CT420 REAL-TIME SYSTEMS

THE PTP PROTOCOL

Dr. Michael Schukat

Ollscoil na Gaillimhe
University of Galway

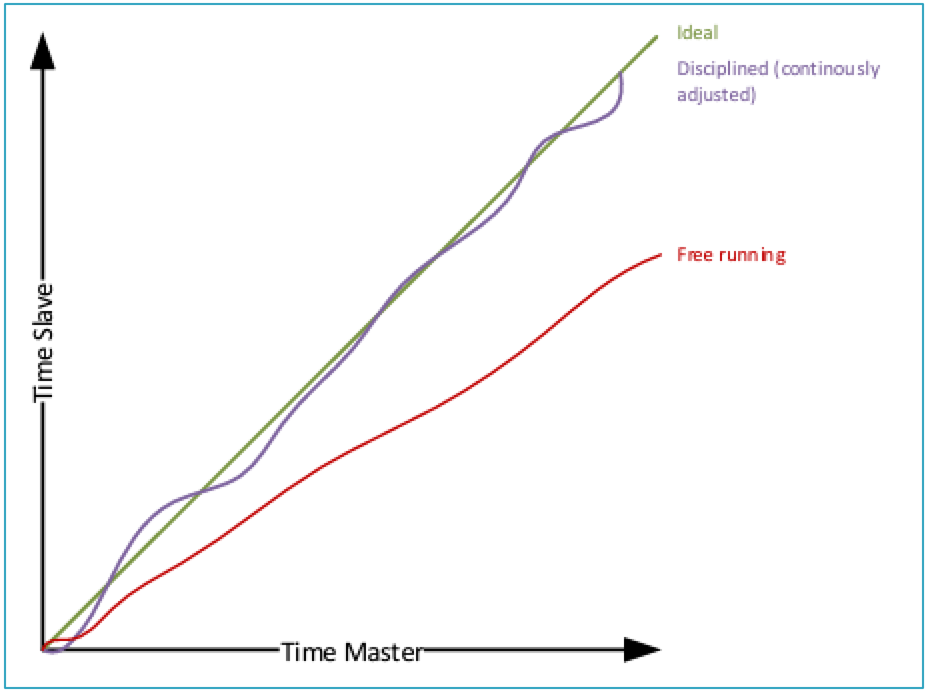
Recap: Typical Time Synchronisation Requirements of critical Infrastructure

- Accurate time synchronisation is required in many domains including critical infrastructure, transportation, and financial services
- NTP may or may not be good enough to provide required levels of synchronisation
- Examples for high levels of synchronisation that cannot be achieved by NTP:

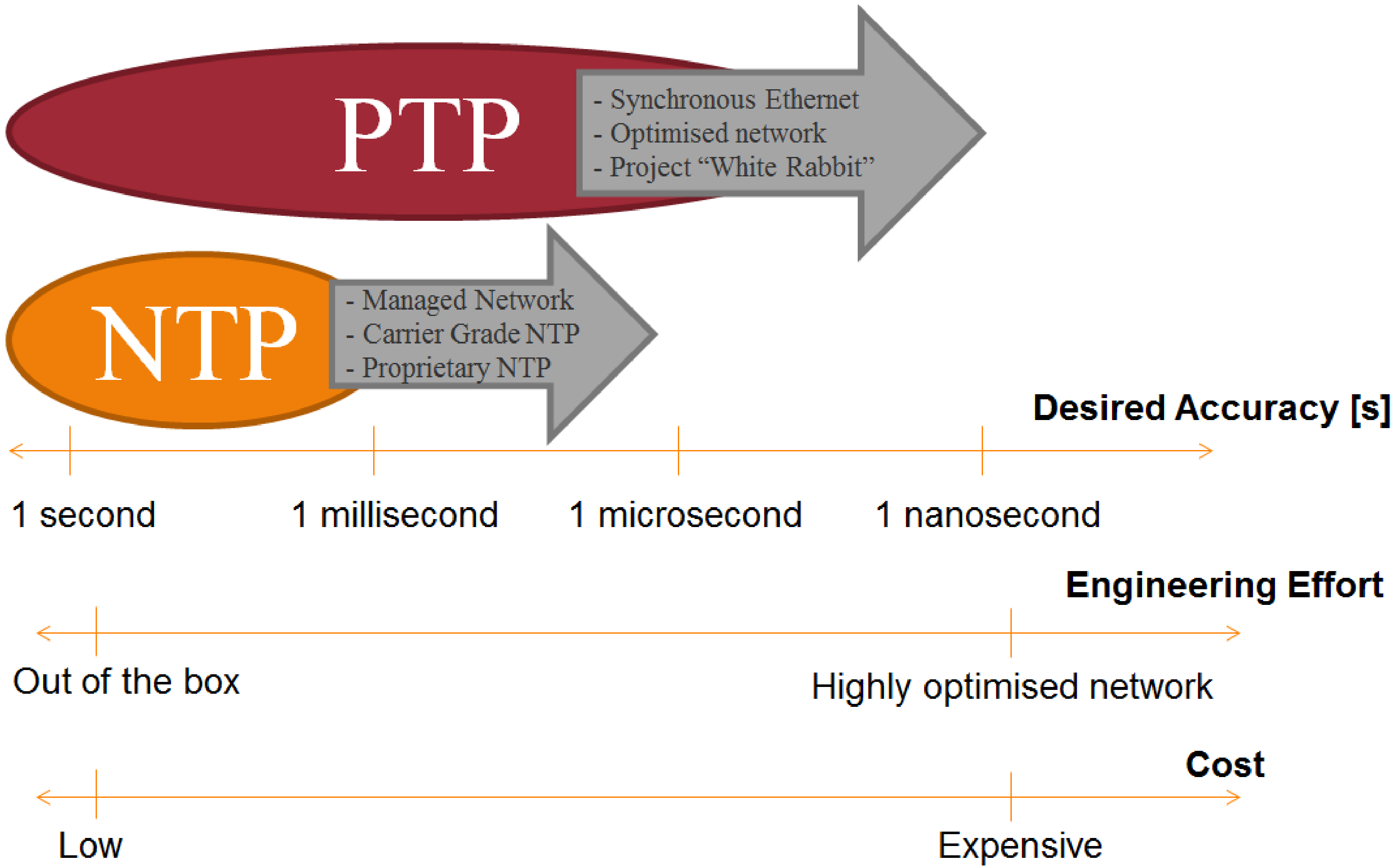
Domain / Standard	Application	Required Accuracy
North American Electric Reliability Cooperation (NERC)	Monitoring power distribution network dynamics (Synchrophasors)	< 1.7 µs
TDD and LTE-A systems	Network packet synchronisation	< 1.5 µs
Markets in Financial Instruments Directive (MiFID II)	Timestamping of financial transactions	< 100 µs

Recap: Free-Running versus NTP/PTP corrected Clocks

- The disciplined clock is never set to an earlier time
- Instead, the clock ticks slower or faster to catch up with the reference time or master clock



Recap: NTP versus PTP



IEEE 1588 (PTP) Overview

- PTP is designed for systems that require up to microsecond / sub-microsecond synchronisation
  - Differs from NTP – rather than relying on various time sources interconnected via an unmanaged network (i.e. WAN), we rely on a **single** time reference (the **grandmaster clock**, i.e., the **master**) interconnected to **multiple slaves** via a **managed network**
  - Hardware timestamping on devices → later
  - Network hardware support → later
  - More frequent polling (to compensate local clock skew)
- This comes at a price! PTP expects that
  - the underlying network is tightly managed while network and components are selected / configured to minimise asymmetry
  - network traffic patterns are controlled so that traffic variation is minimised
- Ideally PTP messages should be prioritised and network hardware should be replaced by PTP-aware devices

Overview

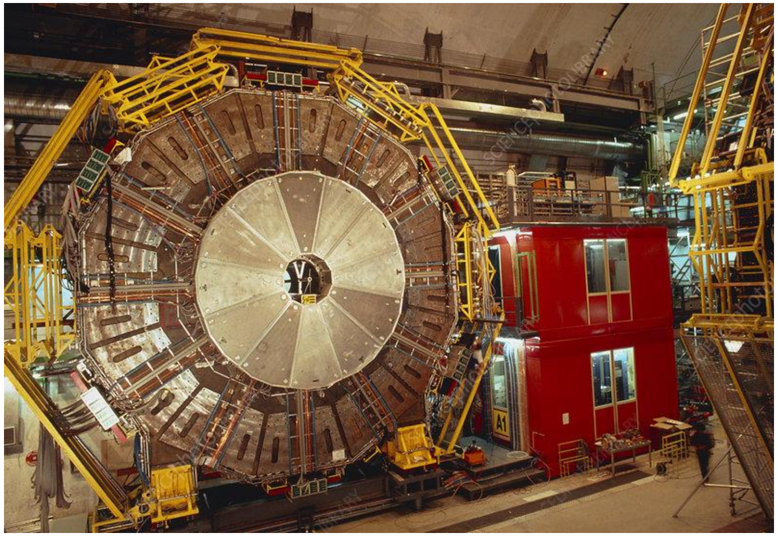
- The Precision Time Protocol (PTP) is typically deployed in LAN or WAN
- Version 1 of PTP, IEEE 1588-2002, was published in 2002
- IEEE 1588-2008, also known as PTP Version 2 (not backward compatible with the original 2002 version) introduced among other things
  - PTP-aware network components
  - a profile concept that defines PTP operating parameters and options for specific applications, e.g., telecommunications and electric power distribution
  - an experimental (and terribly flawed) security extension (Annex K)
- IEEE 1588-2019 was published in November 2019 and includes backward-compatible improvements to the 2008 publication, including security extensions in Annex P

Use Case CERN / Large Hadron Collider (LHC)

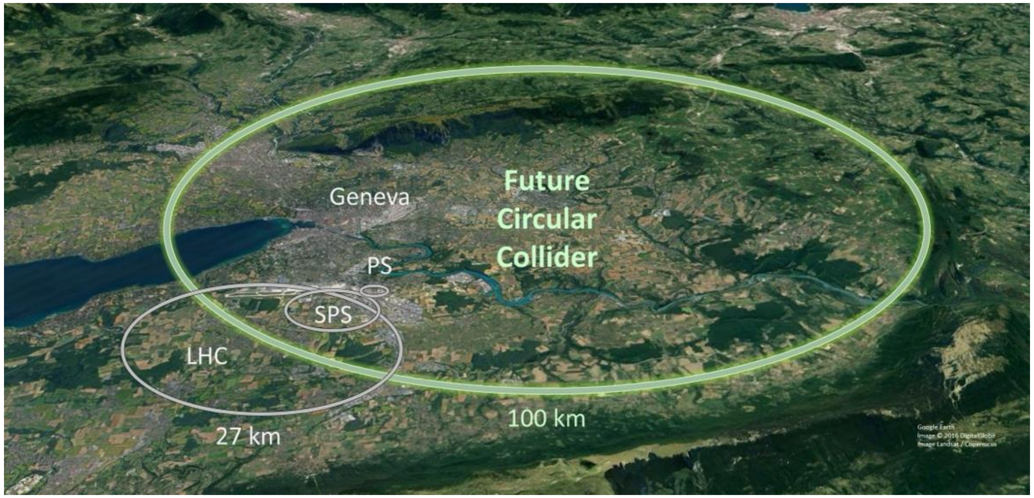
- The LHC is the world's largest and highest-energy particle collider
  - A collider is a type of a particle accelerator which brings two opposing particle beams together such that the particles collide
  - Analysis of the byproducts of these collisions provide evidence of the structure of the subatomic world and the laws of nature governing it
- It is a ring-shaped machine that lies in a tunnel 27 kilometres in circumference beneath the France–Switzerland border
- The collider has four crossing points where the accelerated particles collide
- Seven detectors, each designed to detect different phenomena, are positioned around the crossing points to observe / measure the collisions and their byproducts
- Many of these byproducts decay after very short periods of time
- Therefore the detectors need to be **exactly time synchronised** to correlate the signals they detect

CERN / Large Hadron Collider (LHC) – Some Stock Images

- The ALEPH particle collider



- LHC dimensions



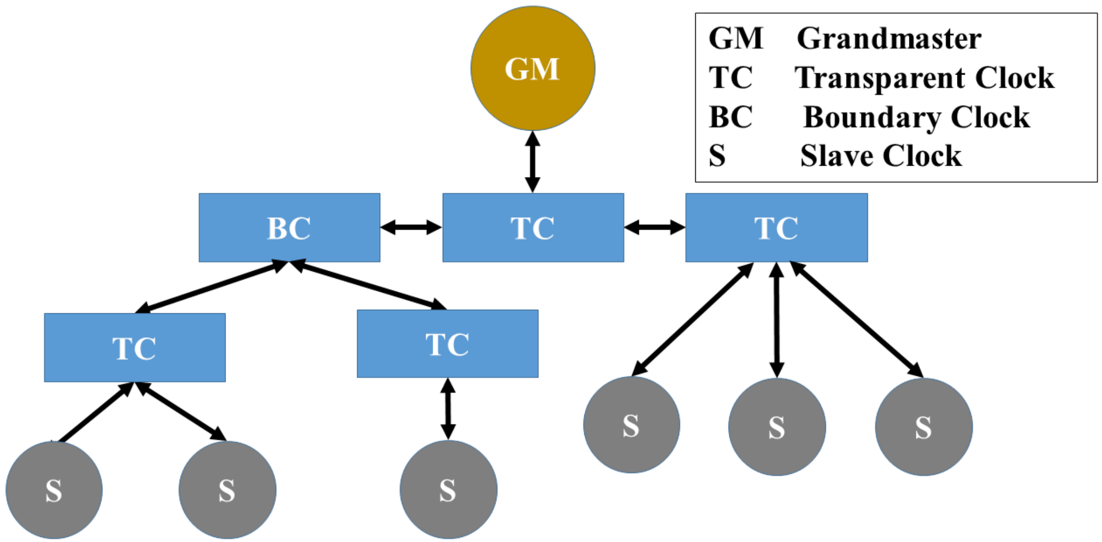
Use Case: Project White Rabbit

- Collaborative CERN project that developed a
  - fully deterministic Ethernet-based network for general purpose data transfer
  - End-point synchronisation of 1000+ nodes with sub-nanosecond accuracy via fiber or copper cables of up to 10 km of length
- Based on PTP (of course) and Synchronous Ethernet
  - Synchronous Ethernet is an ITU-T standard that provides mechanisms to transfer an accurate 125 MHz square signal over the Ethernet physical layer
  - This provides a common clock reference for all endpoints, i.e. no clock skew!
  - PTP is subsequently used for offset corrections
- The hardware designs as well as the source code are publicly available
  - See https://ohwr.org/projects/white-rabbit/

PTP Clock Types

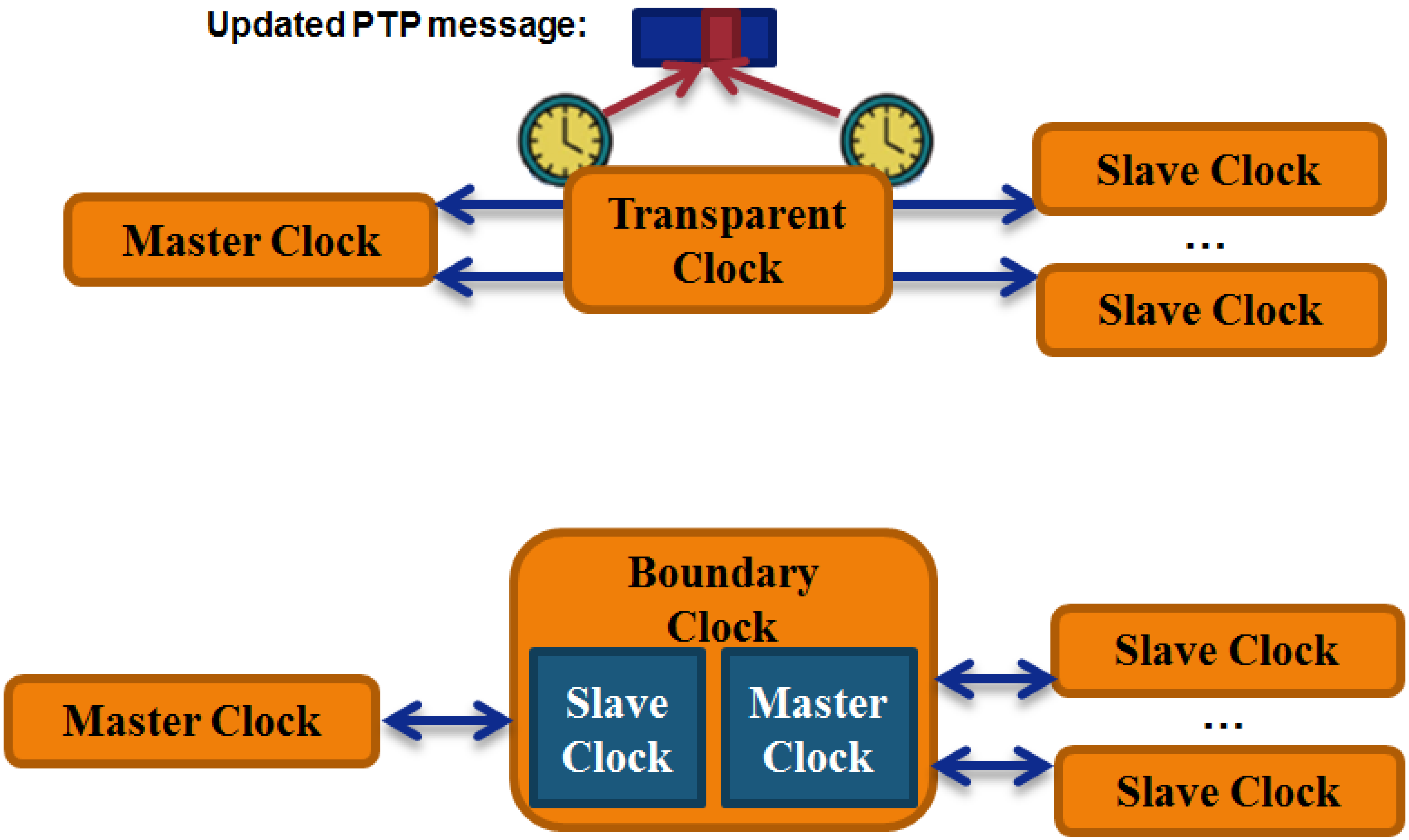
1. (Grand) Master clock
   - Single time reference for all other clocks
   - Is chosen dynamically among all clocks in a network
2. Ordinary clock, can be one of the following:
   1. Slave only clock, receiving time from the above master clock
   2. Preferred grandmaster, only acts as a master, never as a slave
   3. Master clock or slave clock
3. Boundary clock
   - A network switch that gets time from a master clock, but acts as a master to multiple downstream slaves
4. Transparent clock
   - A network switch that performs hardware timestamping whenever a time synchronisation message arrives or departs, thus correcting for residency time via correction field

Example PTP Master / Slave Hierarchy



- A single GM is responsible for the synchronisation of multiple slave clocks over a (tightly managed) LAN
  - E.g. traffic throttling, over-provisioning of bandwidth
- While ordinary network switches can be used, these are often replaced by or complemented with PTP-aware infrastructure components that allow for a better time synchronisation

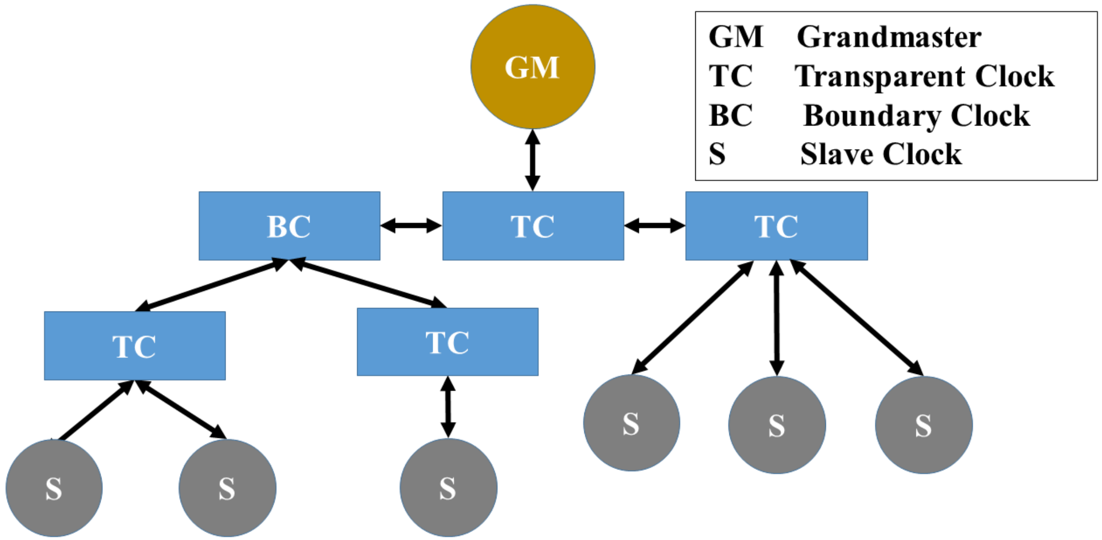
Transparent Clock versus Boundary Clock



PTP Domains

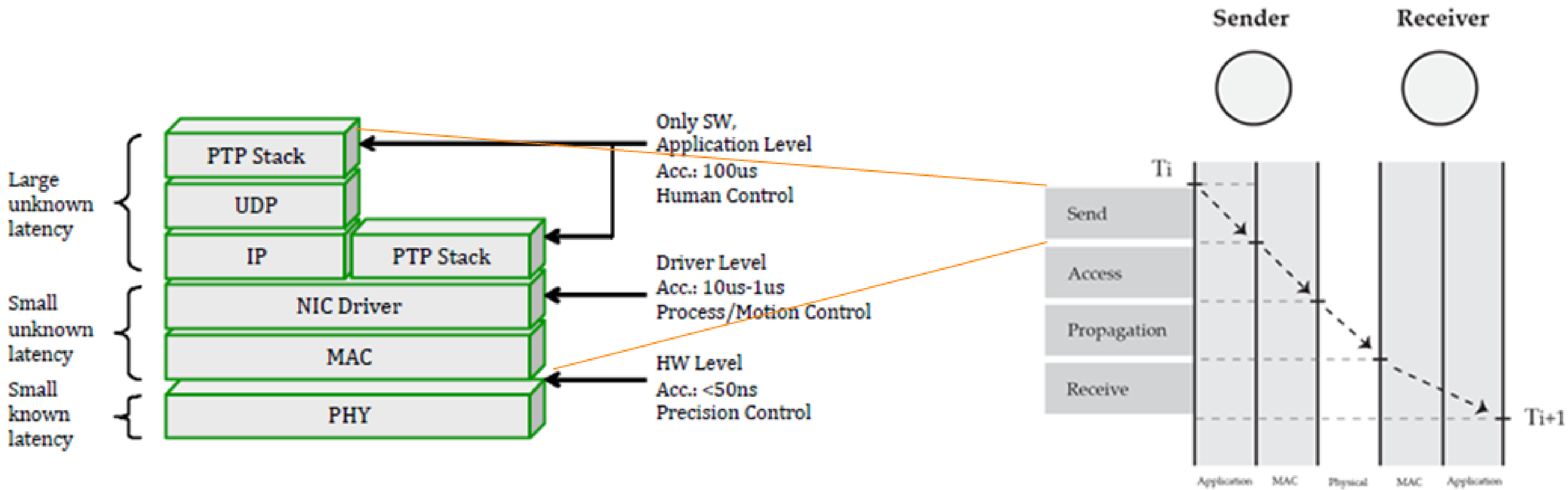
- Clocks in a PTP network may be clustered, as they operate on different operational parameters
  - Different to NTP, where clients are configured individually
- A domain is a group of PTP nodes / clocks that communicate with each other on a link
- One network can contain different PTP domains, but they are considered independent and operate independent
- The frame of a PTP message provides information on the domain number (domainNumber), see slide with common message header
- Domain numbers ranging range between 0 and 255

Example PTP Master / Slave Hierarchy with a single Domain



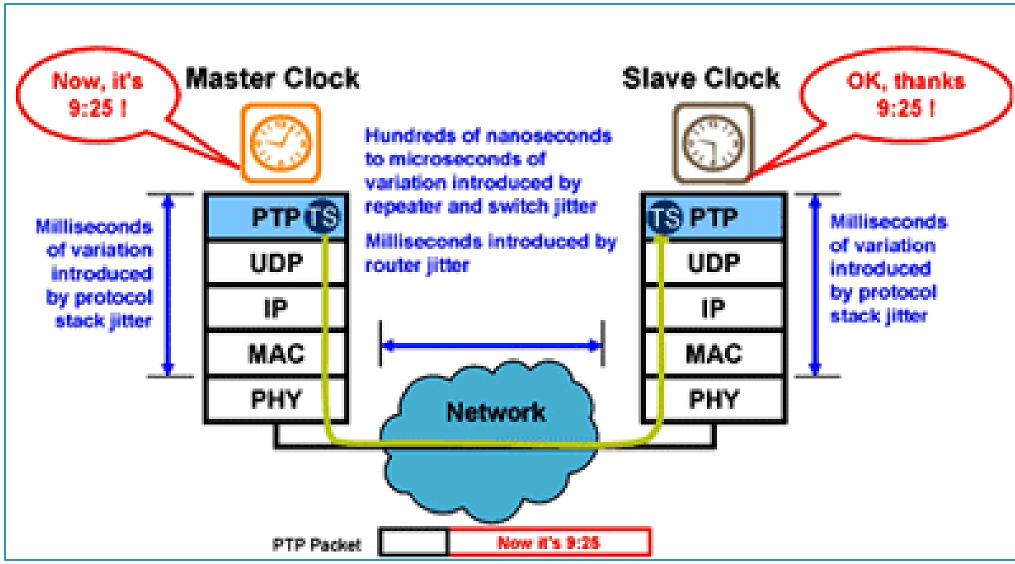
- Here we see a single domain (→ next slide) consisting of a single grandmaster (GM) and multiple slaves (S), that are interconnected via a boundary clock (BC) and four transparent clocks (TC)

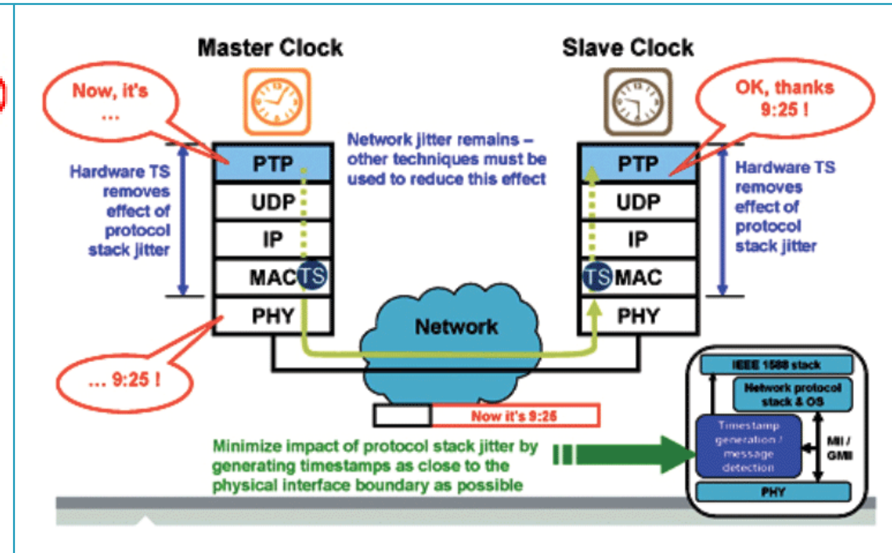
Message Latency and Hardware Timestamping

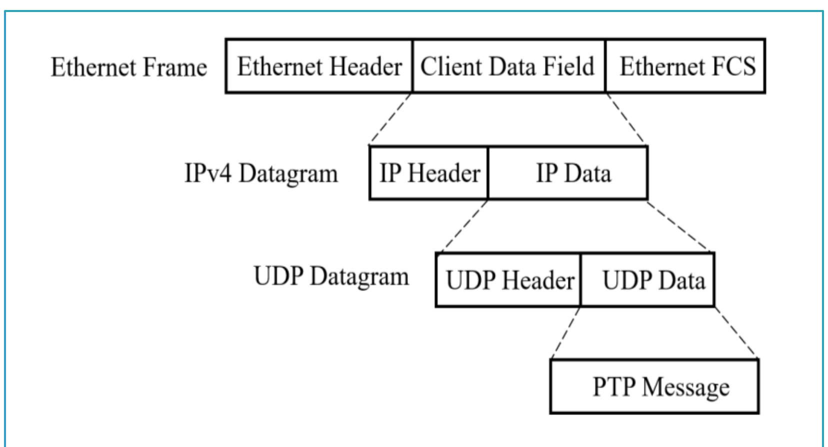


- Various non-deterministic latency components
- Latencies can be reduced by time stamping message transmission and reception events at lower levels in the communication hierarchy
  → **Hardware time stamping, as supported by PTP**

No Hardware Timestamping versus Hardware Timestamping





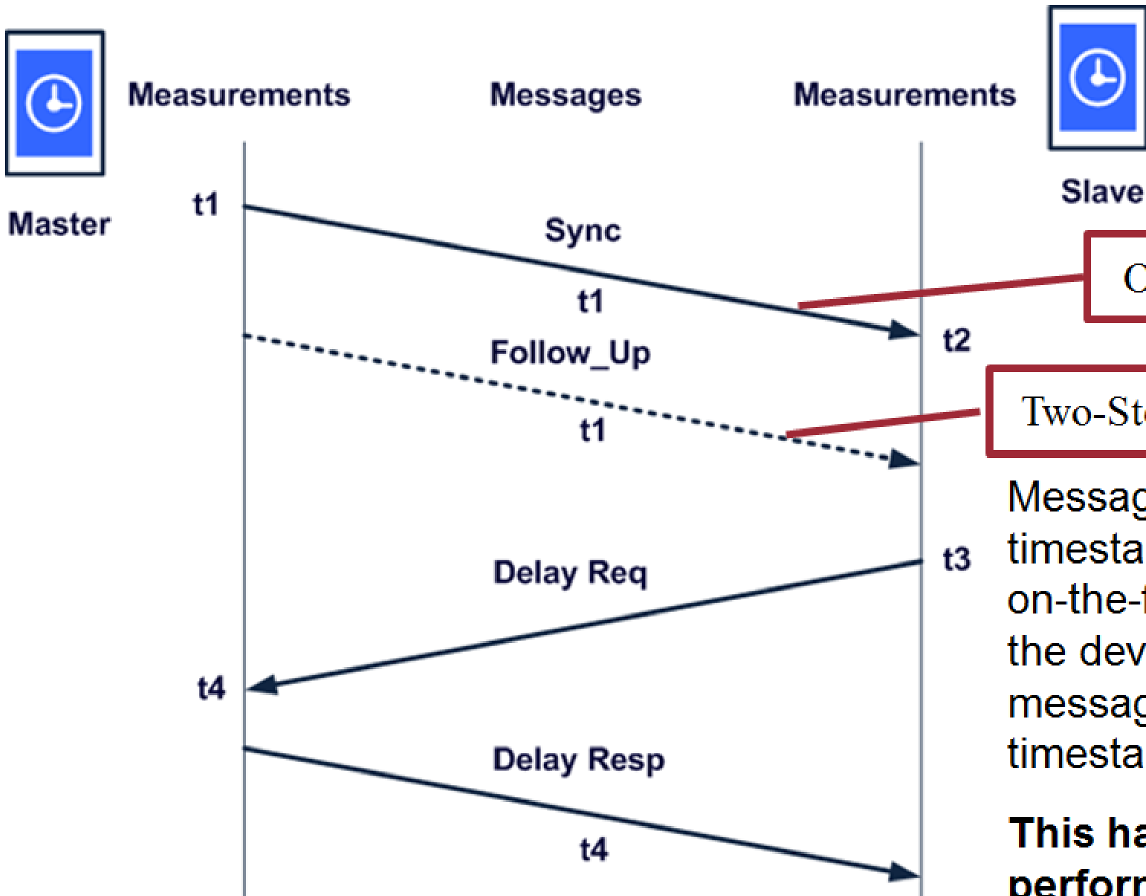


- A PTP aware NIC records the precise time when a packet is sent (left) or when it arrives (right)
- The latter is made available to the slave's PTP daemon directly
- The former has to be put into a PTP packet by the master:
  - In one-step mode the NIC manipulates the corresponding PTP message (i.e. adds the timestamp and corrects CRC) just before it is sent (see Ethernet frame structure)
  - In two-step mode the corresponding PTP message is sent without the timestamp, but directly followed by a second message that contains that timestamp

PTP Time Synchronisation Overview

- NTP is a typical client/server protocol with the client initiating a synchronisation message exchange
- PTP is not a typical client/server protocol, as it is the computer containing the time reference, i.e. the Master clock, to initiate a synchronisation cycle
- Here the master send out a multicast ***Sync*** (possibly followed by a ***Follow_Up***) message (one-step versus two-step mode) to the clients / slaves of a given domain
- This is followed by a series of unicast messages (→ next slide) initiated by each slave
  - The wording master/slave is widely used, but politically incorrect, so apologies
- Synchronisation messages that belong to the same cycle share the same sync sequence id *sequenceID*, a 16-bit counter that is incremented with each cycle

One-Step and Two-Step Operation



One-Step (t1 in Sync message)

Two-Step (t1 in Follow_Up message)

Message either gets an accurate timestamp placed in the message on-the-fly as it is about to leave the device (one-step), or a second message carries the accurate timestamp (two-step)

This has no implications on performance etc.

Both modi co-exist in a network

$$offset = ((T2 - T1) - (T4 - T3)) / 2$$
$$delay = ((T2 - T1) + (T4 - T3)) / 2$$

Offset and Delay Calculations in PTP

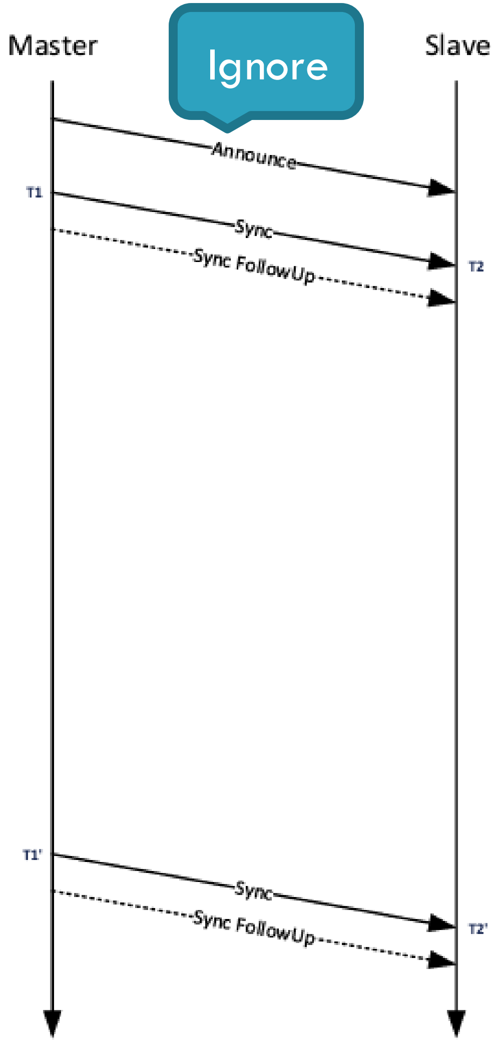
- Consider timestamps t1, t2, t3 and t4
- We have a symmetric network latency of D [ms]
- The master is +X [ms] ahead to the slave
- Offset calculation:
  (t2 – t1) – (t4 – t3) = (-X + D) – (X + D) = -2X, ergo
  Offset X = ((t2 – t1) – (t4 – t3)) / 2 (correct slave clock by X [ms])
- Delay calculation:
  (t2 – t1) + (t4 –t3) = (-X + D) + (X + D) = 2D, ergo
  Delay D = ((t2 – t1) – (t4 – t3)) / 2 (uplink + downlink)
- This measurement is repeated in defined intervals

Correcting a Slave's Clock Frequency

- Beside offset corrections PTP also supports frequency error corrections based on ***Sync*** message timestamps as follows:

$$\text{clock skew} = \frac{(T2' - T2) - (T1' - T1)}{(T1' - T1)}$$

- In order to compensate variations in transmission delays of ***Sync*** messages, consecutive skew values may be averaged over a sliding window
- These average values are subsequently used to adjust the slave clock's frequency
  - Both clocks will be syntonised (i.e. the time as measured by each advances at the same rate)



Some PTP Message Formats

Sync Message

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header								34	0
originTimestamp								10	34

Follow_Up Message

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header								34	0
preciseOriginTimestamp								10	34

Delay_Resp Message

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
header								34	0
receiveTimestamp								10	34
requestingPortIdentity								10	44

Matches identifier in corresponding Delay_Req message

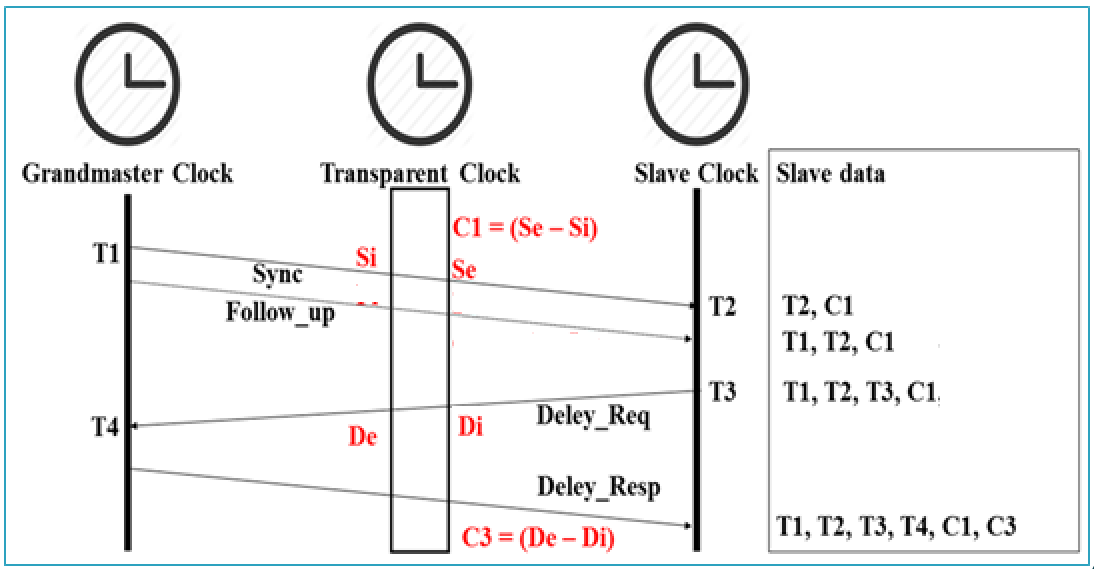
Common PTP Message Header

- Transparent clocks add the residence time of a given message to its ***CorrectionField*** that can be found in the PTP *common message header*

Bits								Octets	Offset
7	6	5	4	3	2	1	0		
transportSpecific/majorSdoId				messageType				1	0
reserved/minorVersionPTP				versionPTP				1	1
messageLength								2	2
domainNumber								1	4
reserved/minorSdoId								1	5
flagField								2	6
correctionField								8	8
reserved/messageTypeSpecific								4	16
sourcePortIdentity								10	20
sequenceId								2	30
controlField								1	32
logMessageInterval								1	33

Offset and Delay E2E (End-to-End) Calculation using *CorrectionField*

- C1 is the residence time of the ***Sync*** message in the TC, stored in the message's ***correctionField***
  - Two-step mode doesn't matter, as the residence time of the ***Follow_up*** message has no purpose
- C3 is the residence time of the ***Delay_Req*** message in the TC, stored again in ***correctionField***
- The slave incorporates ***correctionField*** values C1 and C3, when calculating delay and offset
- As a result, we only consider (fixed) signal propagation delays, but eliminate (variable) residence times of messages

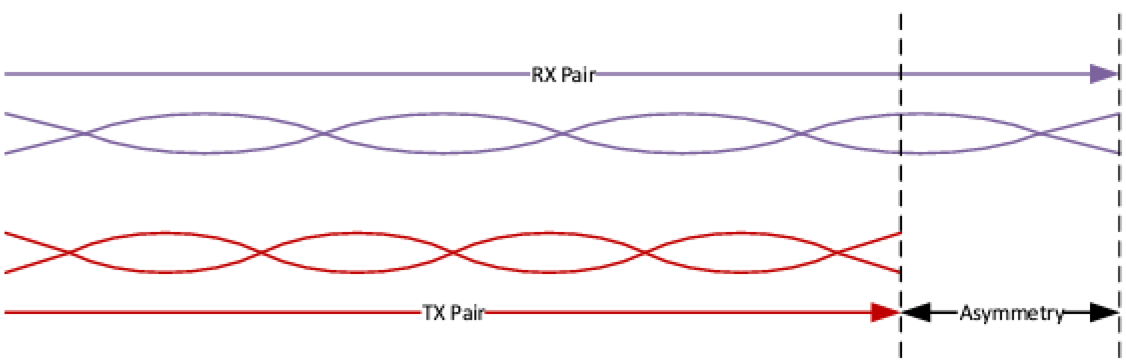


$$offset = ((T2 - T1 - C1) - (T4 - T3 - C3)) / 2$$
$$delay = ((T2 - T1 - C1) + (T4 - T3 - C3)) / 2$$

- ***Sync*** and potentially ***Follow_up messages*** provide T1, T2 and C1
- ***Delay_Req*** contains T3 when sent by the slave
- GM receives ***Delay_Req* (at time T4)** containing T3 as well as C3
- T3, T4 and C3 are copied into ***Delay_Resp*** which is sent back to the slave

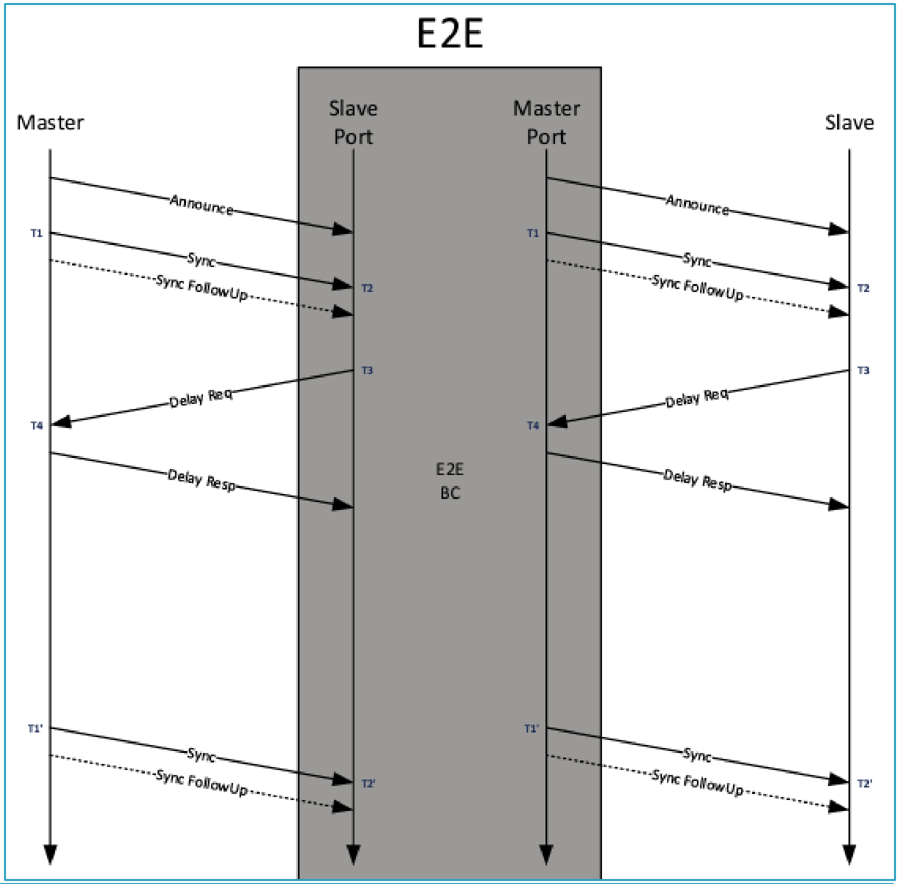
Issues with symmetrical Transmission Delays

- Now that we can accommodate variable residence times of PTP messages in TCs, exact time synchronisation should be achievable
- However, the still have to accommodate for symmetric transmission delays, i.e. <u>uplink and downlink cables need to have exactly the same length</u>
- Additionally, different twist rate of twisted line pairs leads to delays that impact on symmetry:
  - CAT 5/6 cables allow for up to 50 ns per 100 meter cable
  - CAT 7 cables allow for up to 30 ns per 100 meter cable



Boundary Clocks and their Operation in E2E Mode

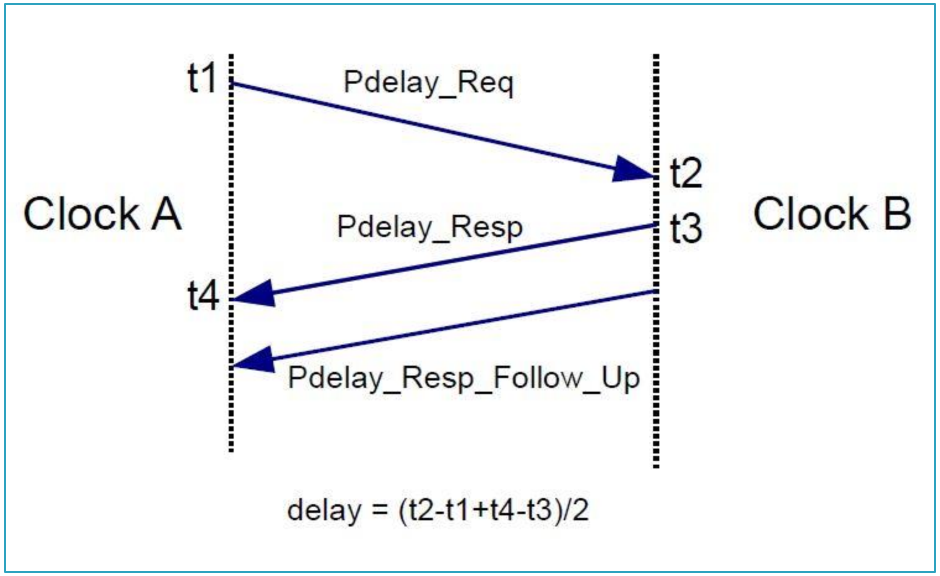
- A boundary clock (BC) is a network switch
- It has one port which is in a slave state, getting time from a (grand) master clock
- Multiple other ports are in a master state and synchronise downstream slaves
- Instead of tracking **Sync** messages and updating correction fields (as done in TCs), it
  - absorbs arriving ***Sync*** messages,
  - completes a synchronisation cycle as seen before to set its own clock, and
  - generates new ***Sync*** messages to be sent out of all of its master ports
- Note that a BC is not a GM, since its synchronise its own clock from an upstream grandmaster or boundary clock
- A BC can operate both in one-step or two-step mode
- Note that the ***Announce*** message above will be handled later, and has no function here



Boundary clocks ensure that PTP masters are not over-solicited, which greatly improves the synchronisation levels and system scalability

E2E versus P2P Delay Calculations

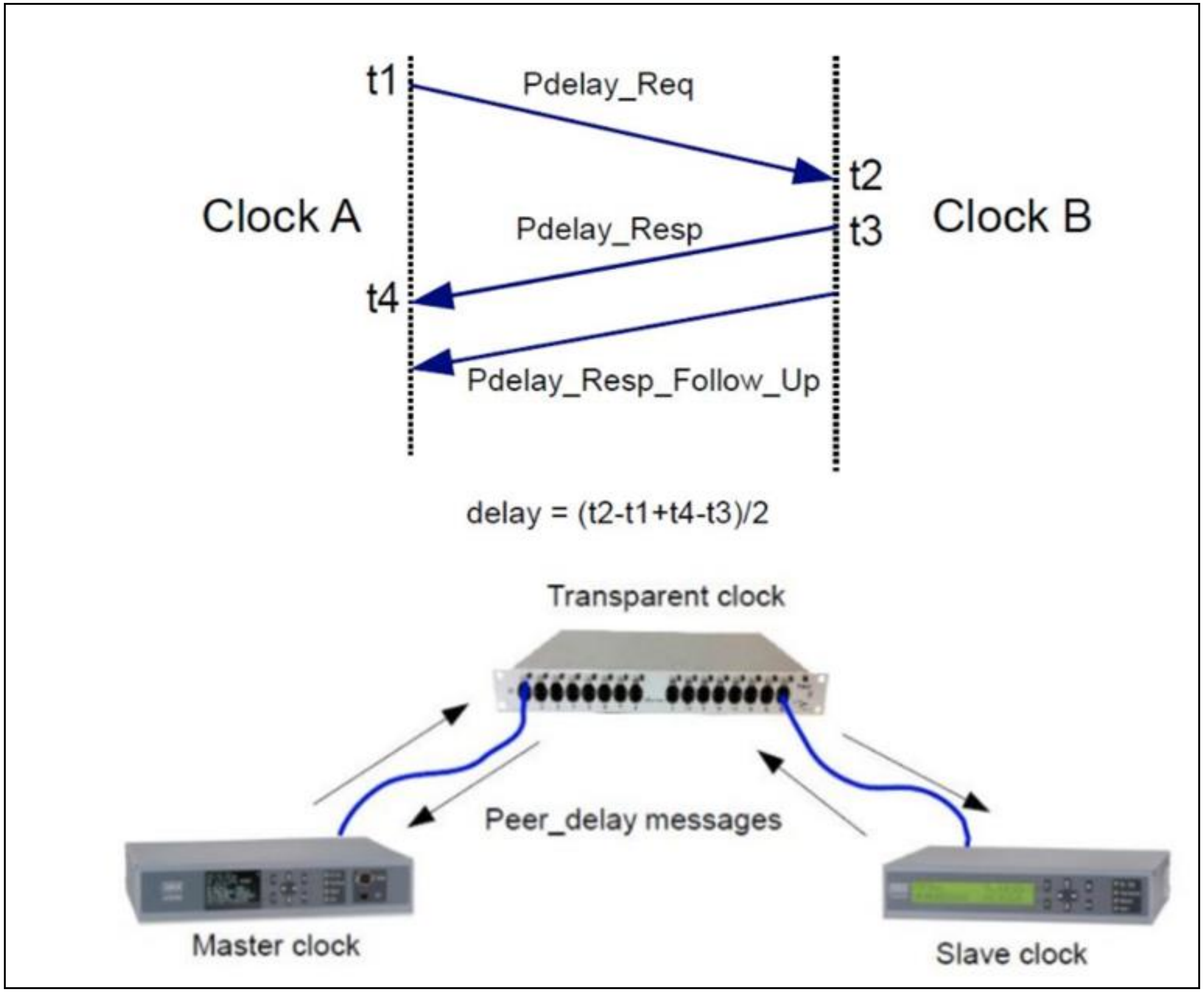
- As already seen in **End-to-End** mode the delay measurements take place between master and slave
- If a transparent clock is in the packet propagation path, ***correctionField*** will be updated
- However, E2E also works with non PTP-aware normal network switches



- **In Peer-to-Peer** mode all network equipment in the packet propagation path is PTP capable
- Beside *correctionField* calculations by TCs, network switches (i.e., TCs and BCs) do also calculate the delay to their direct uplink / downlink peers
- By doing so, the overall amount of network traffic, particularly traffic to be processed by the GM, can be greatly reduced
- The messages used to calculate the delay between 2 peers are shown in the diagram on the left
- The overall ***network delay*** between a GM and a slave clock is the sum of all P2P delays in the path, ergo slave time = master time + ***network delay***

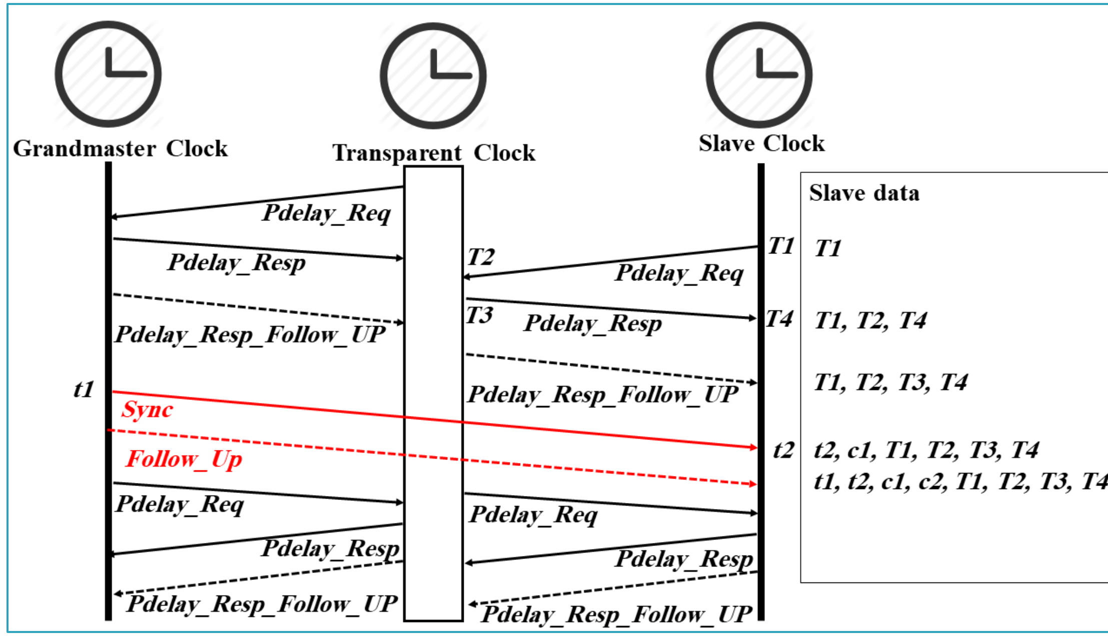
P2P Delay Mechanism

- Clock A initiates a P2P delay measurement, thereby acquiring t1 – t4
- Clock B may use one-step or two-step mode (as shown in the diagram) to send t3 back to Clock A
- In the diagram below the TC (and in fact all P2P-aware network infrastructure components) both send and receive *Pdelay_Req* messages to all their (uplink and downlink) neighbours



PTP Timestamps and Time Synchronisation Messages in P2P Delay Mechanism

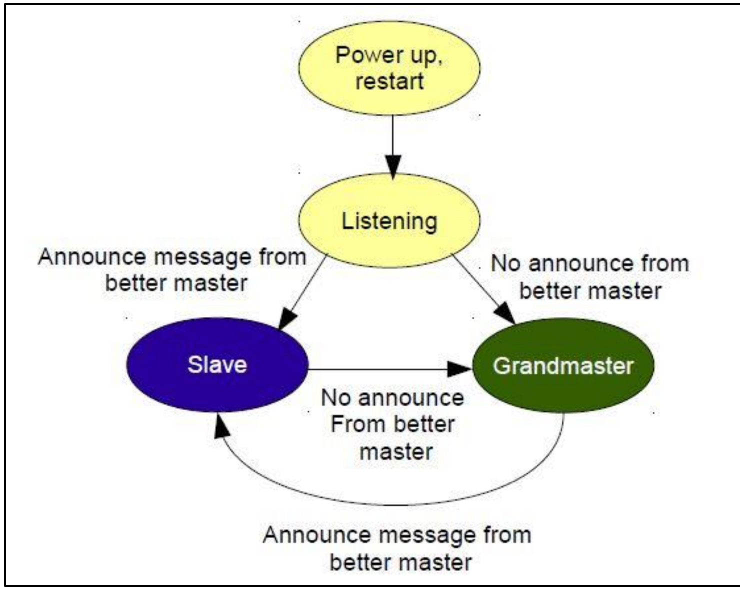
- In the diagram GM, TC and SC do (uplink/downlink) P2P delay measurements using ***Pdelay****_** messages in two-step mode
- As a result every node keeps track on P2P delays to their direct up-stream/downstream neighbour
- ***Sync*** / ***Follow_Up*** are GM broadcast messages containing T1
- The ***Sync*** message's ***correctionField*** is updated by the TC (not shown)



- All P2P delays along the message path from GM to SC need to be added to calculate the network delay
- Therefore, each **Sync** message is amended when passing a BC/TC, by adding the P2P delay between itself and the next hop upstream (i.e., the GM in the diagram) as well as the packet residence time to the *CorrectionField*
- Finally, the SC calculates its offset using t1, t2, the ***CorrectionField*** value Cx in the ***Sync*** message, and the delay (Pdelay) between SC and the previous hop (i.e. the TC): Offset = t2 − t1 − Cx – Pdelay

The Best Master Clock Algorithm (BCMA)

- After power up all ordinary clocks determine which one becomes the grandmaster
- Each clock sends out multicast ***Announce*** messages (see earlier diagrams), which contain the properties (next slide) of the clock
- If an ordinary clock sees an ***Announce*** message from a better clock, it goes into a slave state, or passive if it is not slave capable (i.e. if it is a redundant GM)



- If the Ordinary Clock does not see an ***Announce*** message from a better clock within the ***Announce Time Out Interval***, then it takes over the role of grandmaster
- This process runs continuously, so master-capable devices are constantly on the lookout for the possible loss of the current master clock
  - If the GM does not send ***Announce*** messages within ***Announce Timeout Interval***, slave clocks assume it is not operational anymore and the selection process start all over again; this provides **redundancy**

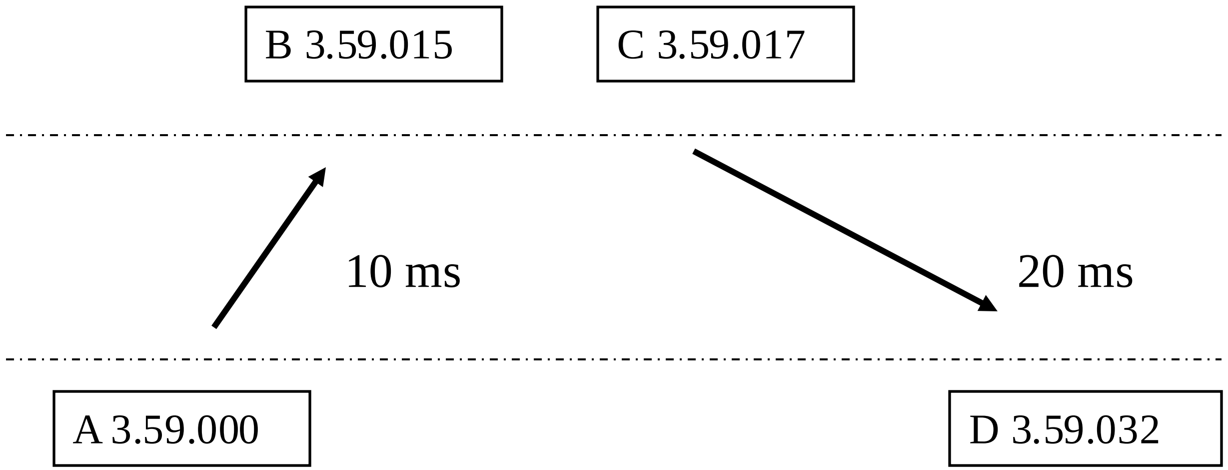
The Best Master Clock Algorithm (BCMA)

- Clock attributes in ***Announce*** messages are evaluated in a decision tree in the following order:
  - **Priority One Field**: An 8-bit user settable value, the lowest number wins
  - **Clock Class**: An enumerated list describing the quality of UTC time reference (e.g. GPS receiver versus free-running clock)
  - **Clock Accuracy**: An enumerated list of ranges of clock skews
  - **Clock Variance**: Characterises the clock drift
  - **Priority 2 Field**: A user settable field, mainly used to identify primary and backup clocks among identical redundant grandmasters
  - **Source Port ID**: A unique number (i.e. the Ethernet MAC address) used to break a tie

FYI: What makes PTP so vulnerable to cyberattacks

- PTP is widely used for time synchronisation of critical infrastructure and financial institutions
  - Attacks on synchronisation would have wide-reaching impact
- However, PTP is vulnerable to attacks by adversaries, as:
  - PTP is an unprotected protocol
  - PTP time synchronisation is based on a single grandmaster
  - PTP required a well-managed network with symmetric uplink/downlink protocols

Recap: Impact of Network Asymmetry on Offset Calculation



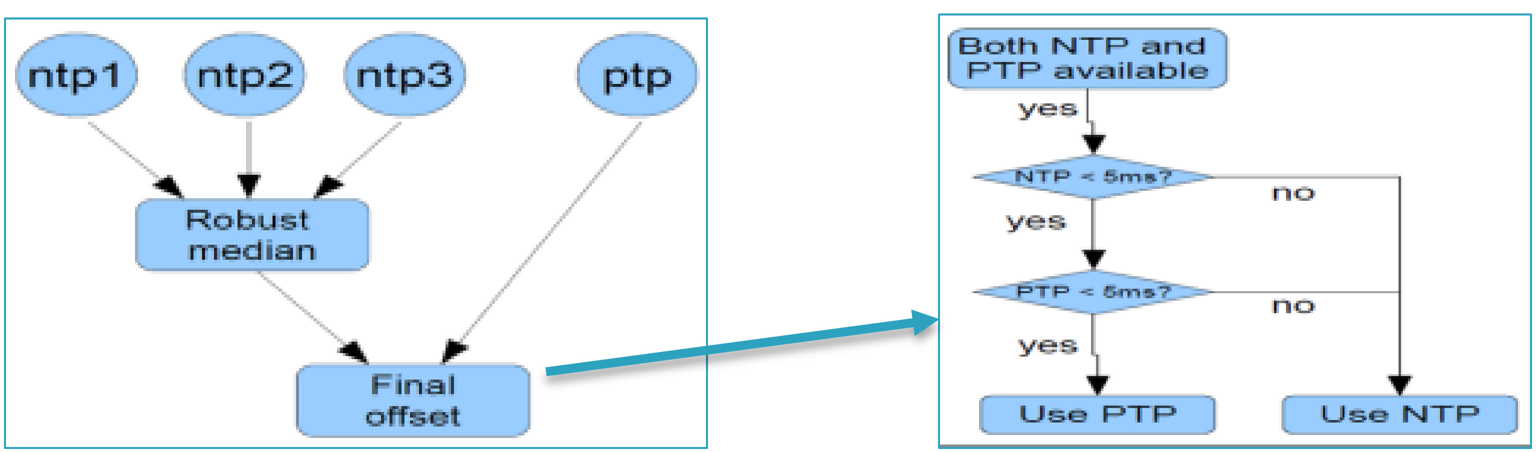
Offset still 5 ms but Asymmetric Network

RTD = (D - A) – (C – B) = 32 – 2 = 30 msec

Offset = ½[(B-A) - (D-C)] = (15 – 15)/2 = 0 ms .. **Error**

FYI: Increasing Time Synchronisation Robustness via Protocol Redundancy

- Based on work by Estrella et al, published in "Using a multi-source NTP watchdog to increase the robustness of PTPv2 in Financial Industry networks"
- Here a slave clock runs both NTP (using multiple stratum time sources) and PTP (using a single GM reference)
- Both calculate an offset which is further processed using the decision tree on the right:
  - If NTP calculates a clock offset in relation to UTC larger than a threshold, (i.e. > 5 ms), then NTP takes full control of the clock by applying its own offset, and the offset calculated by PTP is ignored
  - Otherwise the PTP offset is checked too; only if both PTP and NTP determine offsets below that threshold, PTP is allowed to control the clock

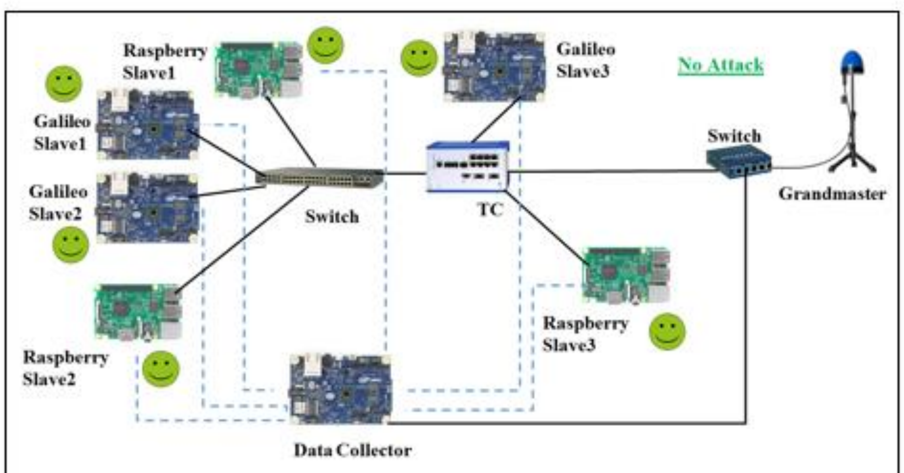


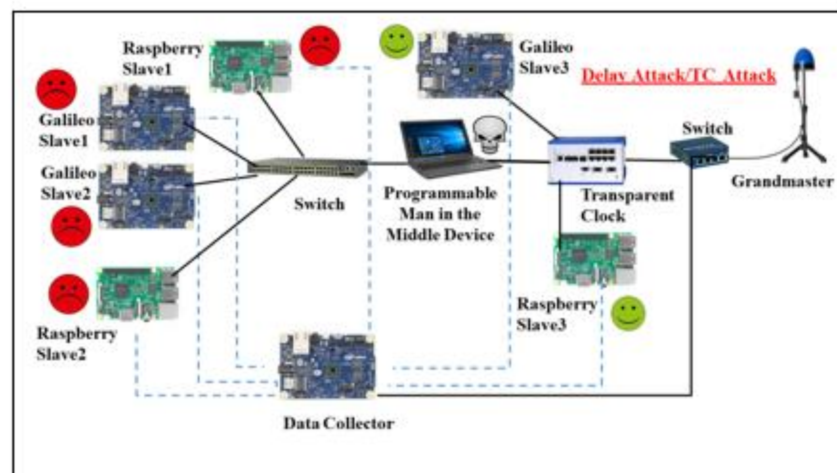
FYI: Simulation of PTP Cyberattacks

- An internal attacker based within a network switch, TC, BC, GM or OC can manipulate PTP messages, e.g. timestamps, and compromise time synchronisation stealthily
- In the example below, a Man-in-the-Middle (MitM) attacker is positioned within a PTP

PTP Testbed

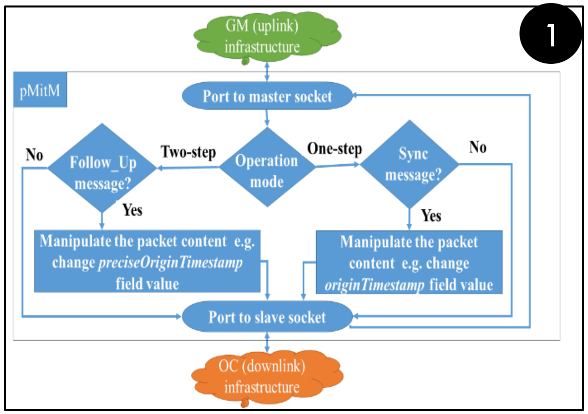
PTP Testbed with MitM attacker

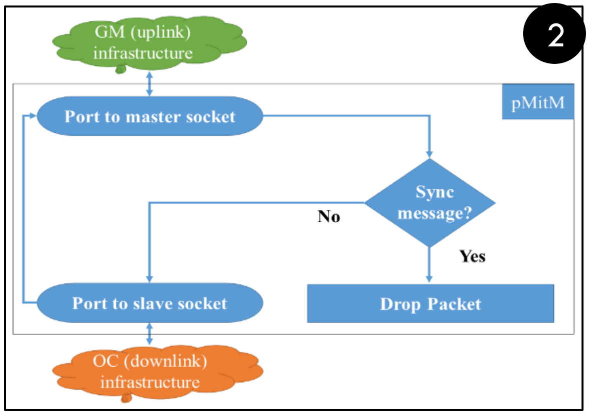


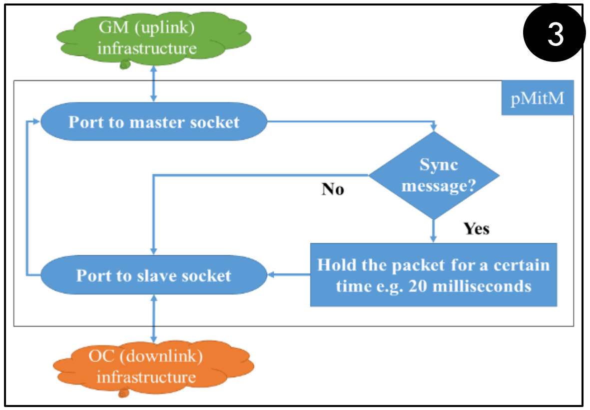


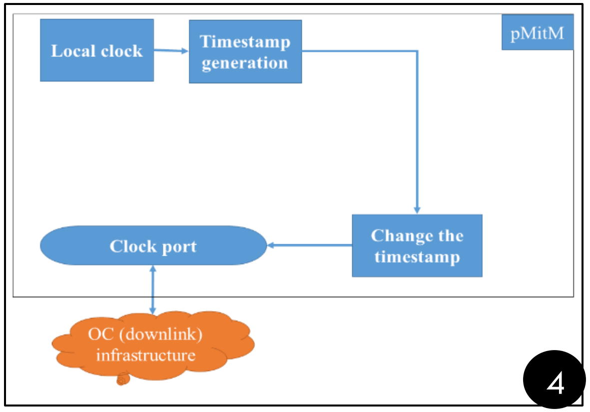
FYI: PTP Attack Strategies

NO	Attack Strategy	Attack Location
1	Packet Content Manipulation	Hub/Switch/TC/Router
2	Packet Removal	Hub/Switch/TC/Router
3	Packet Delay Manipulation	Hub/Switch/TC/Router
4	Time Source Degradation	GM/BC
5	Master Spoofing	OC/Hub/Switch/TC/Router
6	Slave Spoofing	OC/Hub/Switch/TC/Router
7	Compromised BMCA	OC/BC
8	Packet Replay	Hub/Switch/TC/Router
9	Denial of Service	OC/Hub/Switch/TC/Router



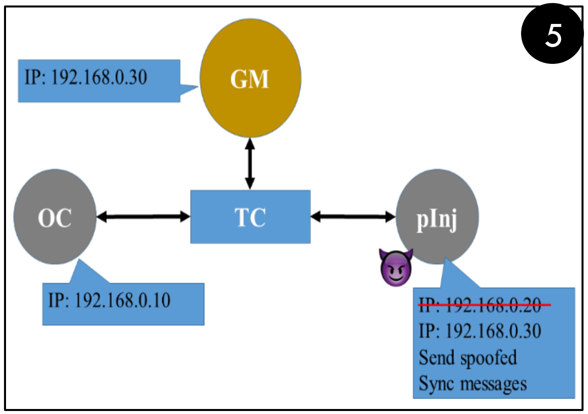


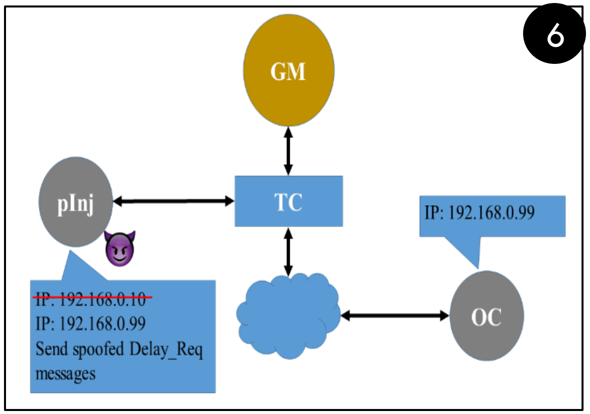


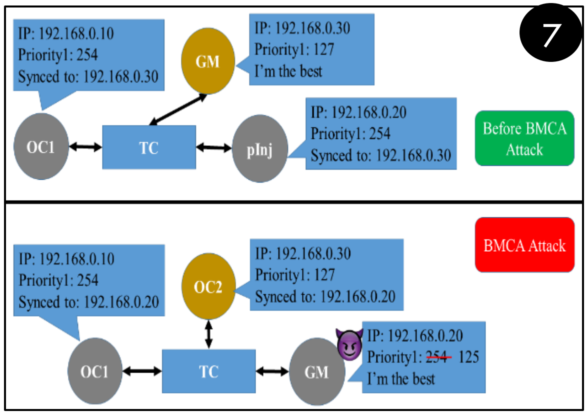


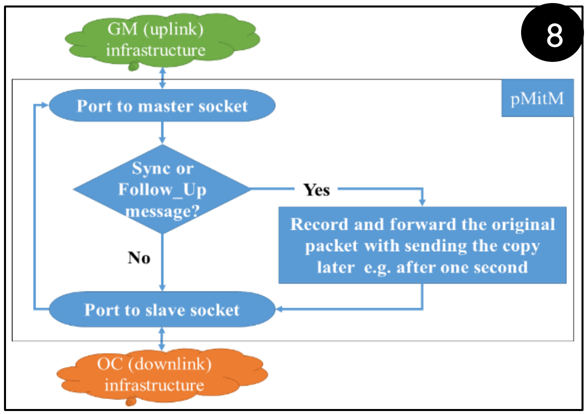
FYI: PTP Attack Strategies

NO	Attack Strategy	Attack Location
1	Packet Content Manipulation	Hub/Switch/TC/Router
2	Packet Removal	Hub/Switch/TC/Router
3	Packet Delay Manipulation	Hub/Switch/TC/Router
4	Time Source Degradation	GM/BC
5	Master Spoofing	OC/Hub/Switch/TC/Router
6	Slave Spoofing	OC/Hub/Switch/TC/Router
7	Compromised BMCA	OC/BC
8	Packet Replay	Hub/Switch/TC/Router
9	Denial of Service	OC/Hub/Switch/TC/Router

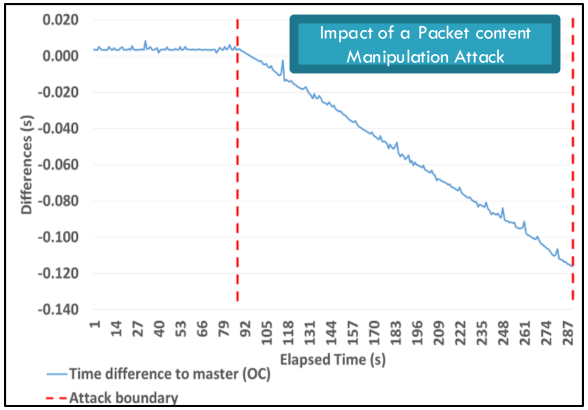


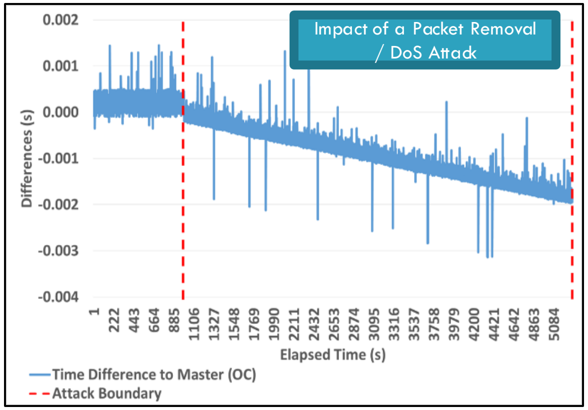


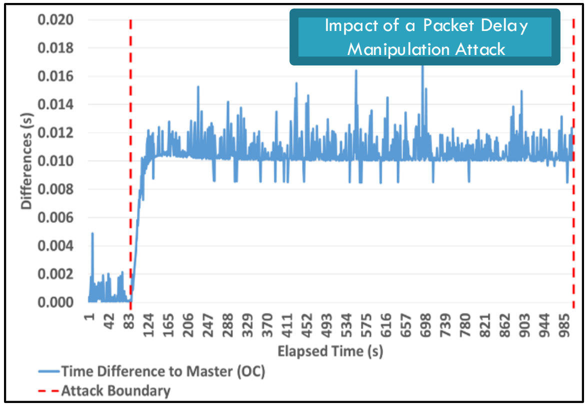


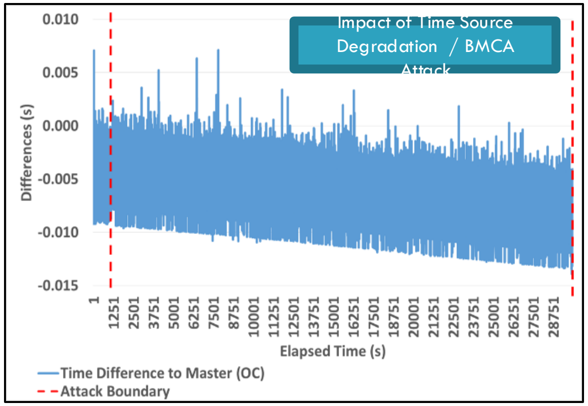


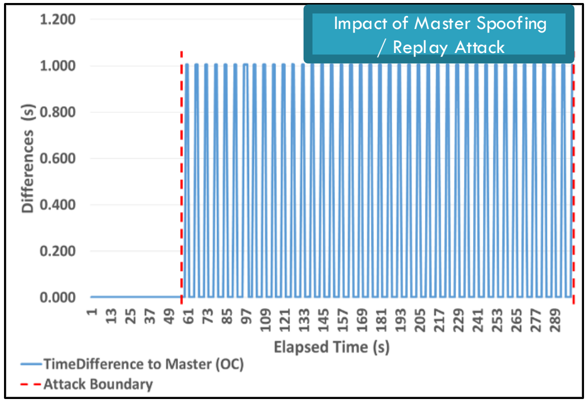
FYI: PTP Attack Impact











Summary

- PTP is a very sophisticated protocol designed for very precise clock synchronisation in well-designed and managed LAN
- In contrast to NTP it relies on a single grandmaster as time reference
- It works best with PTP-aware hardware (i.e. NIC, TC and BC) that allow hardware timestamping and the calculation of packet residence times
- However, much more than NTP, PTP is vulnerable to cyberattacks or equipment failures, as it relies on a single GM