabled; vendor preset: enabled)
   Active: active (running) since Mon 2021-11-08 14:32:49 EST; 3min 26s ago
     Docs: man:timemaster
 Main PID: 19007 (timemaster)
    Tasks: 4 (limit: 4915)
   Memory: 2.4M
      CPU: 110ms
   CGroup: /system.slice/timemaster.service
           ├─19007 /usr/sbin/timemaster -f /etc/linuxptp/timemaster.conf
           ├─19012 /usr/sbin/chronyd -n -f /var/run/timemaster/chrony.conf
           ├─19014 /usr/sbin/ptp4l -l 5 -f /var/run/timemaster/ptp4l.0.conf -H -i ens1f0
           └─19015 /usr/sbin/phc2sys -l 5 -a -r -R 1.00 -z /var/run/timemaster/ptp4l.0.socket -n 0 -E ntpshm -M 0

Nov 08 14:36:10 ECRL-PTP-Test3 ptp4l[19014]: [6671201.240] port 1: received SYNC without timestamp
Nov 08 14:36:10 ECRL-PTP-Test3 ptp4l[19014]: [6671201.243] port 1: received SYNC without timestamp
Nov 08 14:36:10 ECRL-PTP-Test3 ptp4l[19014]: [6671201.350] port 1: received SYNC without timestamp
Nov 08 14:36:10 ECRL-PTP-Test3 ptp4l[19014]: [6671201.352] port 1: received SYNC without timestamp
Nov 08 14:36:10 ECRL-PTP-Test3 ptp4l[19014]: [6671201.458] port 1: received SYNC without timestamp
Nov 08 14:36:11 ECRL-PTP-Test3 ptp4l[19014]: [6671202.418] port 1: received SYNC without timestamp
Nov 08 14:36:12 ECRL-PTP-Test3 ptp4l[19014]: [6671203.476] port 1: received SYNC without timestamp
Nov 08 14:36:13 ECRL-PTP-Test3 ptp4l[19014]: [6671204.427] port 1: received SYNC without timestamp
Nov 08 14:36:14 ECRL-PTP-Test3 ptp4l[19014]: [6671205.486] port 1: received SYNC without timestamp
Nov 08 14:36:15 ECRL-PTP-Test3 ptp4l[19014]: [6671206.446] port 1: received SYNC without timestamp
rob@ECRL-PTP-Test3:~$
```

- PTP displays "port 1: received SYNC without timestamp" error messages, which is a normal occurrence during the source takeover attack



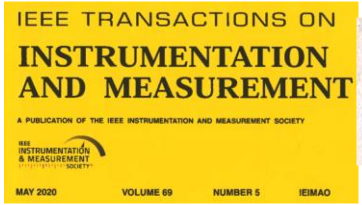
```
rob@ECRL-PTP-Test3:~$ chronyc sources
210 Number of sources = 6
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
#* PTP0                          0   2     0    45    -22ns[  -72ns] +/-   20us
^- 10-11-17-117.ip.ecrl.mar>     3   4   377     1    -94us[  -94us] +/-   27ms
^- chl.la                        2   6   177    42  -2485us[-2485us] +/-   39ms
^- 2602:fe90:300:1a2::8e56:>     2   6   177    44  +1006us[+1006us] +/-  112ms
^- x.ns.gin.ntt.net              2   6   177    43   -794us[ -794us] +/-   40ms
^- time.richiemcintosh.com       2   6   177    42   -290us[ -290us] +/-   41ms
rob@ECRL-PTP-Test3:~$
```

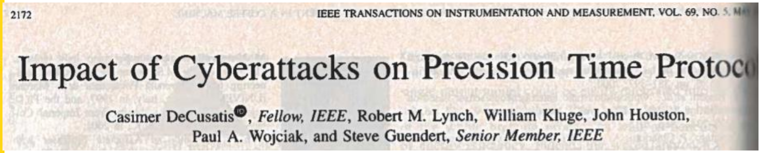
- Chrony still thinks that PTP is the most accurate time source, unaware of any compromise to the protocol.

MARIST
School of Computer Science and Mathematics

Summary and Conclusions

- PTP protocols are susceptible to a number of different attacks
  - Announce DoS
  - Source Spoof
  - Atomic Source Takeover
  - Correction Field MITM
  - Clock Frequency Manipulation


IEEE TRANSACTIONS ON
INSTRUMENTATION AND MEASUREMENT
A PUBLICATION OF THE IEEE INSTRUMENTATION AND MEASUREMENT SOCIETY
MAY 2020 VOLUME 69 NUMBER 5 IEIMAO


2172 IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, VOL. 69, NO. 5, MA

Impact of Cyberattacks on Precision Time Protoco

Casimer DeCusatis, *Fellow, IEEE*, Robert M. Lynch, William Kluge, John Houston, Paul A. Wojciak, and Steve Guendert, *Senior Member, IEEE*

- Mitigation for the first three attacks has been proposed to the IEEE through IBM
- The last two attacks remain poorly understood
- Several covert channel options were also identified
- Ongoing investigations into PTP time cybersecurity