Preventing (Network) Time Travel with Chronos

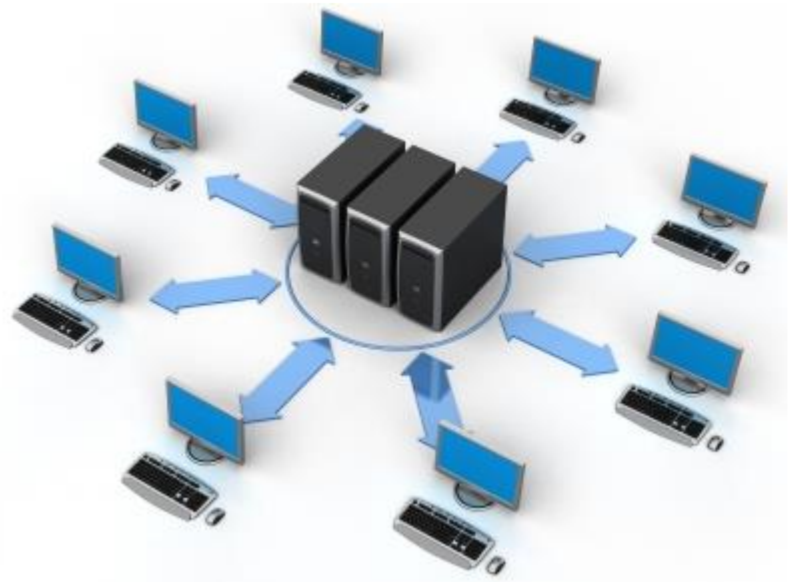
Omer Deutsch, **Neta Rozen Schiff**, Danny Dolev, Michael Schapira

THE HEBREW UNIVERSITY OF JERUSALEM

Network Time Protocol (NTP)

- NTP synchronizes time across computer systems over the Internet.

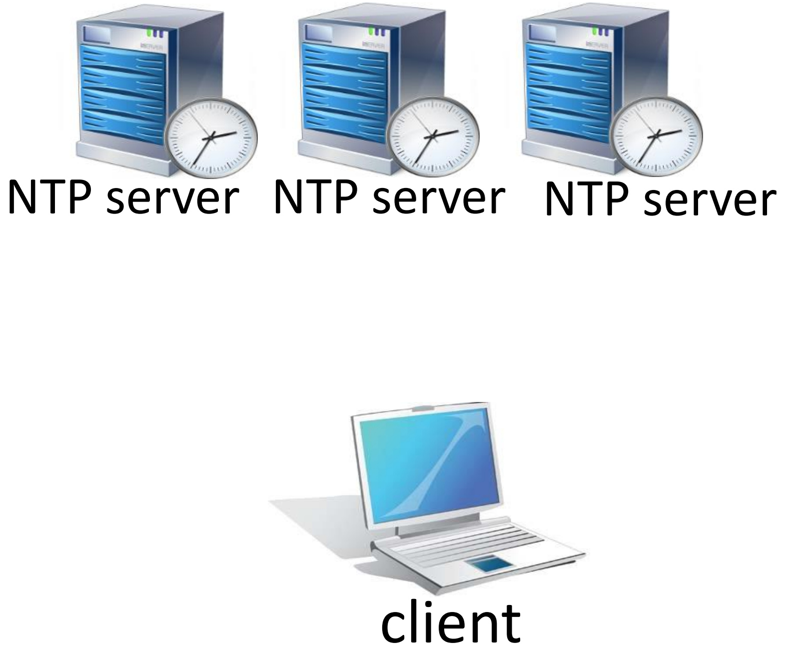
- Many applications rely on NTP for correctness and safety:
    - TLS certificates
    - DNS (and DNSSEC)
    - HTTPS
    - Kerberos
    - Financial applications



NTP Architecture

- NTP's client-server architecture consists of two main steps:

1. **<u>Poll process</u>**:
   The NTP client gathers time samples from NTP servers

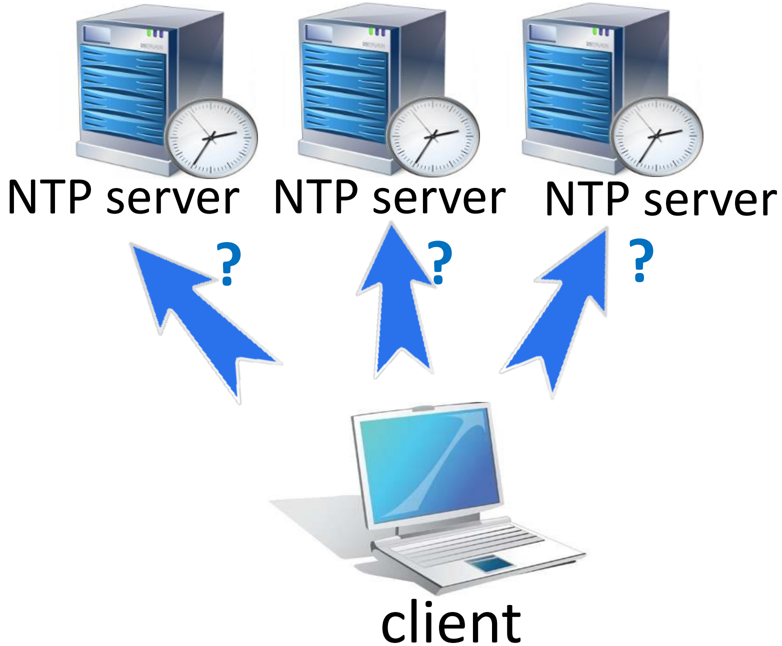


NTP Architecture

- NTP's client-server architecture consists of two main steps:

1. **<u>Poll process</u>**:
   The NTP client gathers time samples from NTP servers

Poll process: **NTP queries**

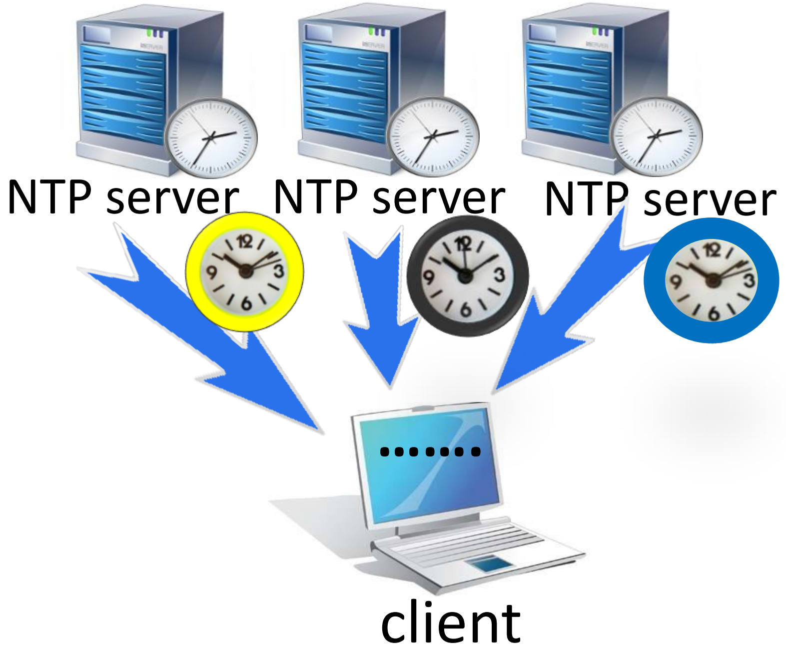


NTP Architecture

- NTP’s client-server architecture consists of two main steps:

  1. **<u>Poll process</u>**:
     The NTP client gathers time samples from NTP servers

Poll process: **NTP responses:**



NTP Architecture

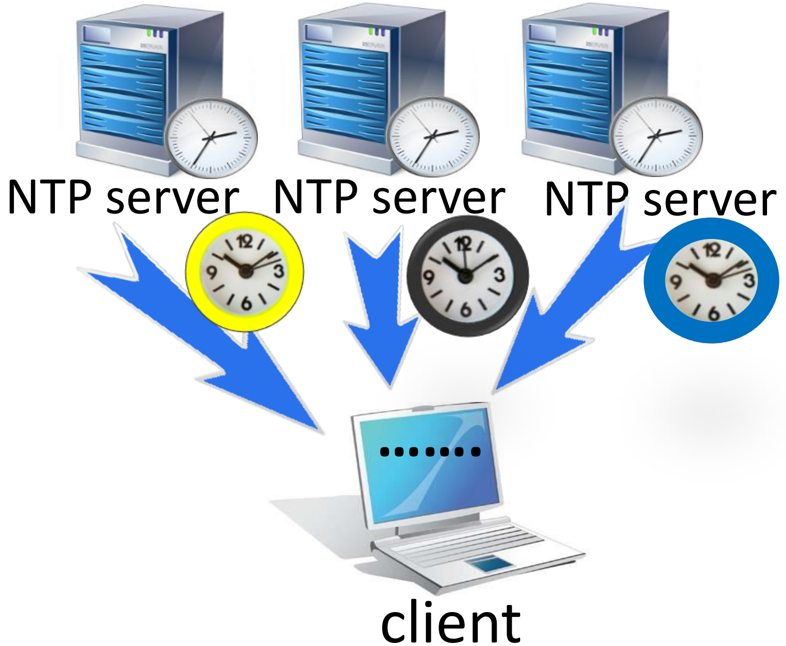
- NTP's client-server architecture consists of two main steps:

1. **<u>Poll process</u>**:
   The NTP client gathers time samples from NTP servers

2. **<u>Selection process</u>**:
   The "best" time samples are selected
   and are used to update the local clock

Poll process: **NTP responses:**

Selection process:



NTP Architecture

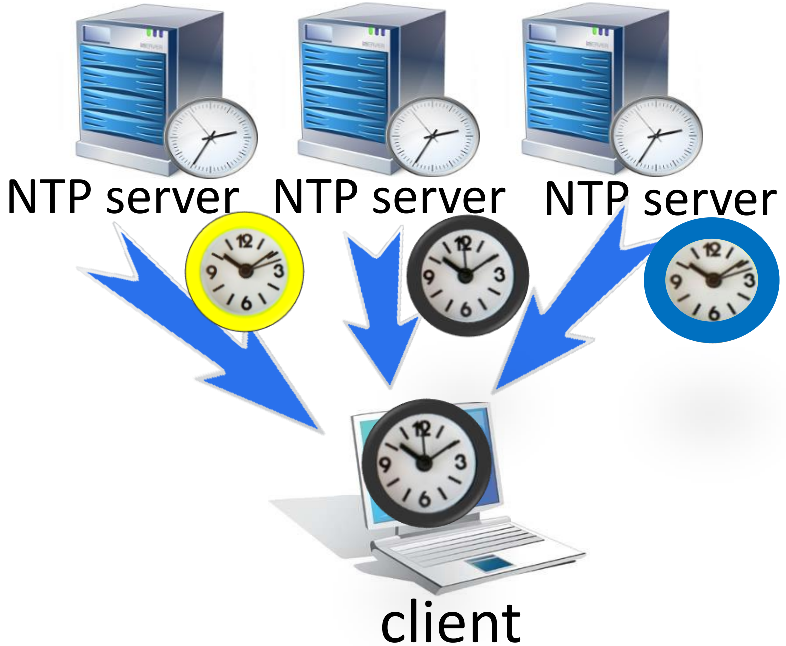
- NTP’s client-server architecture consists of two main steps:

1. **<u>Poll process</u>**:
   The NTP client gathers time samples from NTP servers

2. **<u>Selection process</u>**:
   The “best” time samples are selected
   and are used to update the local clock

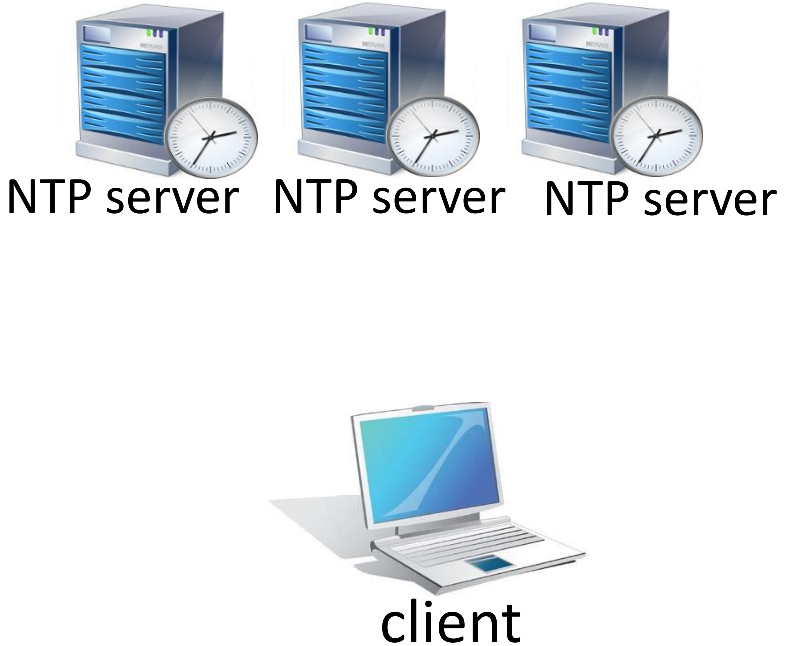
Poll process: **NTP responses:**

Selection process:



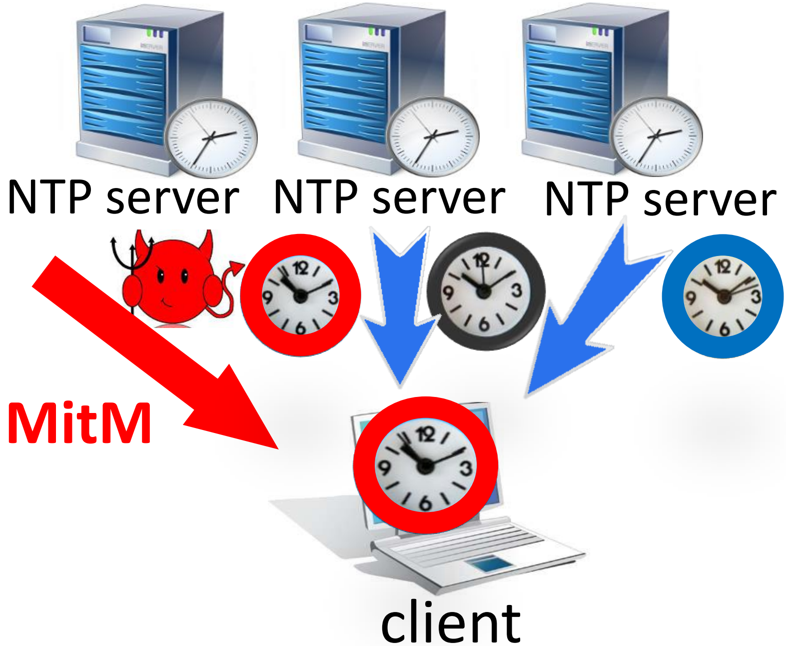
NTP Man-in-the-Middle (MitM) Attack

- NTP is highly vulnerable to time shifting attacks, especially by a MitM attacker



NTP Man-in-the-Middle (MitM) Attack

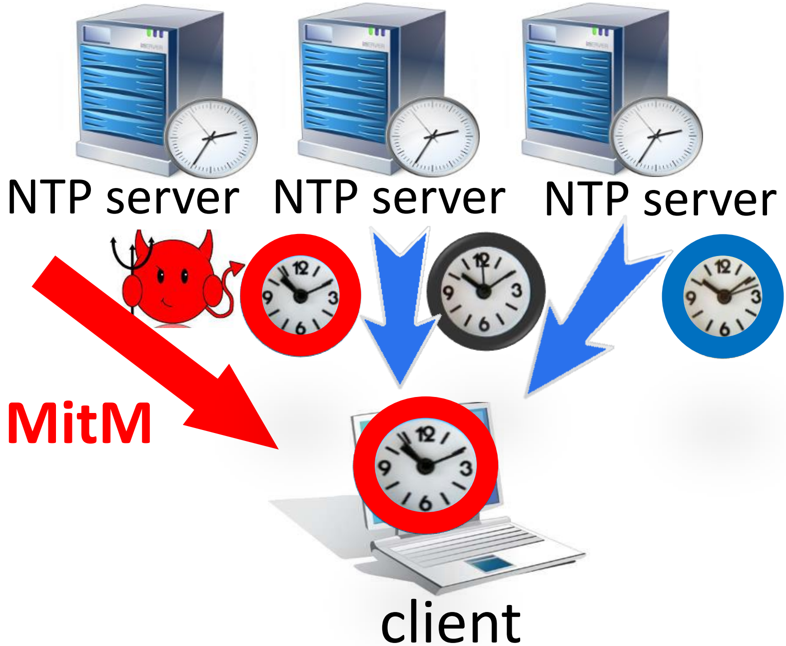
- NTP is highly vulnerable to time shifting attacks, especially by a MitM attacker



NTP Man-in-the-Middle (MitM) Attack

- NTP is highly vulnerable to time shifting attacks, especially by a MitM attacker
    - Can tamper with NTP responses
    - Can impact local time at client simply by dropping and delaying packets to/from servers **(encryption and authentication are insufficient)**

- Previous studies consider MitM as "too strong for NTP"



Why is NTP so Vulnerable to MitM?

- **<u>NTP's poll process</u>** relies on a small set of NTP servers (e.g., from pool.ntp.org), and this set is often DNS-cached (implementation property).

Why is NTP so Vulnerable to MitM?

- **<u>NTP's poll process</u>** relies on a small set of NTP servers (e.g., from pool.ntp.org), and this set is often DNS-cached (implementation property).

Attacker only needs MitM capabilities with respect to few NTP servers

Why is NTP so Vulnerable to MitM?

- **NTP's poll process** relies on a small set of NTP servers (e.g., from pool.ntp.org), and this set is often DNS-cached (implementation property).

Attacker only needs MitM capabilities with respect to few NTP servers

- **NTP's selection process** assumes that inaccurate sources are rare and fairly well-distributed around the UTC (the correct time)

Why is NTP so Vulnerable to MitM?

- **<u>NTP's poll process</u>** relies on a small set of NTP servers (e.g., from pool.ntp.org), and this set is often DNS-cached (implementation property).

Attacker only needs MitM capabilities with respect to few NTP servers

- **<u>NTP's selection process</u>** assumes that inaccurate sources are rare and fairly well-distributed around the UTC (the correct time)

Powerful and sophisticated MitM attackers are beyond the scope of **<u>traditional</u>** threat models

Chronos to the Rescue

The **Chronos NTP client** is designed to achieve the following:

- **Provable security** in the face of fairly powerful MitM attacks
  - ➢ negligible probability for successful timeshifting attacks

Chronos to the Rescue

The **Chronos NTP client** is designed to achieve the following:

- **Provable security** in the face of fairly powerful MitM attacks
  - ➢ negligible probability for successful timeshifting attacks
- **Backwards-compatibility**
  - ➢ no changes to NTP servers
  - ➢ limited software changes to client

Chronos to the Rescue

The **Chronos NTP client** is designed to achieve the following:

- **Provable security** in the face of fairly powerful MitM attacks
  - ➢ negligible probability for successful timeshifting attacks
- **Backwards-compatibility**
  - ➢ no changes to NTP servers
  - ➢ limited software changes to client
- **Low computational and communication overhead**
  - ➢ query few NTP servers

Threat Model

The attacker:

- Controls a large fraction of the NTP servers in the pool (say, ¼)
- Capable of both modifying the content of NTP responses **<u>and</u>** timing when responses arrive at the client
- Malicious

Chronos Architecture

Chronos' design combines several ingredients:

- **Relying on many NTP servers**
  - ➢ Generates a large server pool (hundreds) per client
    - ➢ E.g., by repeatedly resolving NTP pool hostnames and storing returned IPs
  - ➢ Sets a very high threshold for a MitM attacker

Chronos Architecture

Chronos' design combines several ingredients:

- **Relying on many NTP servers**
  - ➢ Generates a large server pool (hundreds) per client
    - ➢ E.g., by repeatedly resolving NTP pool hostnames and storing returned IPs
  - ➢ Sets a very high threshold for a MitM attacker
- **Querying few servers**
  - ➢ Randomly queries a small fraction of the servers in the pool (e.g., 10-20)
  - ➢ Avoids overloading NTP servers

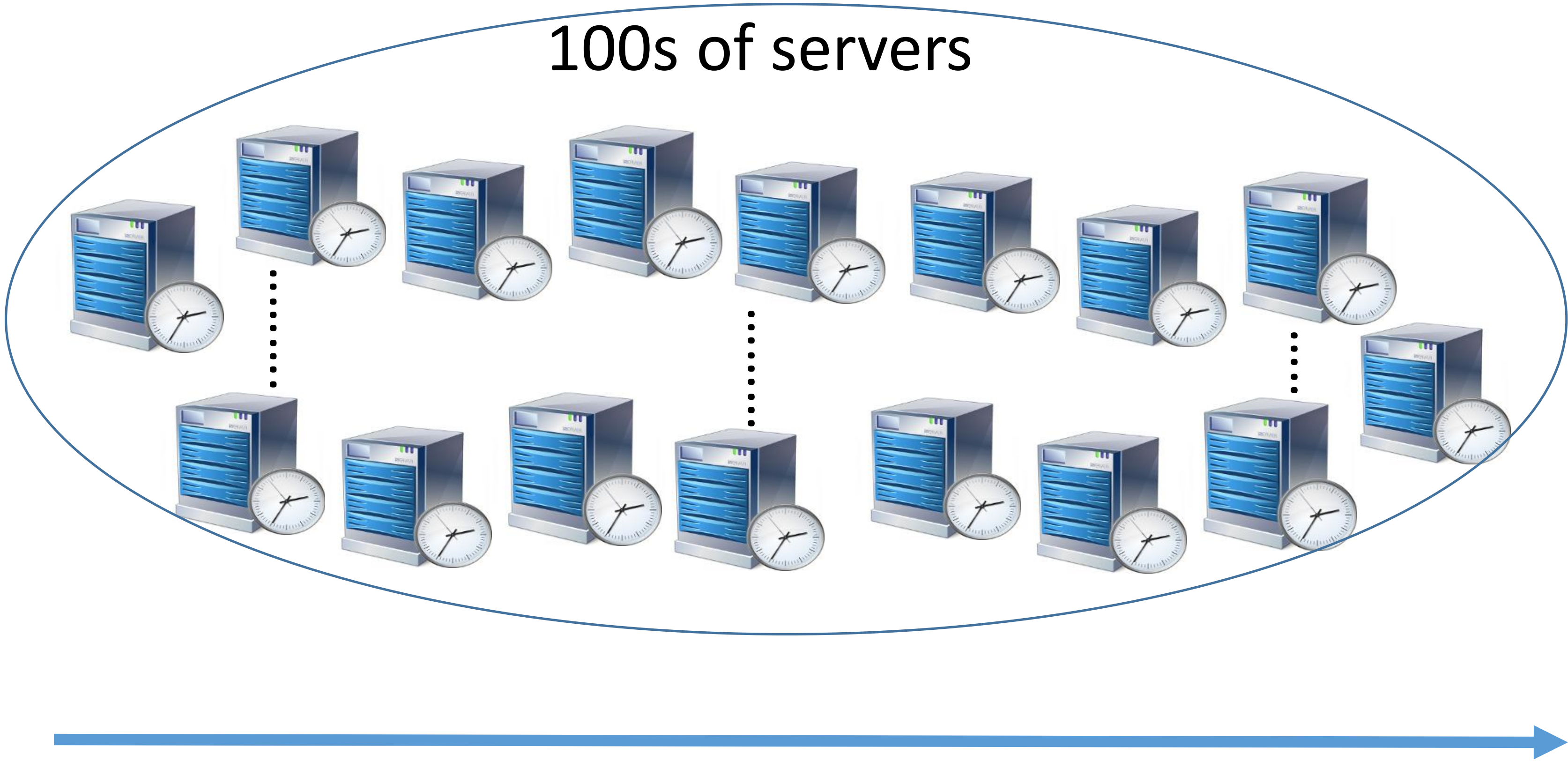
Chronos Architecture

Chronos' design combines several ingredients:

- **Relying on many NTP servers**
  - ➢ Generates a large server pool (hundreds) per client
    - ➢ E.g., by repeatedly resolving NTP pool hostnames and storing returned IPs
  - ➢ Sets a very high threshold for a MitM attacker
- **Querying few servers**
  - ➢ Randomly queries a small fraction of the servers in the pool (e.g., 10-20)
  - ➢ Avoids overloading NTP servers
- **Smart filtering**
  - ➢ Removes outliers via a technique used in approximate agreement algorithms
  - ➢ Limits the MitM attacker's ability to contaminate the chosen time samples

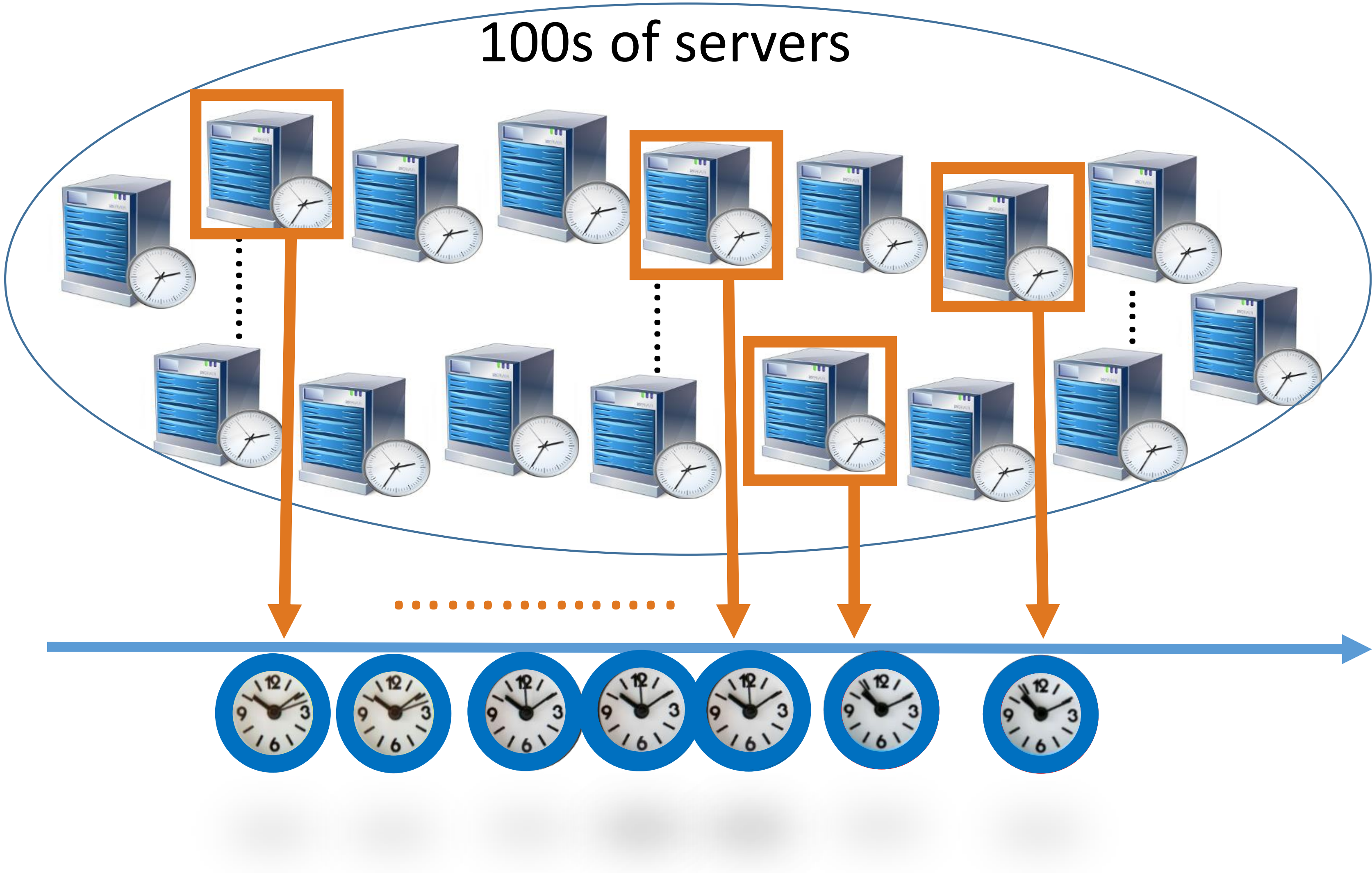
Chronos' Time-Update Algorithm: Informal

- Query m (10s of) servers **<u>at random</u>**



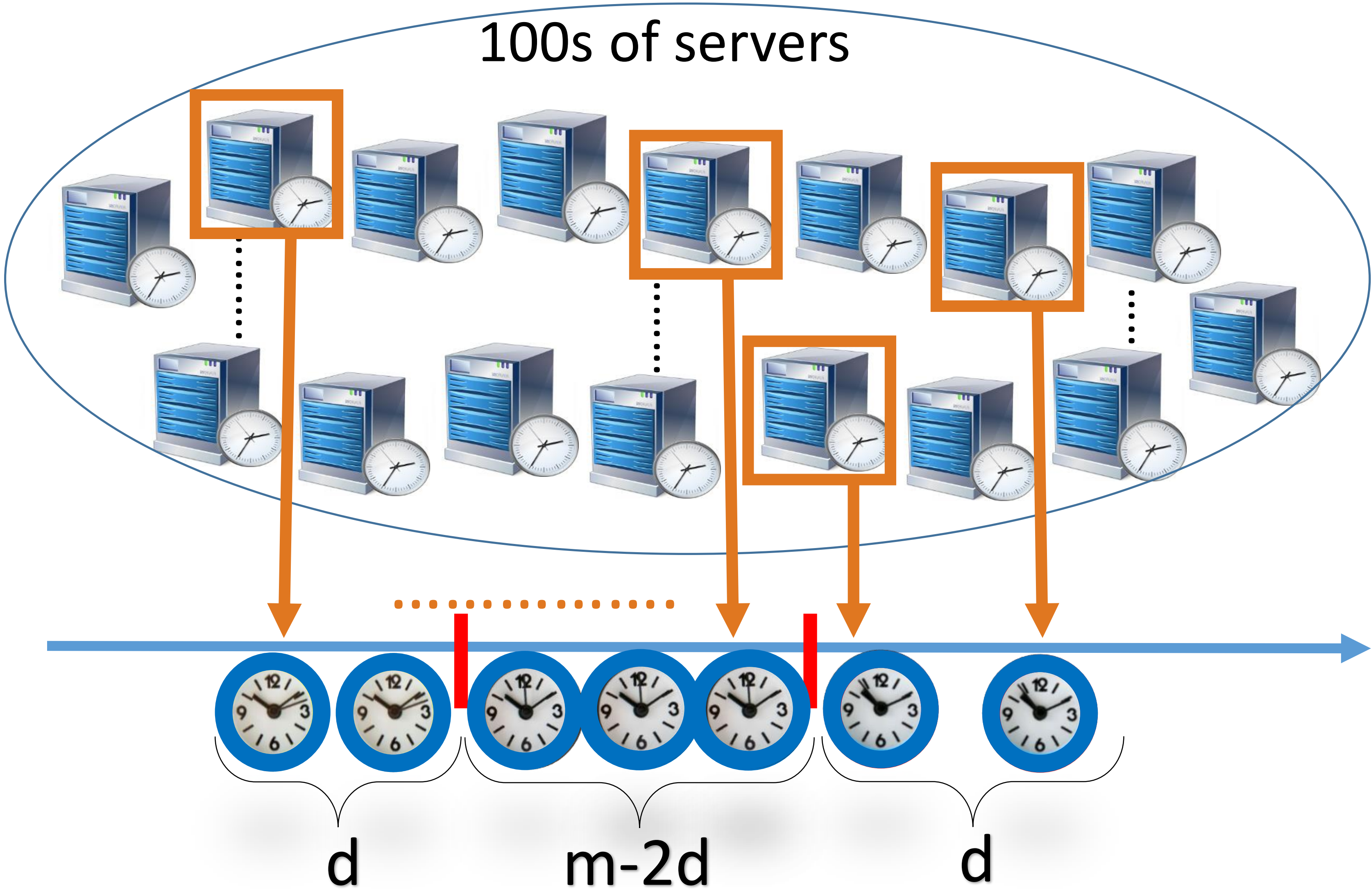
Chronos' Time-Update Algorithm: Informal

- Query m (10s of) servers **<u>at random</u>**
- Order time samples from low to high



Chronos' Time-Update Algorithm: Informal

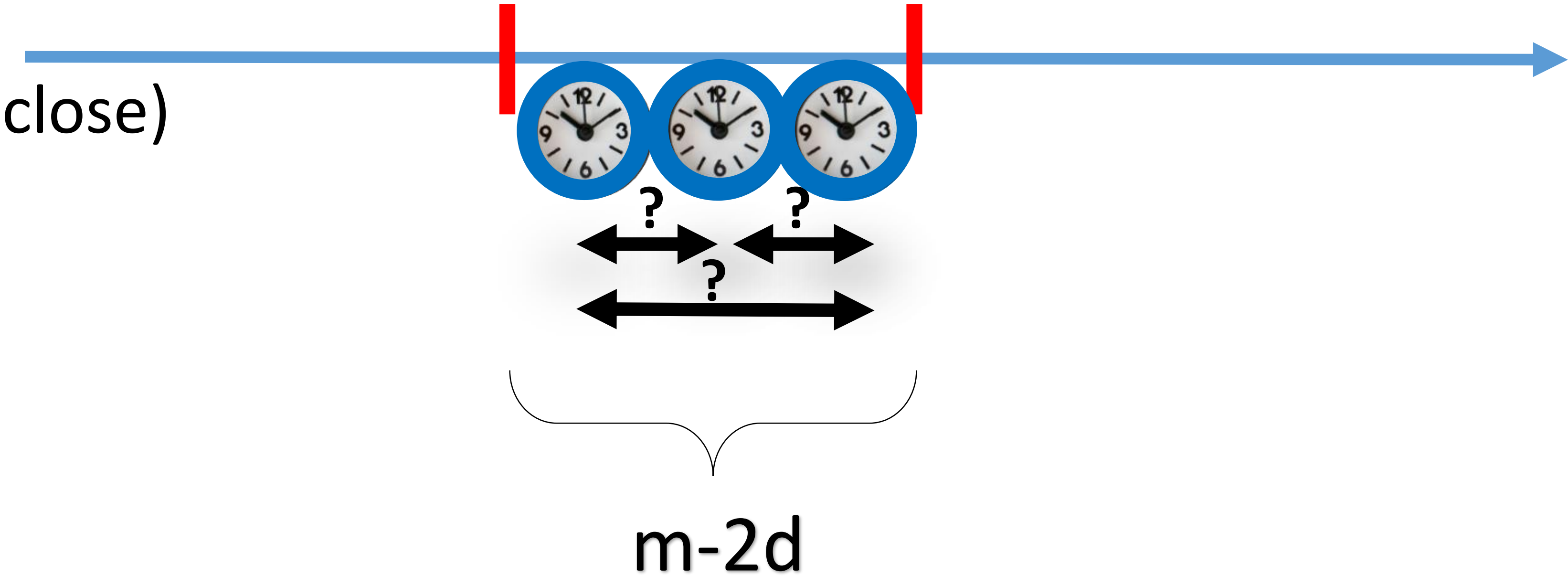
- Query m (10s of) servers **<u>at random</u>**
- Order time samples from low to high
- Remove the d lowest and highest time samples



Chronos' Time-Update Algorithm: Informal

Check:

If (the remaining samples are close)

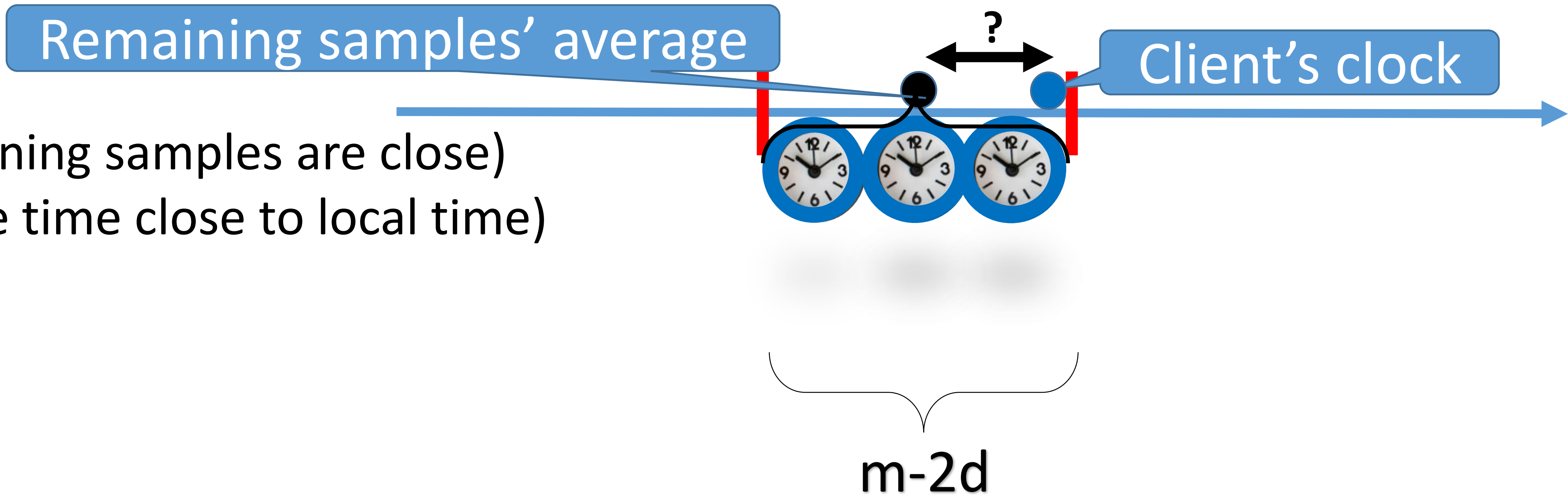


Chronos' Time-Update Algorithm: Informal

Check:

If (the remaining samples are close)

and (average time close to local time)



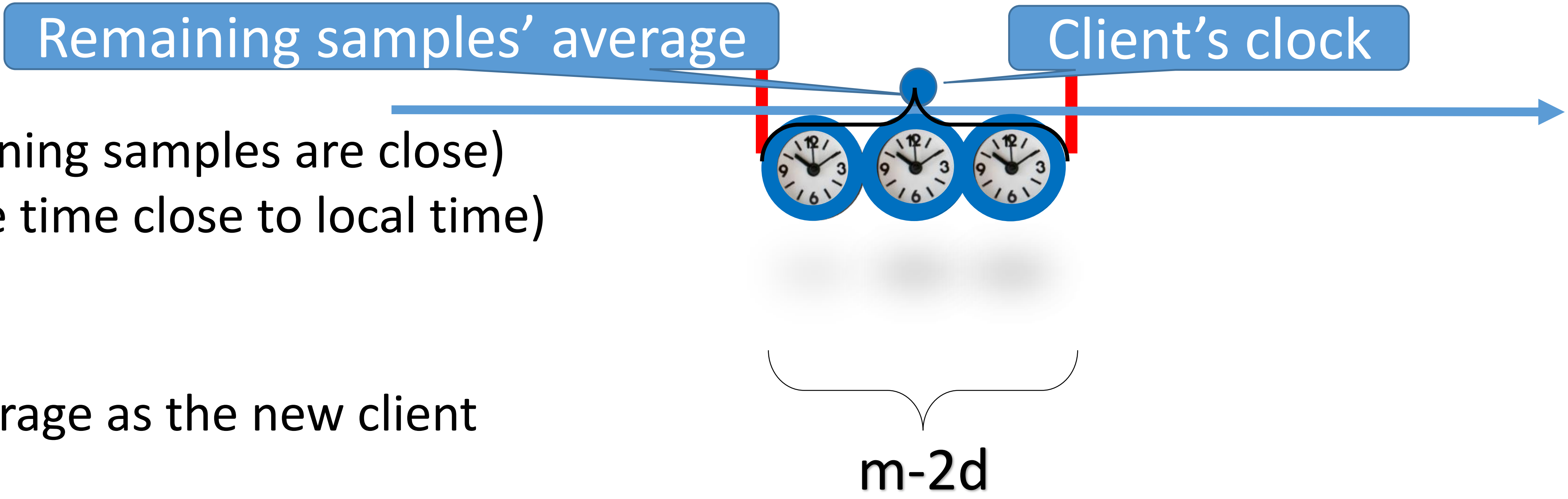
Chronos' Time-Update Algorithm: Informal

Check:

If (the remaining samples are close)

and (average time close to local time)

- **Then:**
  - Use average as the new client time



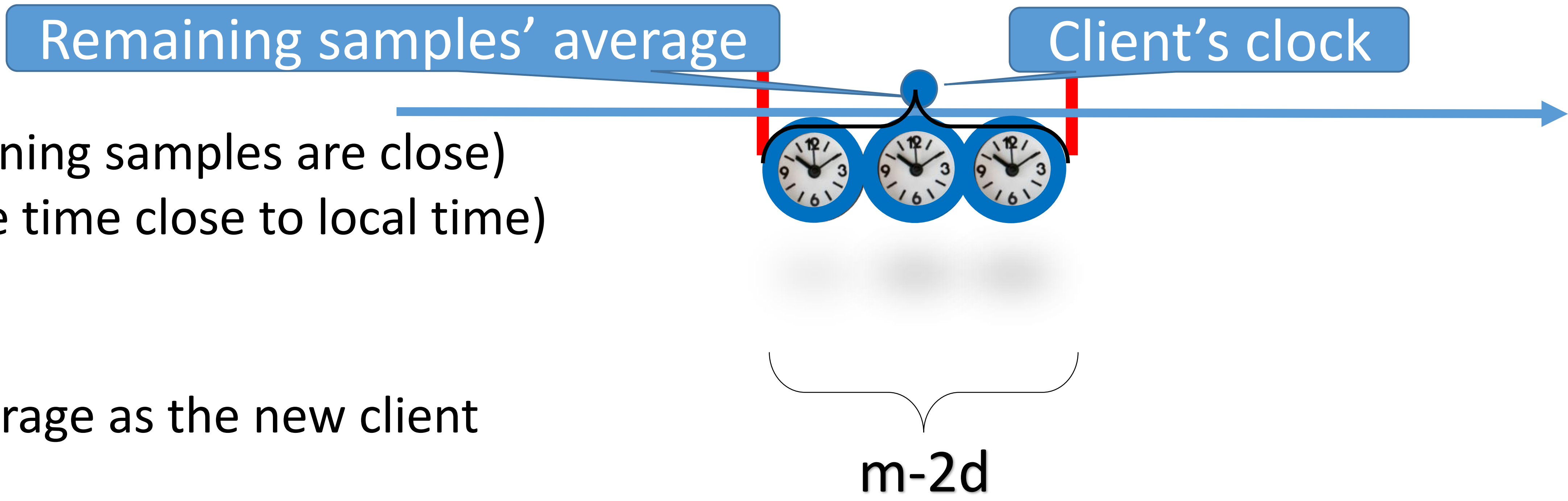
Chronos' Time-Update Algorithm: Informal

Check:

If (the remaining samples are close)

and (average time close to local time)

- **Then:**
  - Use average as the new client time
- **Else**
  - Resample



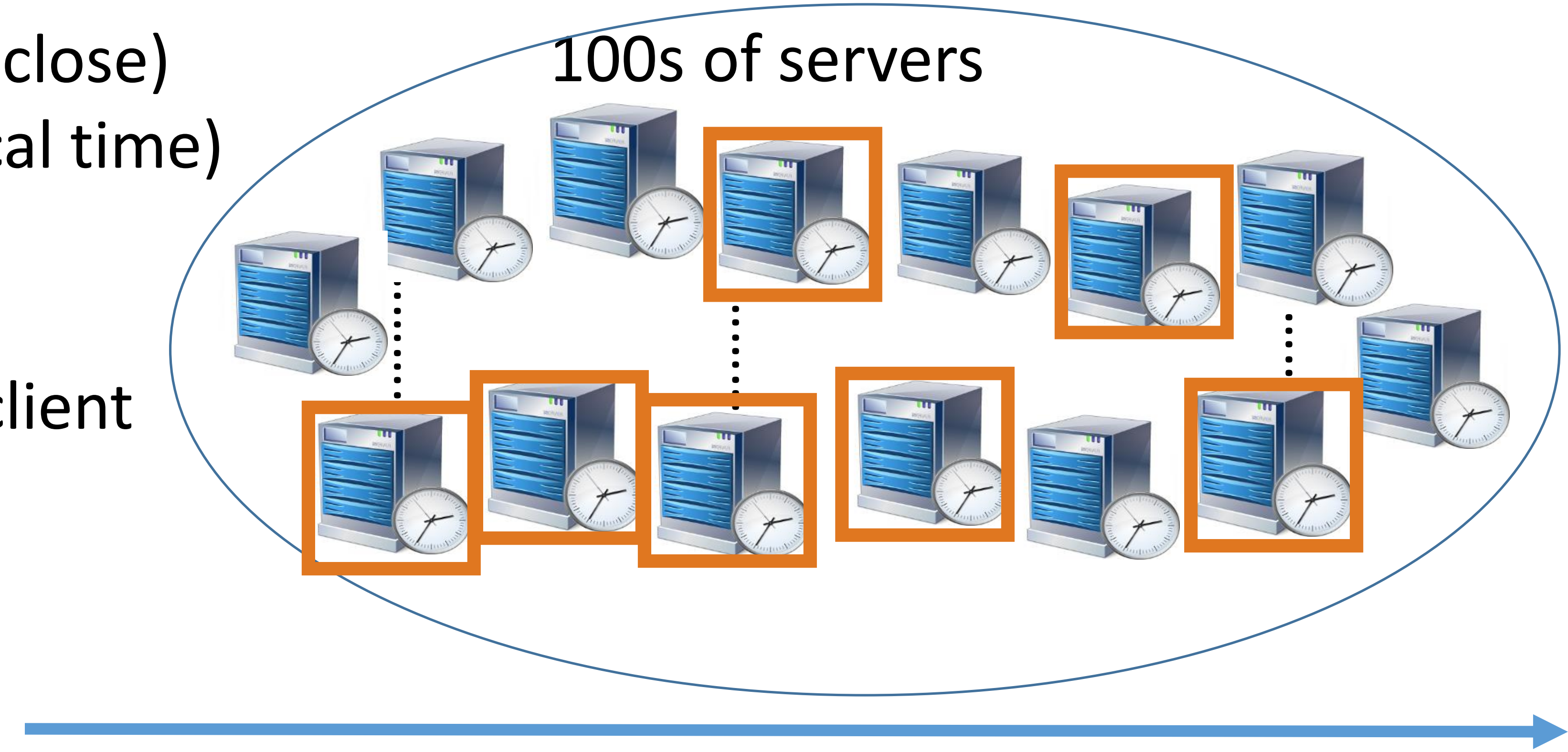
Chronos' Time-Update Algorithm: Informal

Check:

If (the remaining samples are close)

and (average time close to local time)

- **Then:**
    - Use average as the new client time
- **Else**
    - Resample



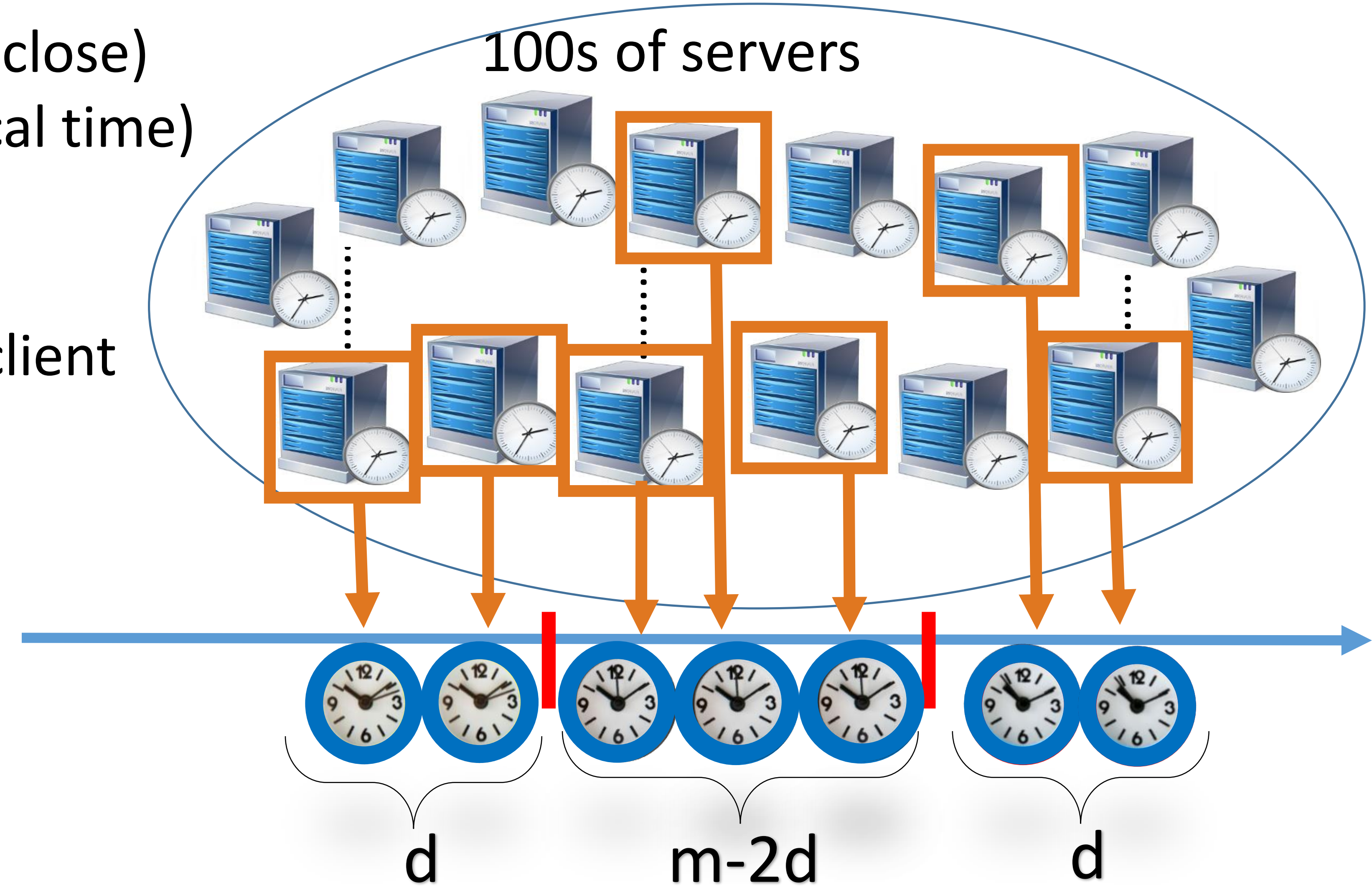
Chronos' Time-Update Algorithm: Informal

Check:

If (the remaining samples are close)

and (average time close to local time)

- **Then:**
  - Use average as the new client time
- **Else**
  - Resample



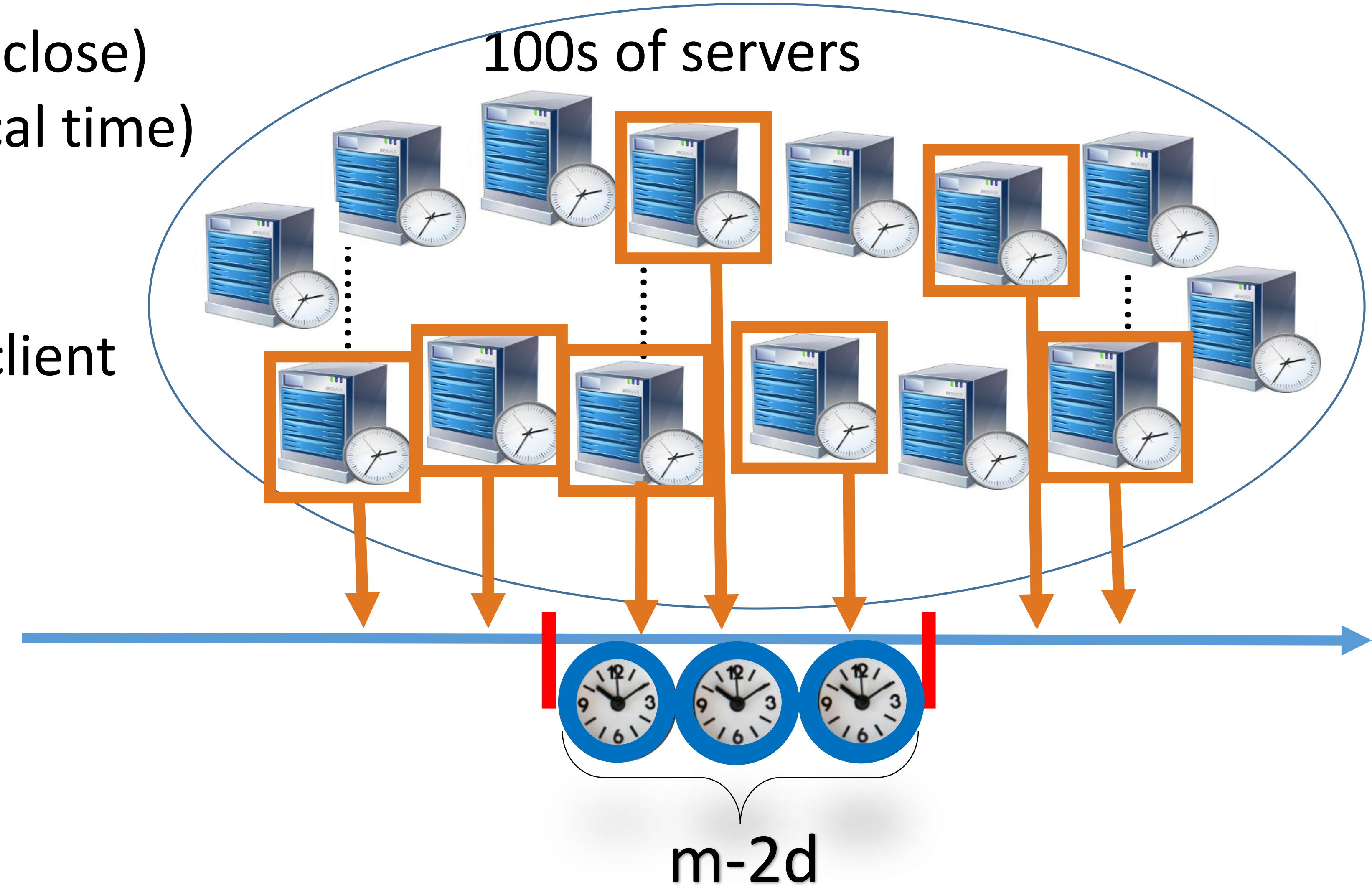
Chronos' Time-Update Algorithm: Informal

Check:

If (the remaining samples are close)

and (average time close to local time)

- **Then:**
  - Use average as the new client time
- **Else**
  - Resample



Chronos' Time-Update Algorithm: Informal

if check & resample failed k times:

\\ **panic mode**

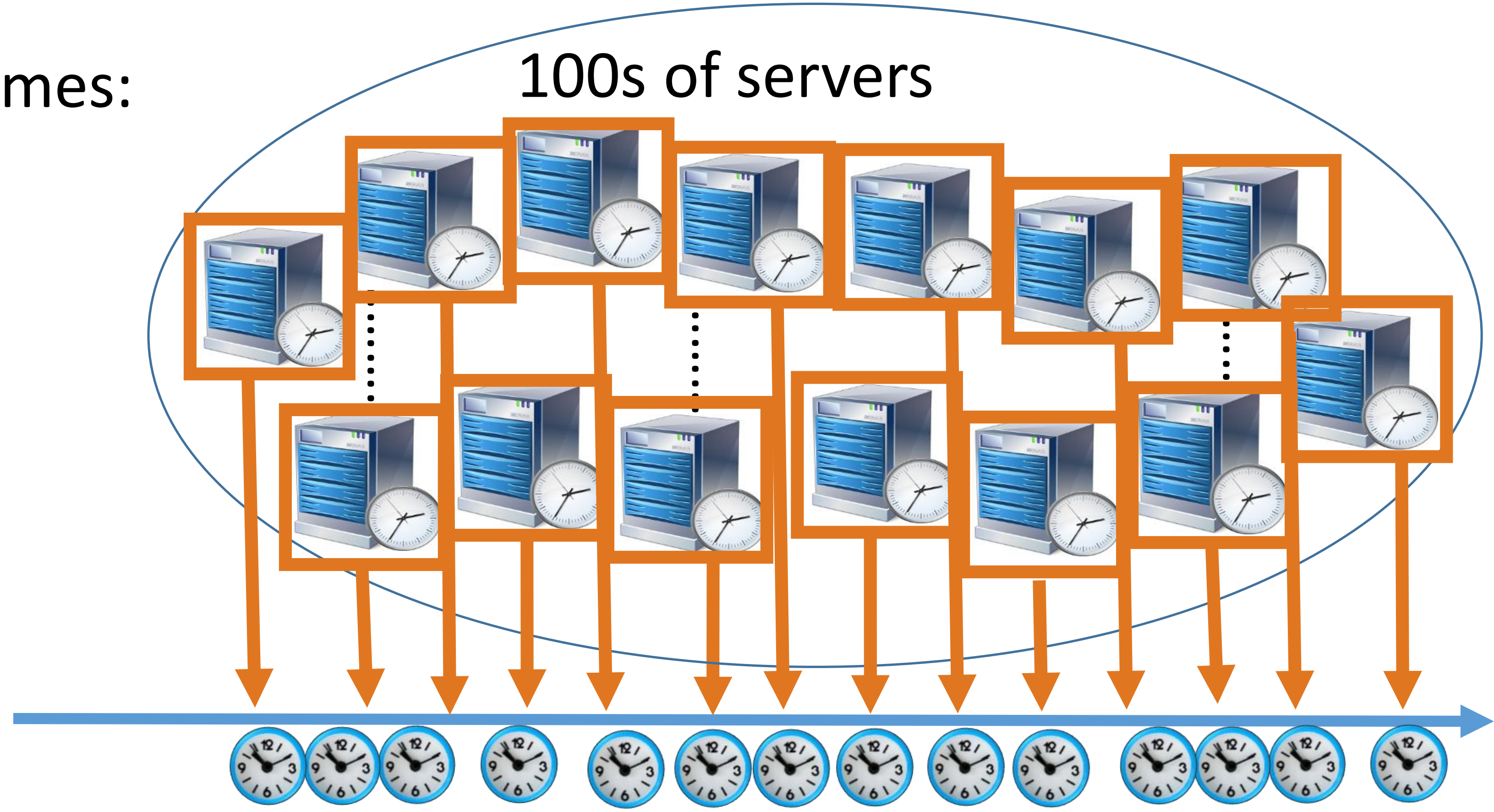
- Sample all servers

Chronos' Time-Update Algorithm: Informal

if check & resample failed k times:

\\ **panic mode**

- Sample all servers

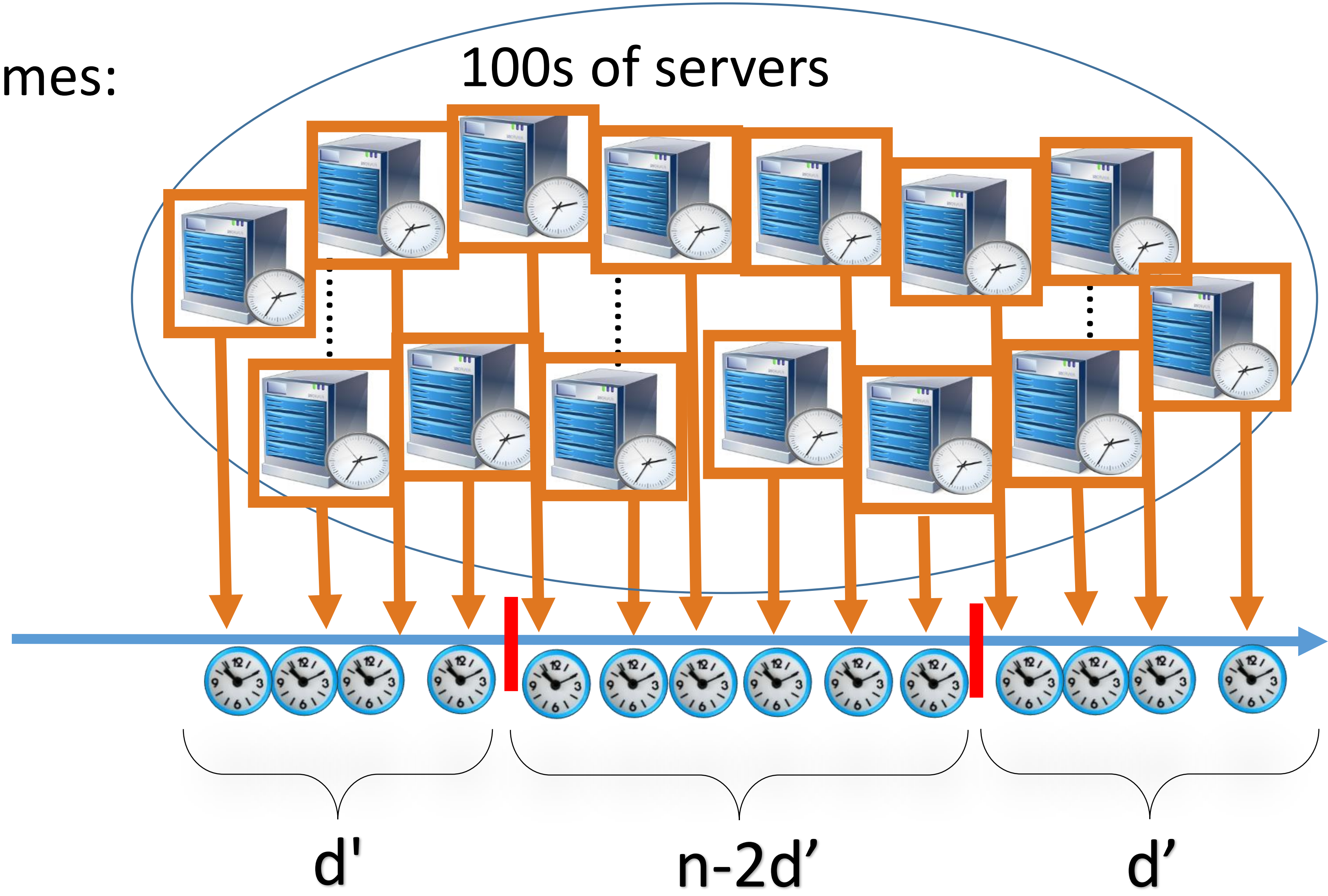


Chronos' Time-Update Algorithm: Informal

if check & resample failed k times:

\\ **panic mode**

- Sample all servers
- Drop outliers

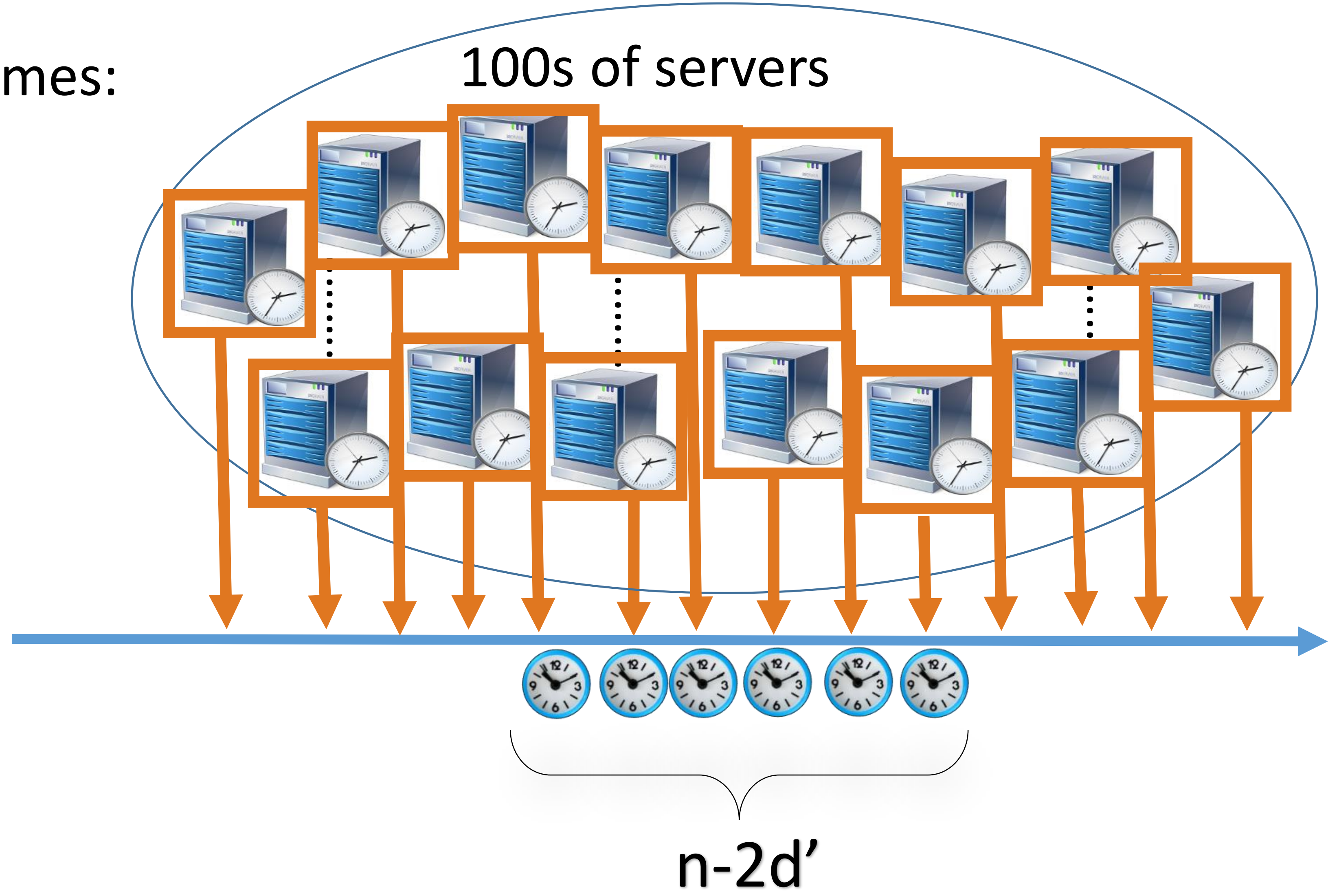


Chronos' Time-Update Algorithm: Informal

if check & resample failed k times:

\\ **panic mode**

- Sample all servers
- Drop outliers

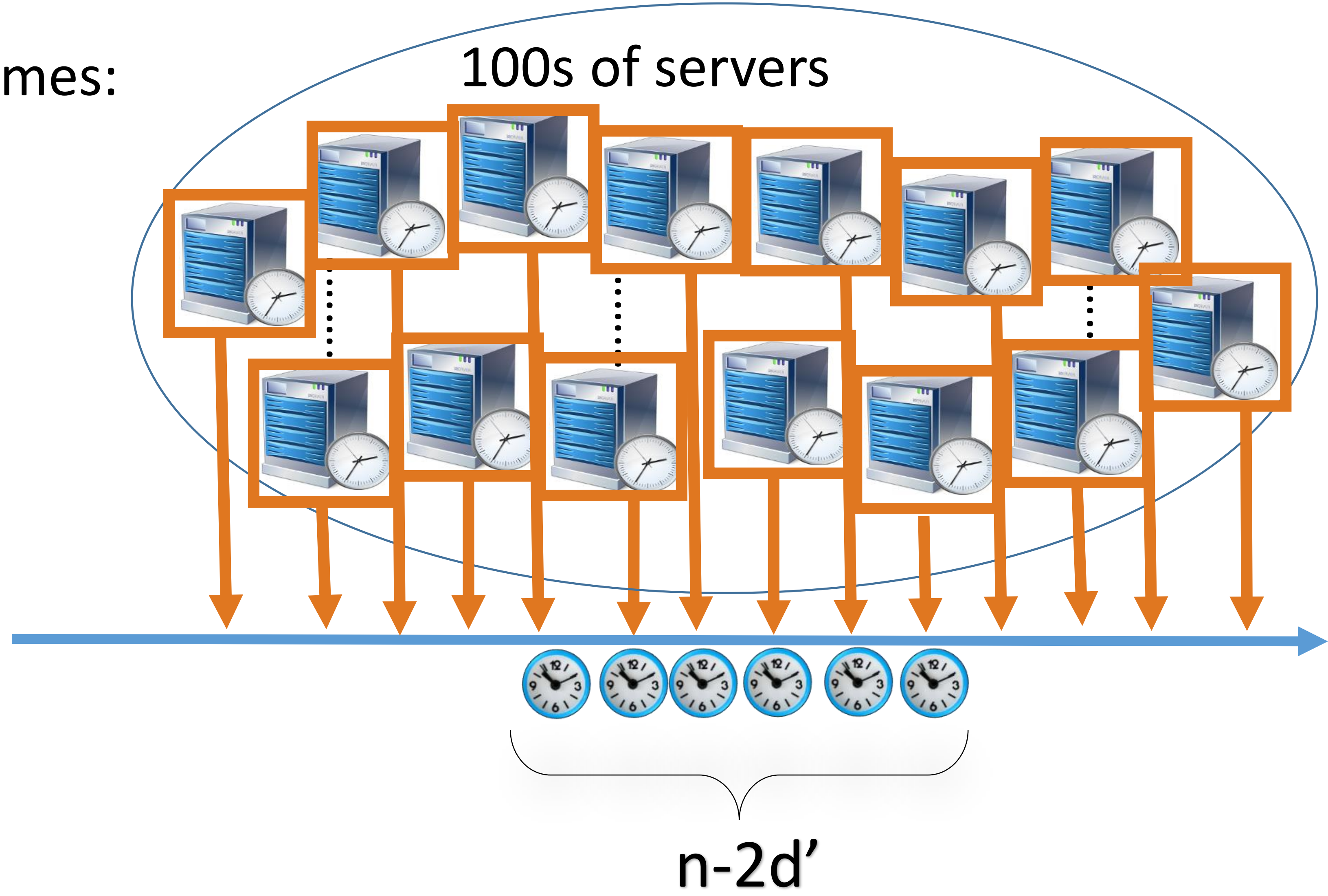


Chronos' Time-Update Algorithm: Informal

if check & resample failed k times:

\\\\ **panic mode**

- Sample all servers
- Drop outliers
- Use average as new client time

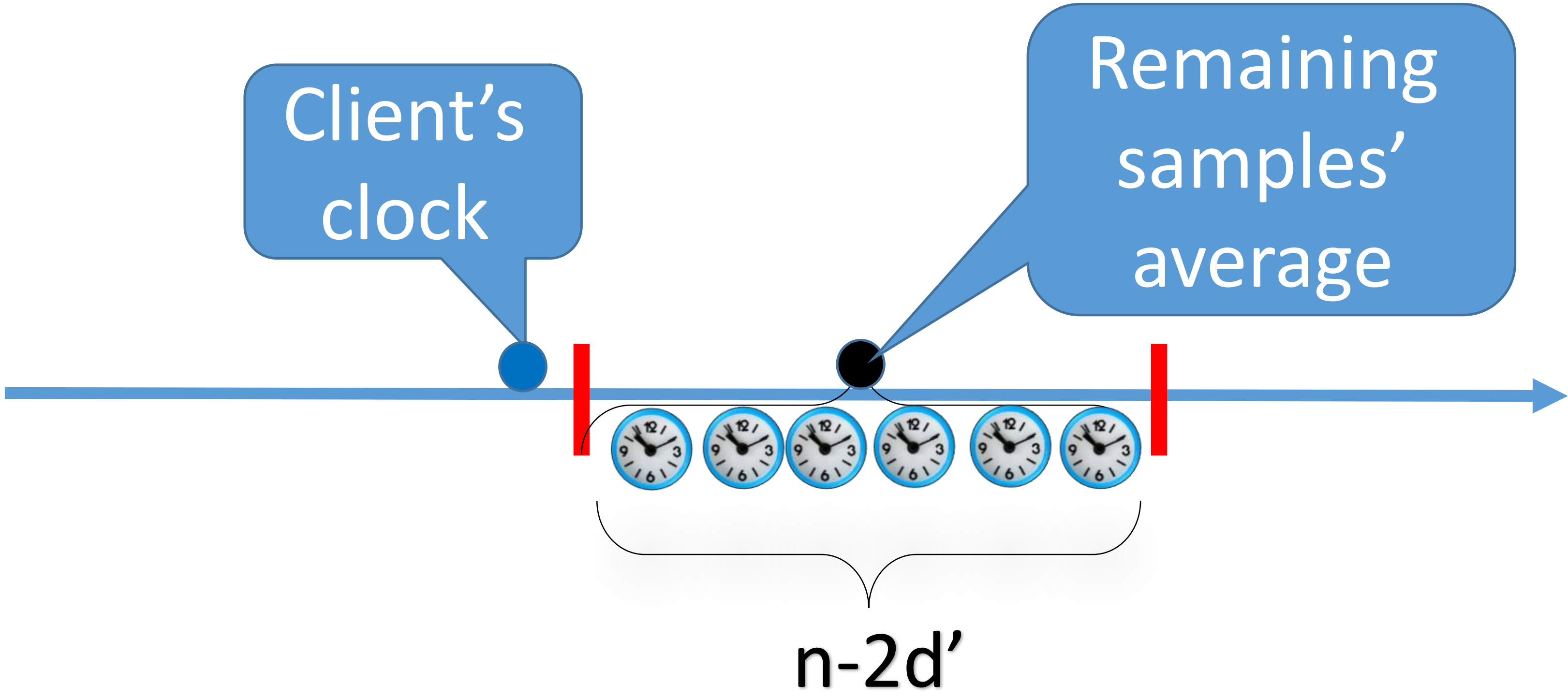


Chronos’ Time-Update Algorithm: Informal

if check & resample failed k times:

\\ **panic mode**

- Sample all servers
- Drop outliers
- Use average as new client time

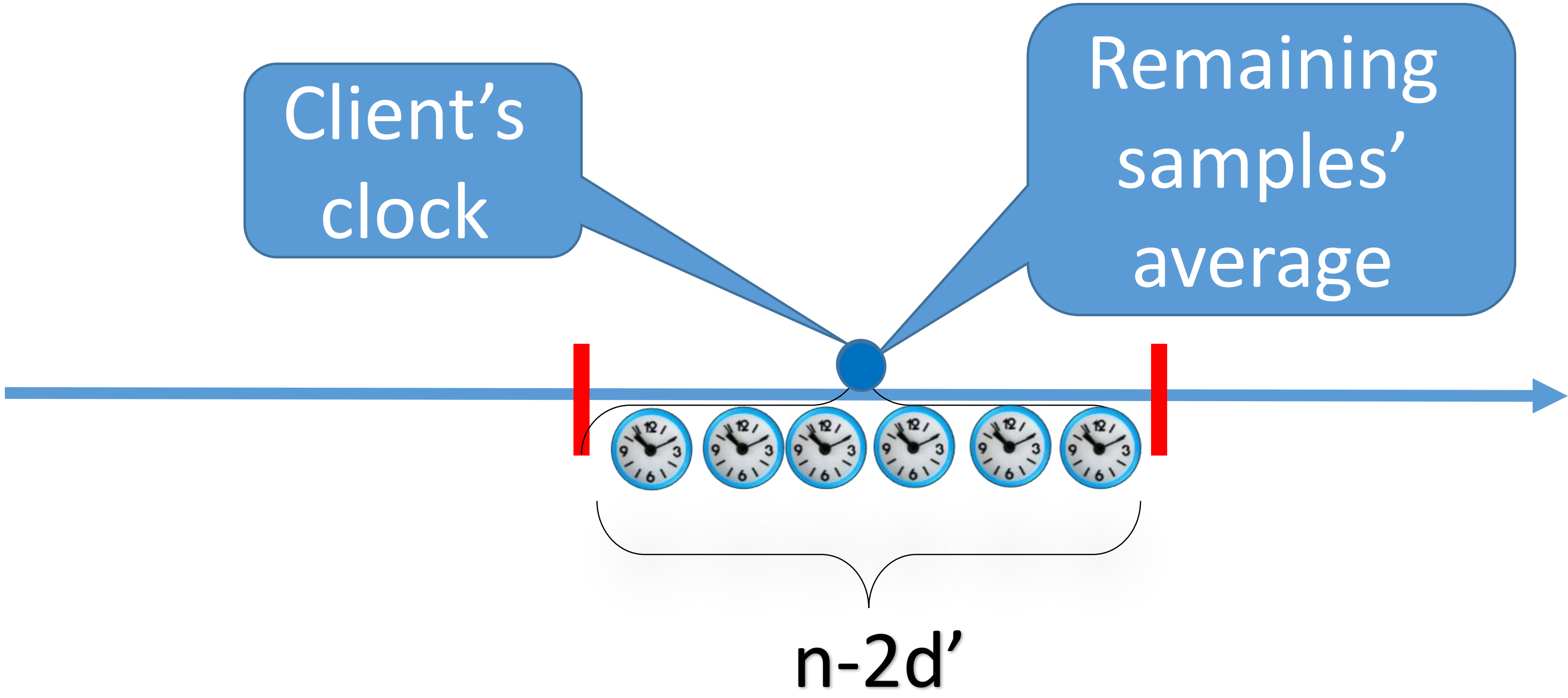


Chronos' Time-Update Algorithm: Informal

if check & resample failed k times:

\\\\ **panic mode**

- Sample all servers
- Drop outliers
- Use average as new client time



Security Guarantees

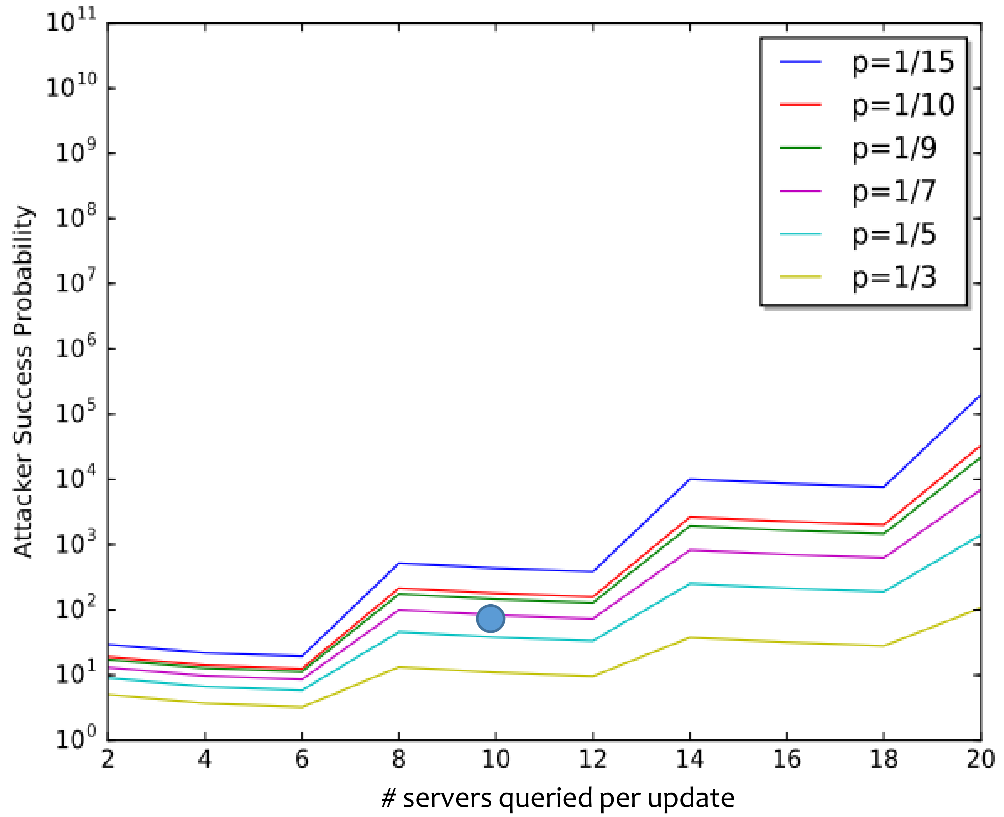
Shifting time at a Chronos client by at least **100ms** from the UTC will take the attacker at least **22 years** in expectation

- … when considering the following parameters:
  - Server pool of 500 servers, of whom 1/7 are controlled by an attacker
  - 15 servers queried once an hour
  - Good samples are within 25ms from UTC (ω=25)

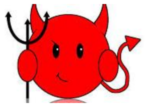
- These parameters are derived from experiments we performed on AWS servers in Europe and the US

Chronos vs. Current NTP Clients

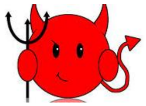
- Consider a pool of 500 servers, a p-fraction of which is controlled by an attacker.
- We compute the attacker's probability of successfully shifting the client's clock
  - ➢ for traditional NTP client
  - ➢ for Chronos NTP client
- We plot the ratio between these probabilities



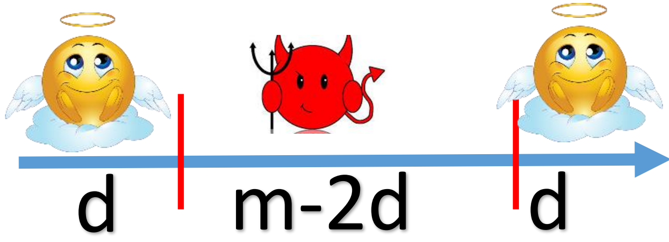
Security Guarantees: Intuition

Scenario 1: #() > d #() < m-d

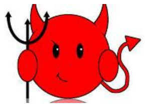
Security Guarantees: Intuition

Scenario 1: #() > d #() < m-d

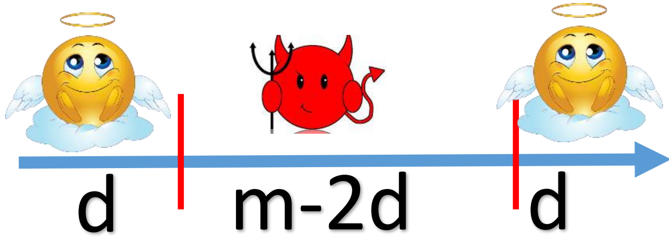
- **<u>Option I</u>**: Only malicious samples remain
  - ➢ <u>Assumption</u>: every good sample at most ω-far from UTC
  - ➢ At least one good sample on each side
    - → All remaining samples are between two good samples
    - → All remaining samples are at most ω-away from UTC



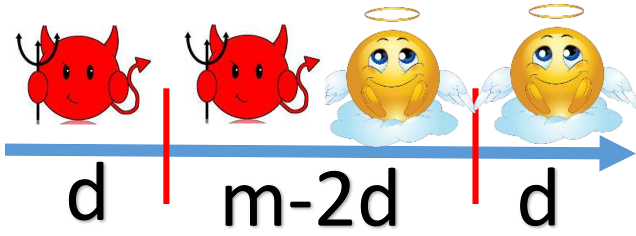
Security Guarantees: Intuition

Scenario 1: #() > d #() < m-d

- **<u>Option I</u>**: Only malicious samples remain
  - ➢ <u>Assumption</u>: every good sample at most ω-far from UTC
  - ➢ At least one good sample on each side
    - → All remaining samples are between two good samples
    - → All remaining samples are at most ω-away from UTC



- **<u>Option II</u>**: At least one good sample remains

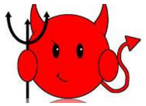


➢ <u>Enforced</u>: Remaining samples within the same 2ω-interval

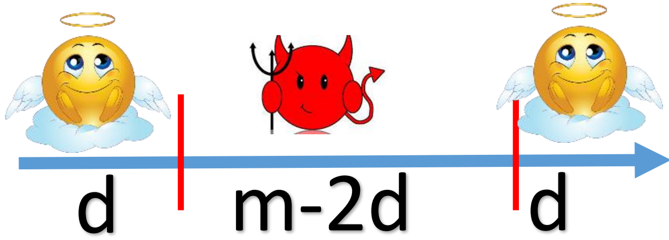
➢ Remaining malicious samples are within 2ω from a good sample

- → Remaining malicious samples are at most 3ω-away from UTC

Security Guarantees: Intuition

Scenario 1: #() > d #() < m-d

- **<u>Option I</u>**: Only malicious samples remain



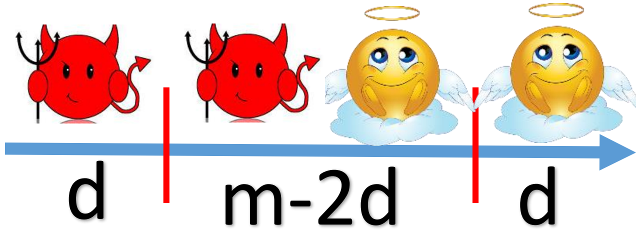
➢<u>Assumption</u>: every good sample at most ω-far from UTC

➢At least one good sample on each side

→ All remaining samples are between two good samples

→ All remaining samples are at most ω-away from UTC

- **<u>Option II</u>**: At least one good sample remains



➢<u>Enforced</u>: Remaining samples within the same 2ω-interval

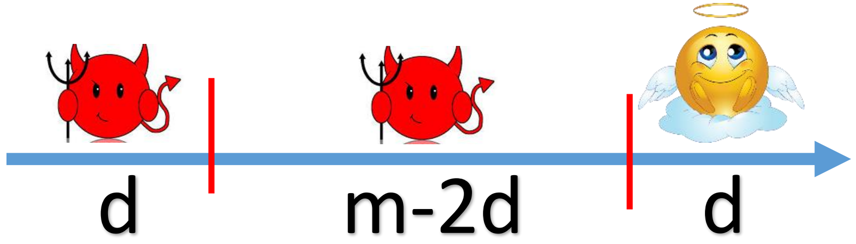
➢Remaining malicious samples are within 2ω from a good sample

→ Remaining malicious samples are at most 3ω-away from UTC

Hence, these attack strategies are ineffective

Security Guarantees: Intuition

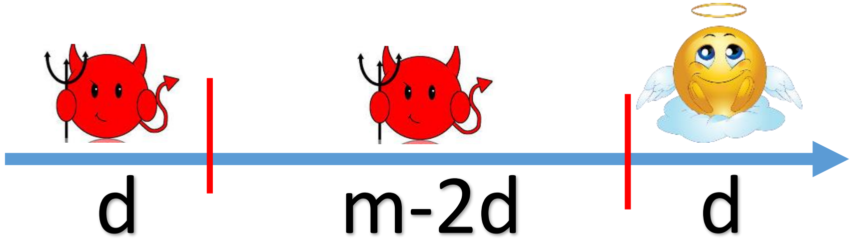
Scenario 2: #() ≤ d #() ≥ m-d



- Optimal attack strategy:
  All malicious samples are lower than all good samples
  (Or, all malicious samples are higher than all good samples)

Security Guarantees: Intuition

Scenario 2: #() ≤ d   #() ≥ m-d

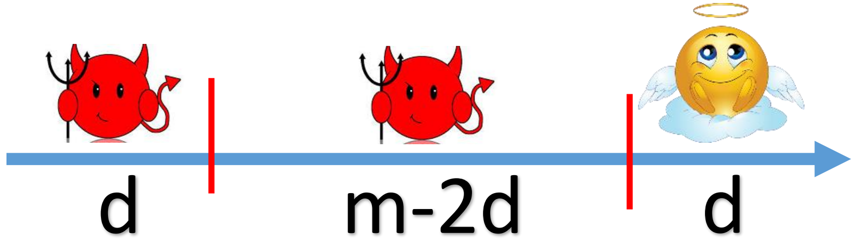


- <u>Optimal attack strategy</u>:
  All malicious samples are lower than all good samples
  (Or, all malicious samples are higher than all good samples)

- **Chronos enforces an upper bound of 4ω on the permissible shift from the local clock** (otherwise the server pool is re-sampled)

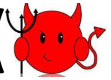
Security Guarantees: Intuition

Scenario 2: #() ≤ d #() ≥ m-d



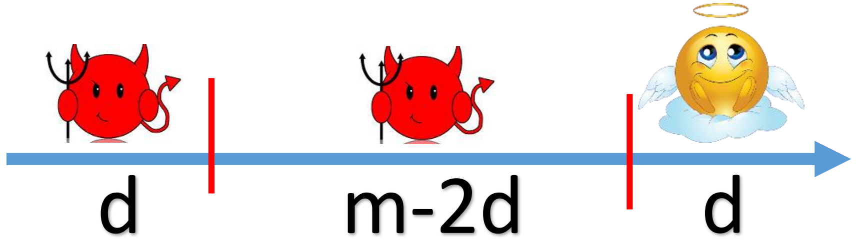
- <u>Optimal attack strategy</u>:
  All malicious samples are lower than all good samples
  (Or, all malicious samples are higher than all good samples)

- **Chronos enforces an upper bound of 4ω on the permissible shift from the local clock** (otherwise the server pool is re-sampled)

- **The probability that #()≥m-d is extremely low** (see paper for detailed analysis)
  The probability of repeated shift is negligible.

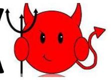
Security Guarantees: Intuition

Scenario 2: #() ≤ d   #() ≥ m-d



- <u>Optimal attack strategy</u>:
  All malicious samples are lower than all good samples
  (Or, all malicious samples are higher than all good samples)

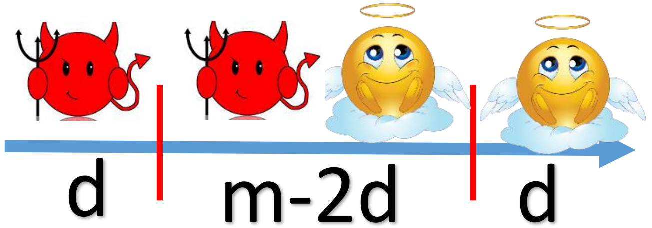
- **Chronos enforces an upper bound of 4ω on the permissible shift from the local clock** (otherwise the server pool is re-sampled)

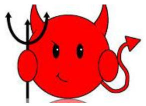
- **The probability that #()≥m-d is extremely low** (see paper for detailed analysis)
  The probability of repeated shift is negligible.

Consequently, a significant time shift is practically infeasible

Can Chronos be exploited for DoS attacks?

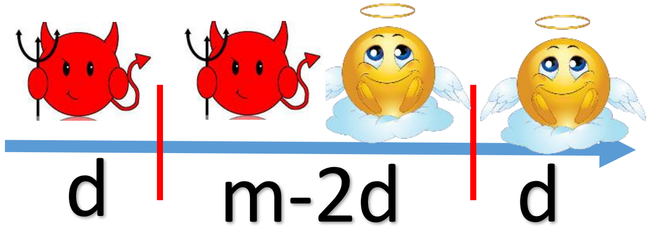
- Chronos repeatedly enters Panic Mode.

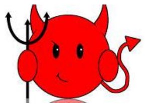


- Optimal attack strategy requires that attacker repeatedly succeed in accomplishing #() > d #() < m-d
  - At least one malicious sample remain
  - Malicious sample violates condition that all remaining samples be clustered
  - This leads to resampling (until Panic Threshold is exceeded).

Can Chronos be exploited for DoS attacks?

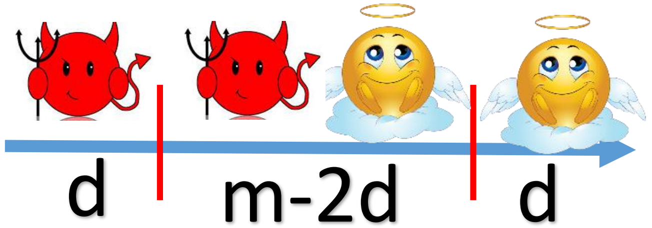
- Chronos repeatedly enters Panic Mode.

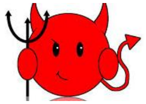


- <u>Optimal attack strategy</u> requires that attacker repeatedly succeed in accomplishing #() > d #() < m-d
  - At least one malicious sample remain
  - Malicious sample violates condition that all remaining samples be clustered
  - This leads to resampling (until Panic Threshold is exceeded).

Can Chronos be exploited for DoS attacks?

- Chronos repeatedly enters Panic Mode.



- Optimal attack strategy requires that attacker repeatedly succeed in accomplishing #() > d #() < m-d
  - At least one malicious sample remain
  - Malicious sample violates condition that all remaining samples be clustered
  - This leads to resampling (until Panic Threshold is exceeded).

Even for low Panic Threshold (k=3), probability of success is negligible (will take attacker decades to force Panic Mode)

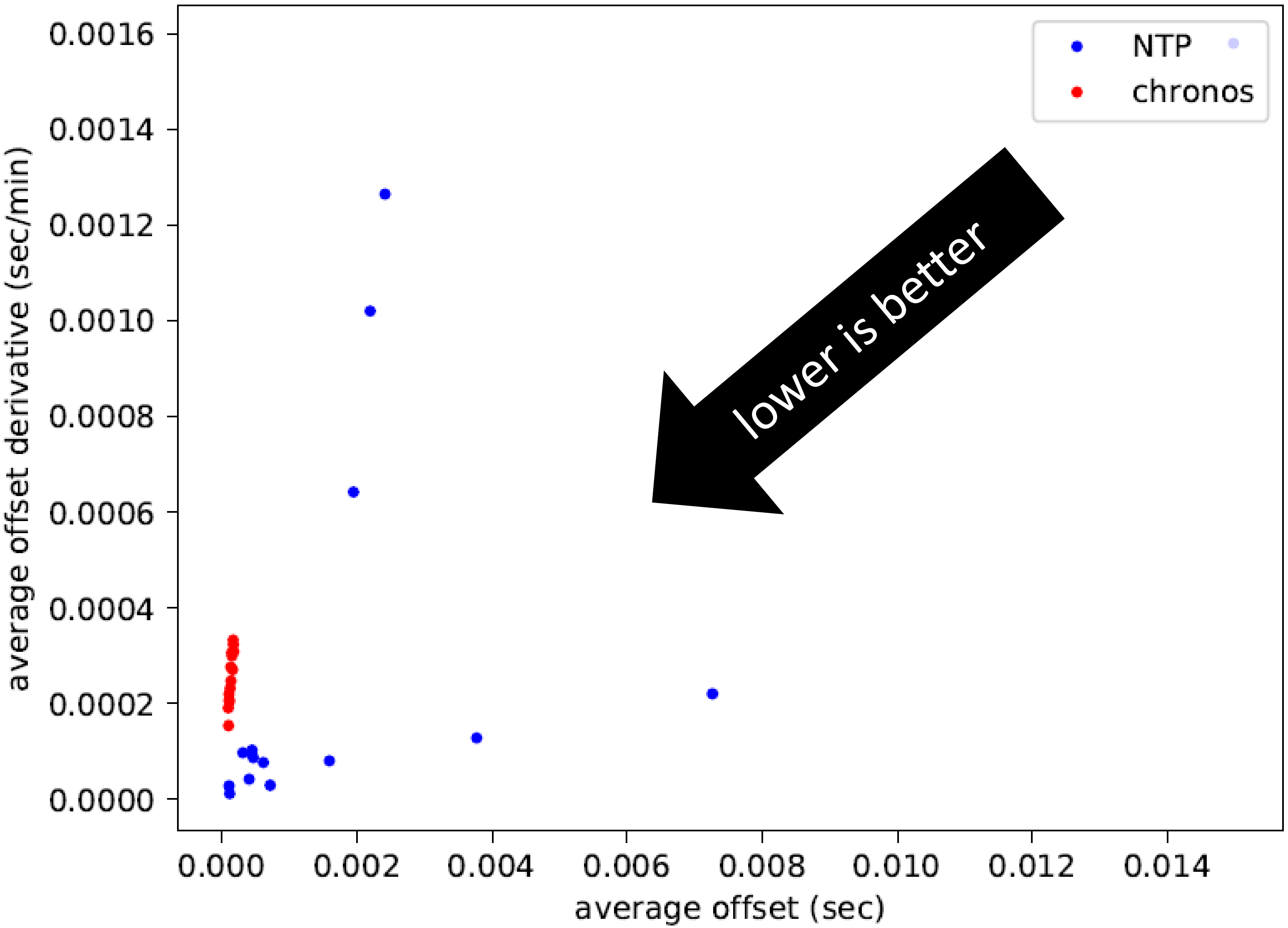
Chronos *vs.* NTPd

- Greater variety of sampled servers over time
- Provable security guarantees
- Avoids (NTPv4) source quality filters
- Possible adverse effects on precision and accuracy.

Chronos' Precision and Accuracy

- To improve precision without sacrificing security, we introduce a smoothing mechanism:
  - Return the minimal sampled offset unless its distance from the average is higher than a predefined value
- We evaluated Chronos at multiple locations in Europe and the US

Average offsets and derivatives



Conclusion

- NTP is highly vulnerable to time-shifting attacks
  - Attacker in control of a few servers/sessions can shift client’s time
- We presented the **Chronos NTP client**
  - provable security
  - backwards-compatibility
  - low overhead
- Chronos’ precision and offsets are close to NTP (around 2ms apart)

Ongoing and Future Efforts

- Evaluate Chronos at scale (security, precision, accuracy, overhead, …)
- Standardize Chronos!
- Extending Chronos to address several attack strategies
- Extensions to other time-synchronization protocols (e.g., PTP)?

Thank You

See full paper (@NDSS'18):
http://wp.internetsociety.org/ndss/wp-content/uploads/sites/25/2018/02/ndss2018_02A-2_Deutsch_paper.pdf

See last IETF draft version:
https://tools.ietf.org/html/draft-schiff-ntp-chronos-02