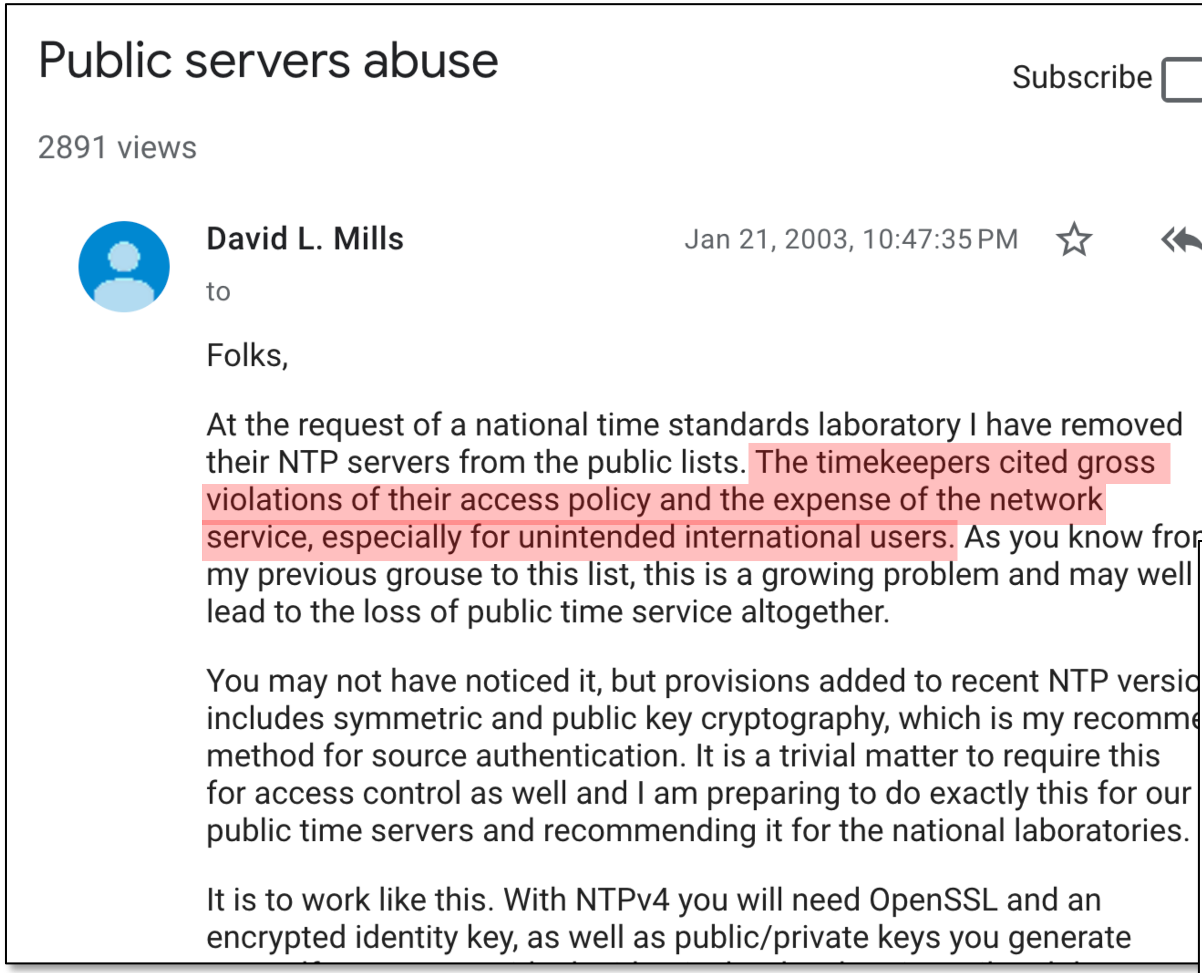
32ND USENIX SECURITY SYMPOSIUM

Did the Shark Eat the Watchdog in the NTP Pool? Deceiving the NTP Pool's Monitoring System

Jonghoon Kwon[+], Jeonggyu Song[*],
Junbeom Hur[*], and Adrian Perrig[+]

[+]ETH Zürich, [*]Korea University

Public Time Service in Earlier Days…



Public servers abuse

Subscribe □

2891 views

David L. Mills Jan 21, 2003, 10:47:35 PM

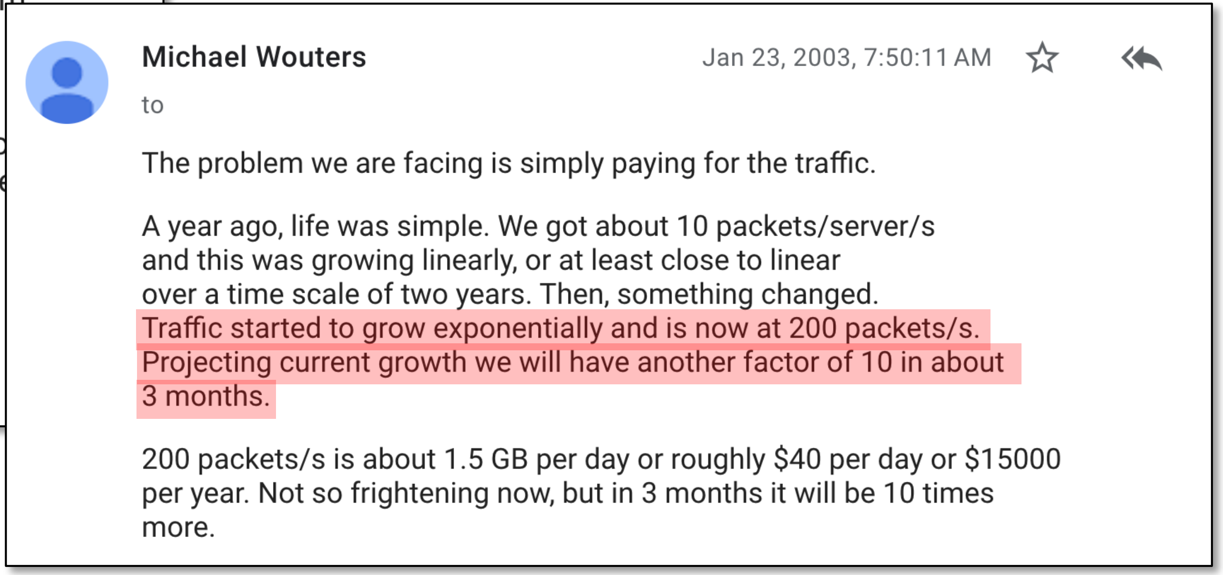
to

Folks,

At the request of a national time standards laboratory I have removed their NTP servers from the public lists. The timekeepers cited gross violations of their access policy and the expense of the network service, especially for unintended international users. As you know from my previous grouse to this list, this is a growing problem and may well lead to the loss of public time service altogether.

You may not have noticed it, but provisions added to recent NTP versio includes symmetric and public key cryptography, which is my recomme method for source authentication. It is a trivial matter to require this for access control as well and I am preparing to do exactly this for our public time servers and recommending it for the national laboratories.

It is to work like this. With NTPv4 you will need OpenSSL and an encrypted identity key, as well as public/private keys you generate



Michael Wouters Jan 23, 2003, 7:50:11 AM

to

The problem we are facing is simply paying for the traffic.

A year ago, life was simple. We got about 10 packets/server/s
and this was growing linearly, or at least close to linear
over a time scale of two years. Then, something changed.
Traffic started to grow exponentially and is now at 200 packets/s.
Projecting current growth we will have another factor of 10 in about
3 months.

200 packets/s is about 1.5 GB per day or roughly $40 per day or $15000
per year. Not so frightening now, but in 3 months it will be 10 times
more.

NTP Pool Project: the Largest NTP Ecosystem

- Response to the increasing resource consumption at popular NTP servers

NTP Pool

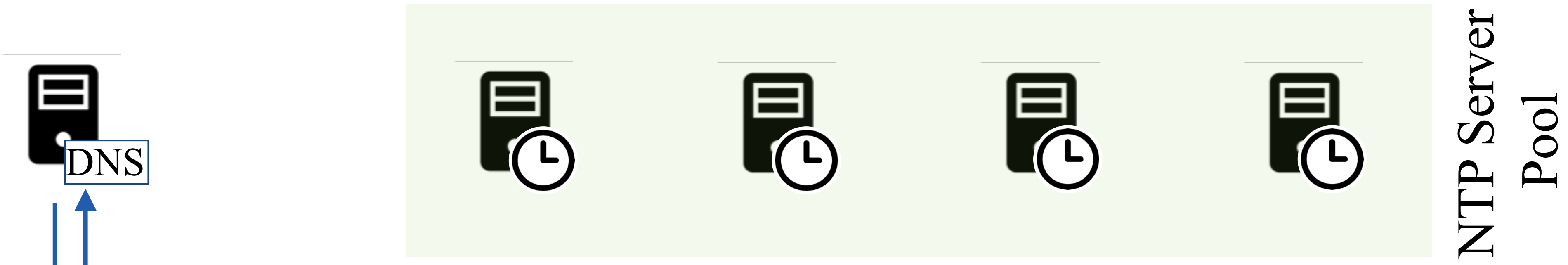
4.6 k public timeservers

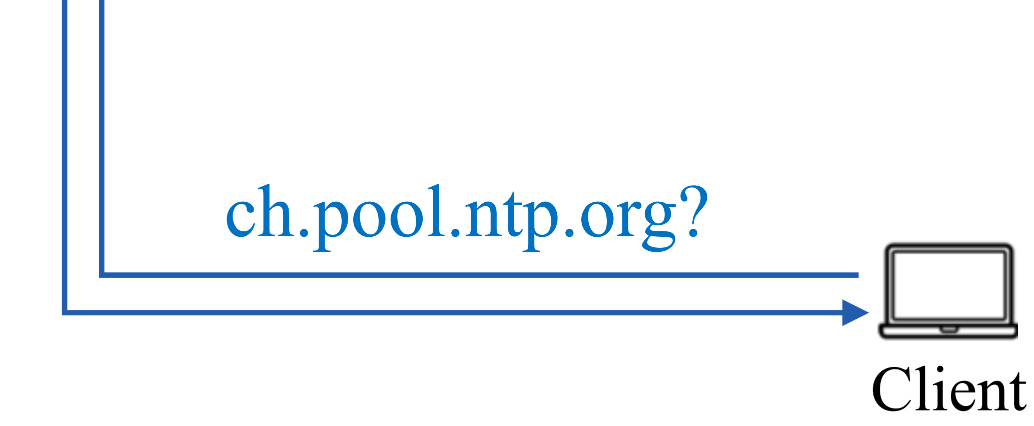
(Aug. 2023)

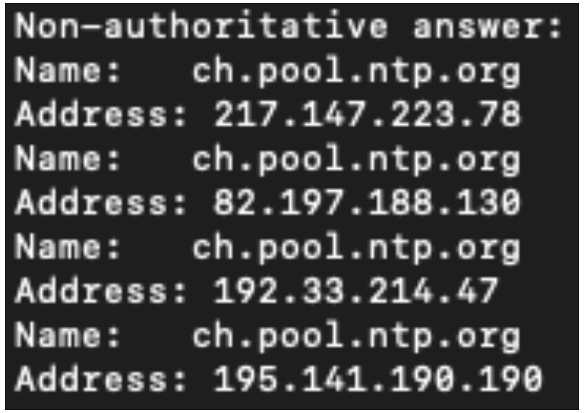
Hundreds of millions of Clients

- Linux distributions (e.g., Debian)
- Networked appliances (e.g., Netgear)
- Android smartphones and IoT devices

NTP Pool Architecture at a Glance

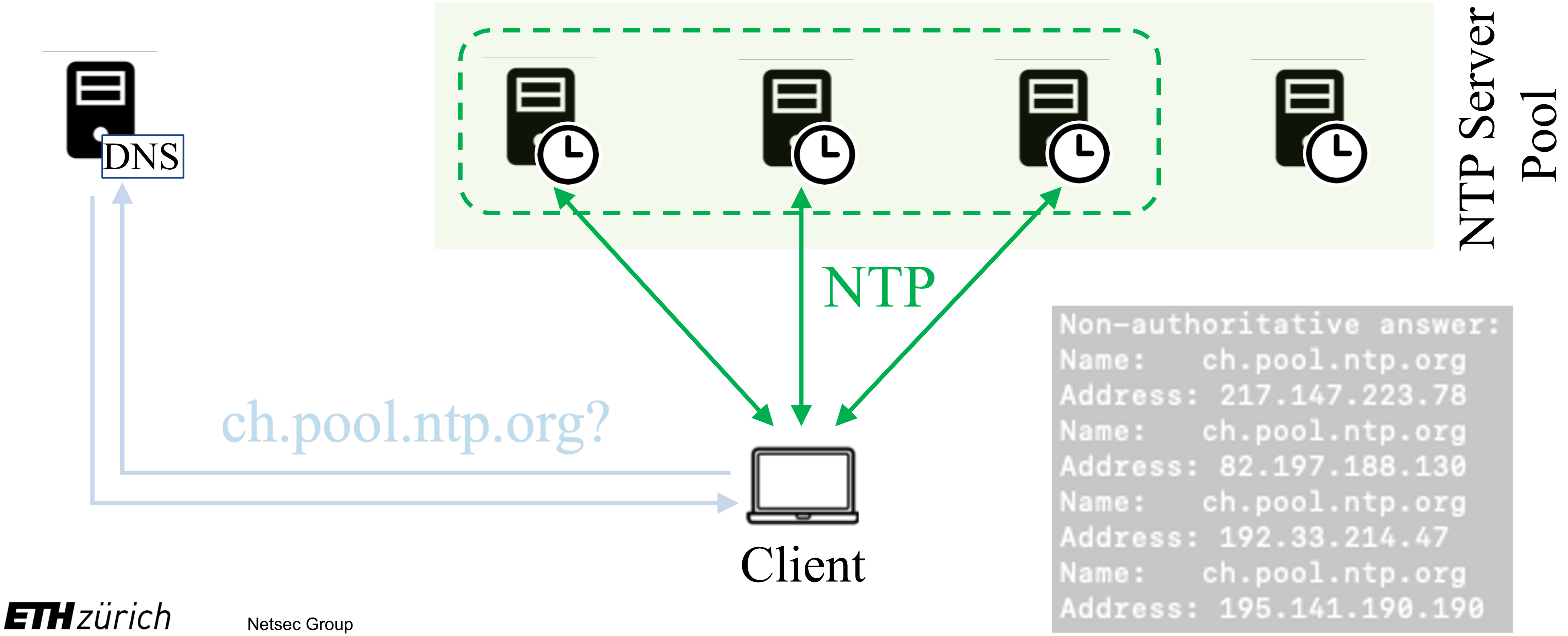






```
Non-authoritative answer:
Name:    ch.pool.ntp.org
Address: 217.147.223.78
Name:    ch.pool.ntp.org
Address: 82.197.188.130
Name:    ch.pool.ntp.org
Address: 192.33.214.47
Name:    ch.pool.ntp.org
Address: 195.141.190.190
```

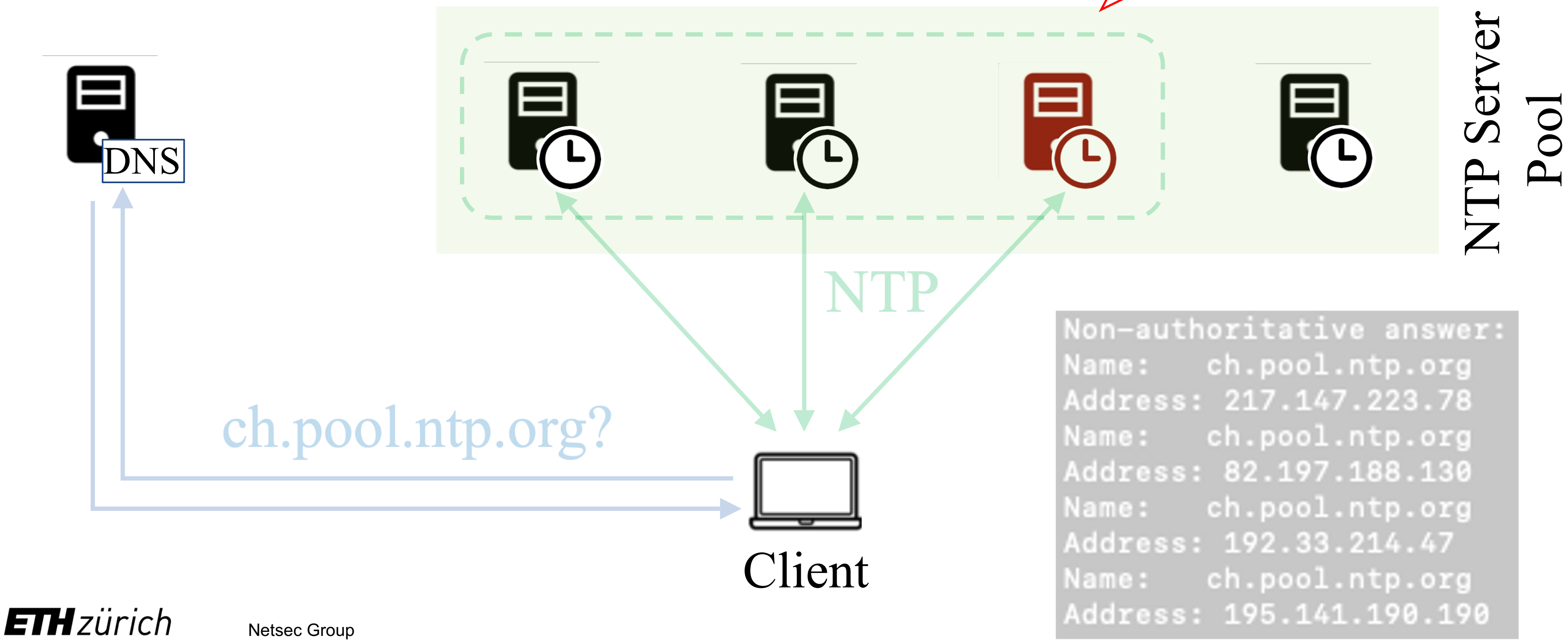
NTP Pool Architecture at a Glance



```
Non-authoritative answer:
Name:    ch.pool.ntp.org
Address: 217.147.223.78
Name:    ch.pool.ntp.org
Address: 82.197.188.130
Name:    ch.pool.ntp.org
Address: 192.33.214.47
Name:    ch.pool.ntp.org
Address: 195.141.190.190
```

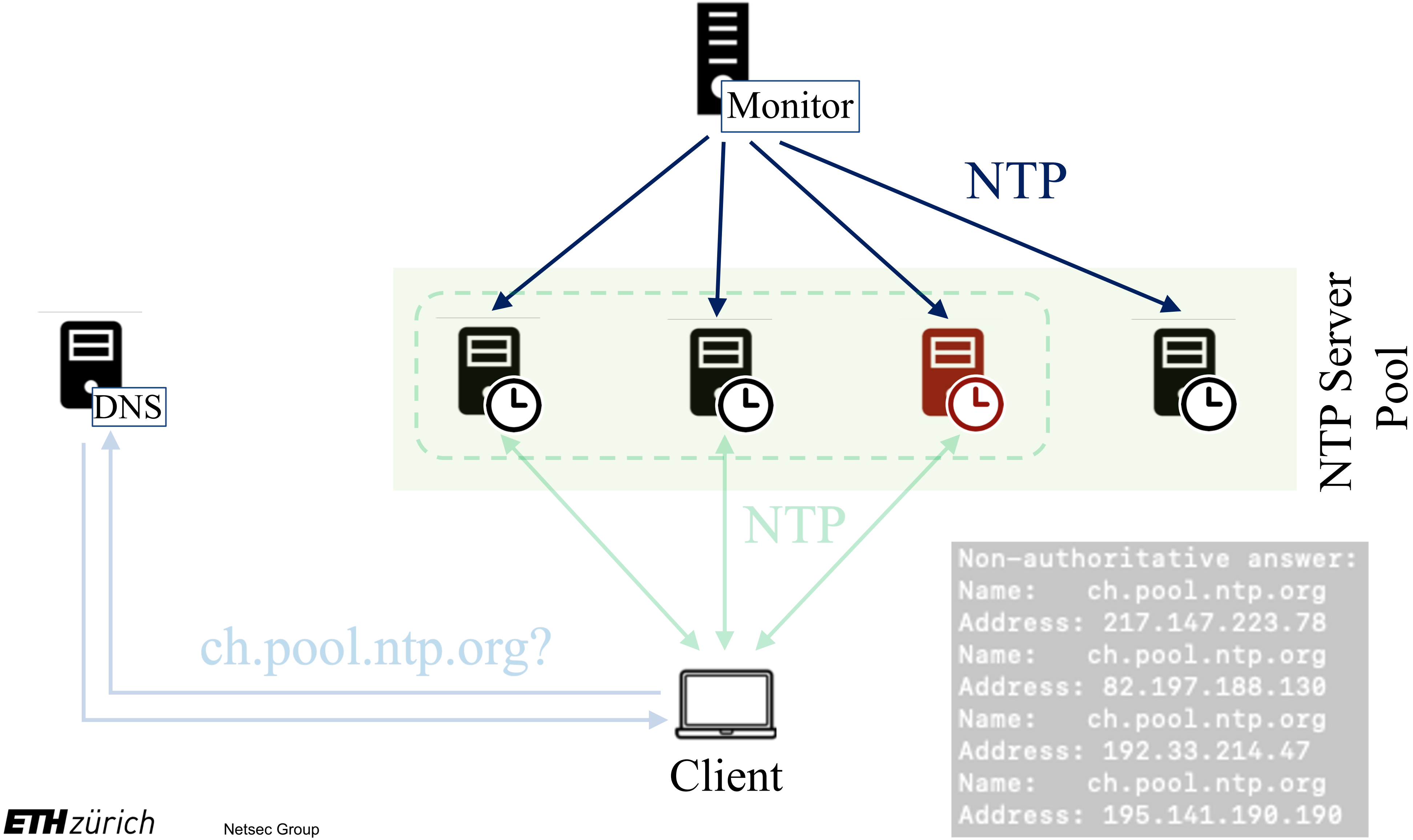
NTP Pool Architecture at a Glance

How is the time accuracy of these public servers managed?



```
Non-authoritative answer:
Name:    ch.pool.ntp.org
Address: 217.147.223.78
Name:    ch.pool.ntp.org
Address: 82.197.188.130
Name:    ch.pool.ntp.org
Address: 192.33.214.47
Name:    ch.pool.ntp.org
Address: 195.141.190.190
```

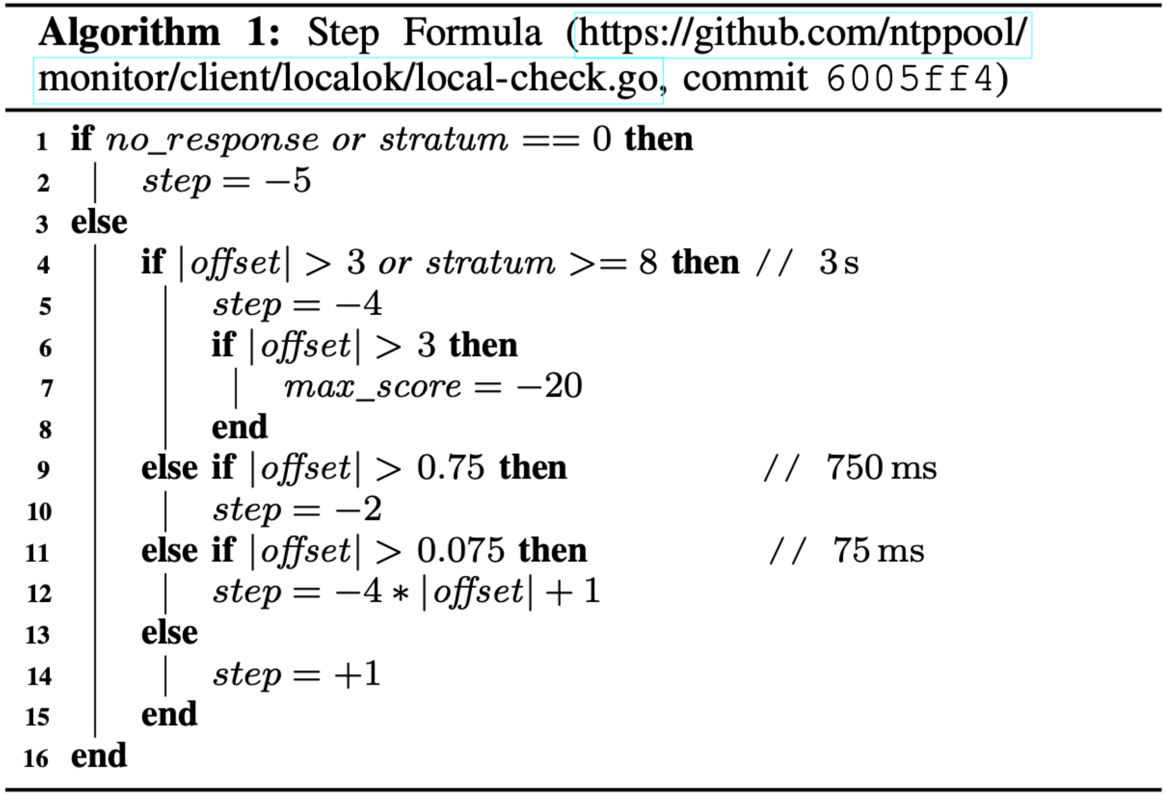
NTP Pool Architecture at a Glance



```
Non-authoritative answer:
Name:    ch.pool.ntp.org
Address: 217.147.223.78
Name:    ch.pool.ntp.org
Address: 82.197.188.130
Name:    ch.pool.ntp.org
Address: 192.33.214.47
Name:    ch.pool.ntp.org
Address: 195.141.190.190
```

NTP Pool Monitoring System

- Scoring algorithm
  - $score_{new} = min(max\_score, (score_{old} * 0.95) + step)$
  - Step formula:

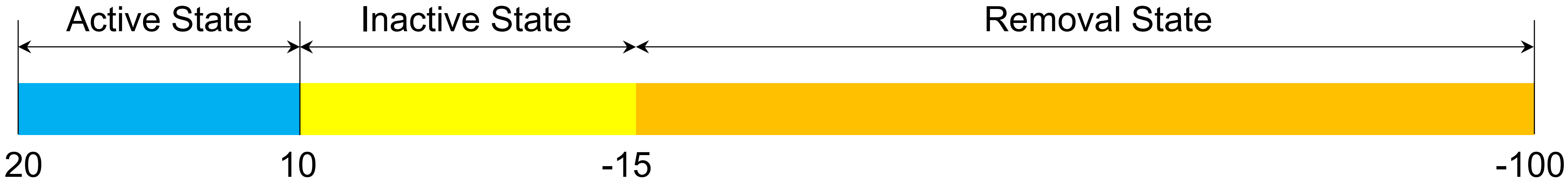


Algorithm 1: Step Formula (https://github.com/ntppool/monitor/client/localok/local-check.go, commit `6005ff4`)

```
if $no\_response$ or $stratum == 0$ then
    $step = -5$
else
    if $|offset| > 3$ or $stratum >= 8$ then   // 3 s
        $step = -4$
        if $|offset| > 3$ then
            $max\_score = -20$
        end
    else if $|offset| > 0.75$ then   // 750 ms
        $step = -2$
    else if $|offset| > 0.075$ then   // 75 ms
        $step = -4 * |offset| + 1$
    else
        $step = +1$
    end
end
```

NTP Pool Monitoring System

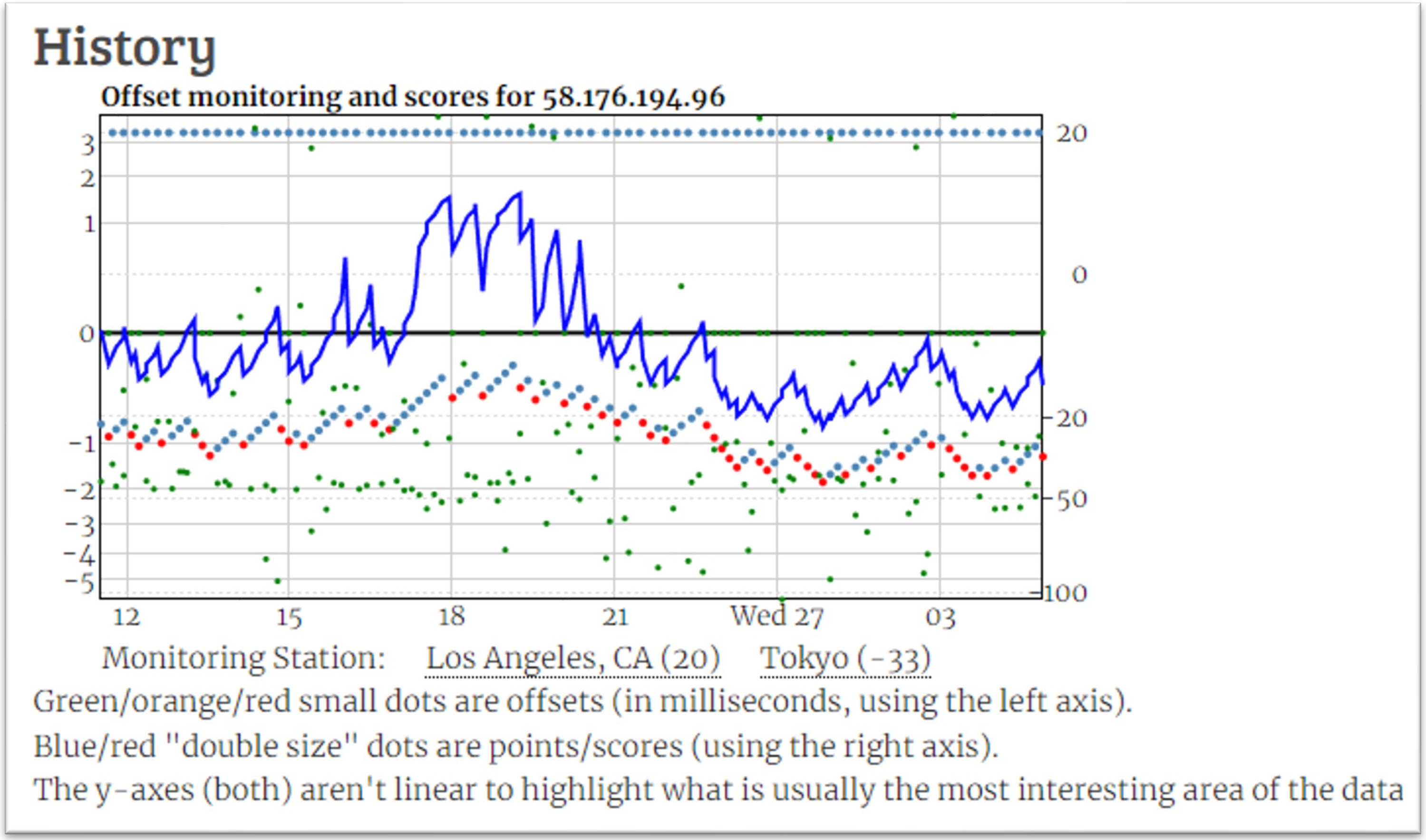
- Monitoring server inspects timeservers approx. every 13 min
  - Each timeserver is scored between 20 to -100



BAD_SERVER_THRESHOLD = -15

NTP Pool Monitoring System

- Monitoring results are publicly available



History

Offset monitoring and scores for 58.176.194.96

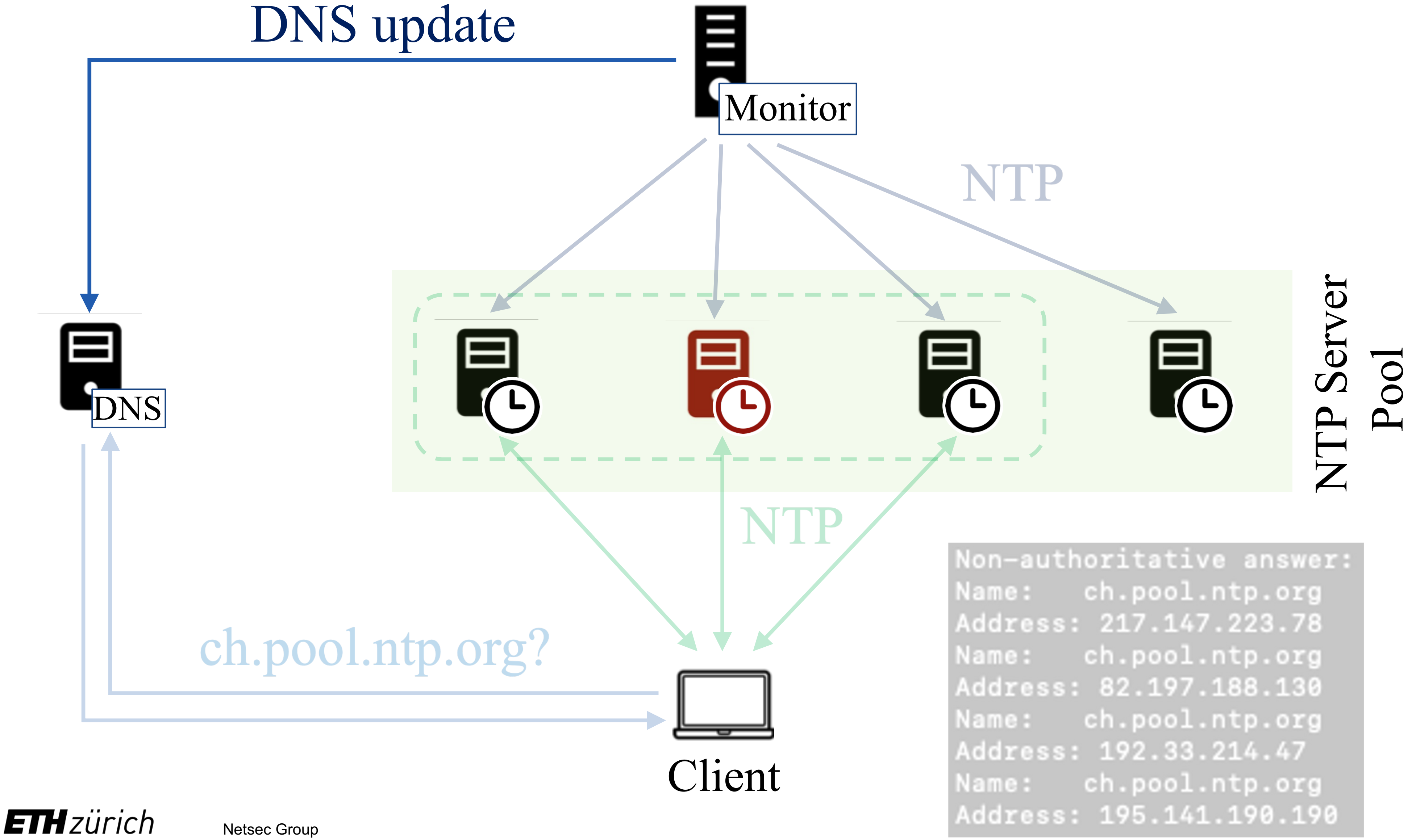
Monitoring Station: Los Angeles, CA (20) Tokyo (-33)

Green/orange/red small dots are offsets (in milliseconds, using the left axis).

Blue/red "double size" dots are points/scores (using the right axis).

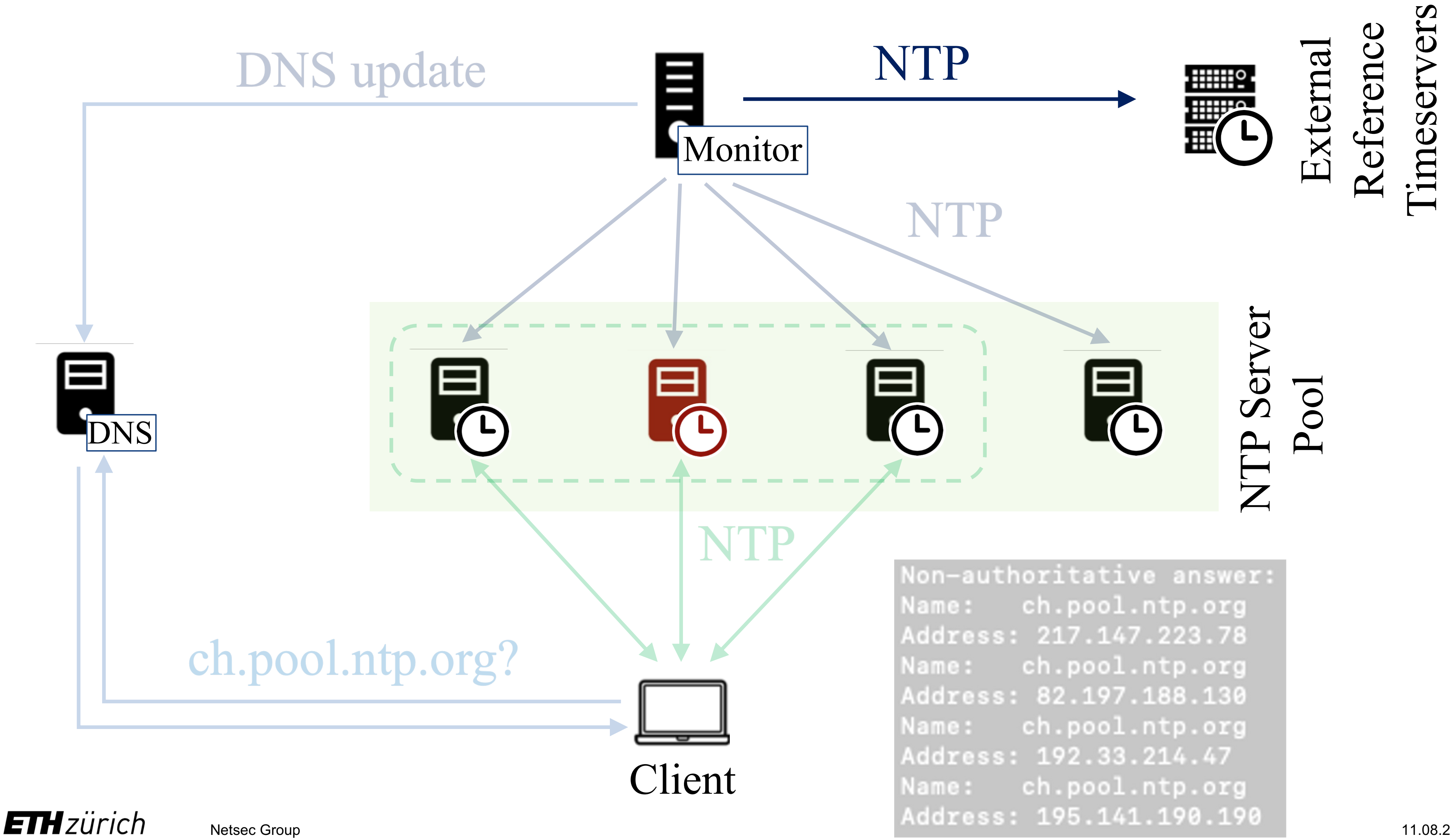
The y-axes (both) aren't linear to highlight what is usually the most interesting area of the data

NTP Pool Architecture at a Glance



```
Non-authoritative answer:
Name:    ch.pool.ntp.org
Address: 217.147.223.78
Name:    ch.pool.ntp.org
Address: 82.197.188.130
Name:    ch.pool.ntp.org
Address: 192.33.214.47
Name:    ch.pool.ntp.org
Address: 195.141.190.190
```

NTP Pool Architecture at a Glance

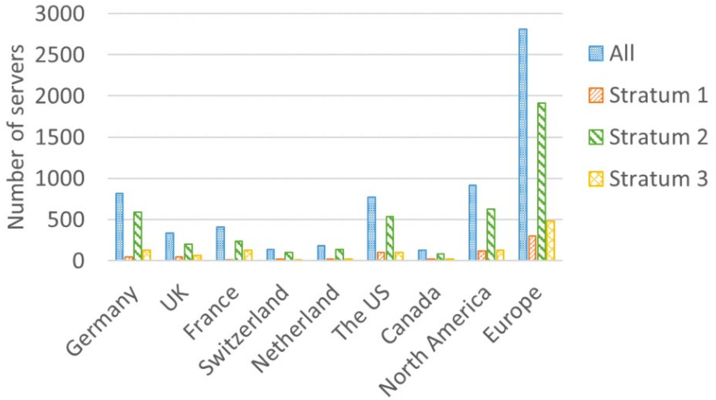


```
Non-authoritative answer:
Name:    ch.pool.ntp.org
Address: 217.147.223.78
Name:    ch.pool.ntp.org
Address: 82.197.188.130
Name:    ch.pool.ntp.org
Address: 192.33.214.47
Name:    ch.pool.ntp.org
Address: 195.141.190.190
```

What if
an attacker can manipulate
the monitoring system?

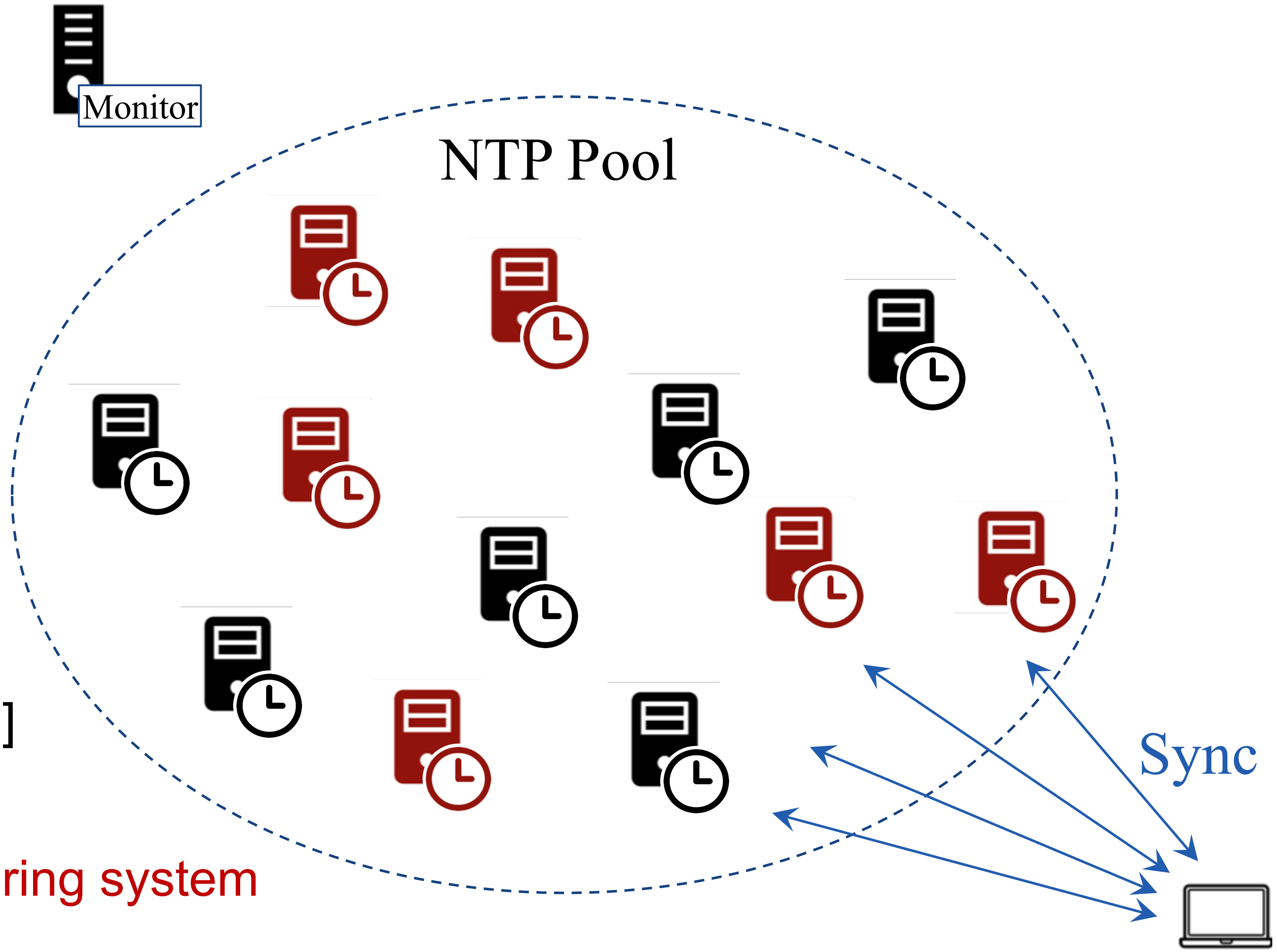
Exploiting NTP Pool Monitoring System

- Attacker needs to influence time at **many** of the servers assigned to the client



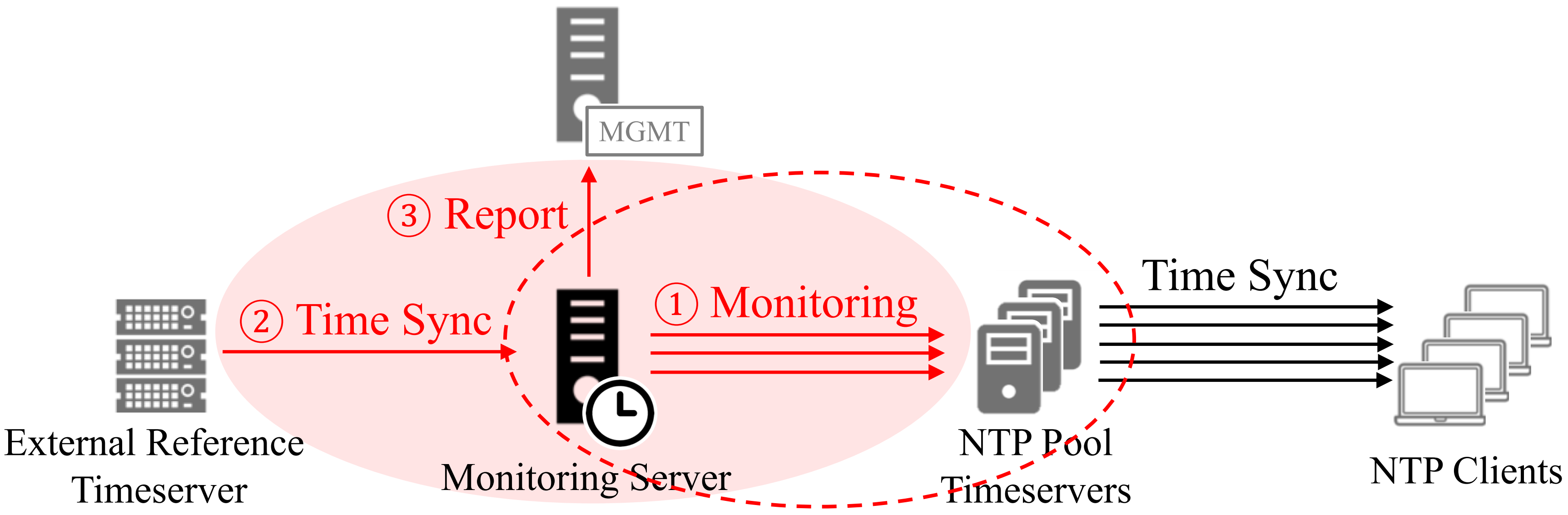
  - Inject or compromise 10s or even 100s of timeservers: Ananke[NDSS’21]

  - Or… remove legitimate timeservers from the pool by leveraging the monitoring system

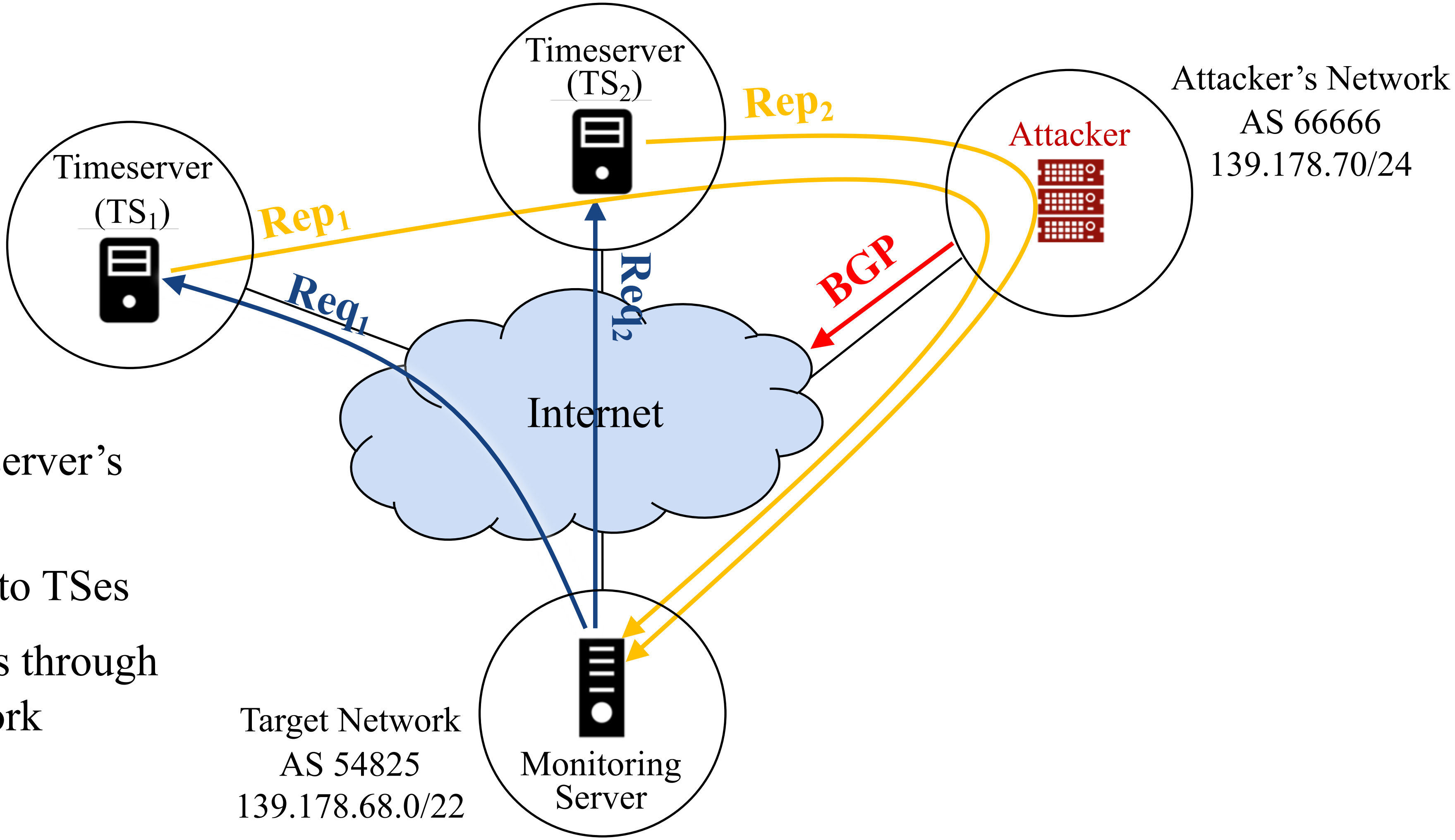


Attack Modeling

- Exploit the NTP pool monitoring system
- Exclude legitimate timeservers from the NTP pool operation
- Silent attack: the target timeservers just turn into inactive state



Injecting Asymmetric Delays to Monitoring Packets

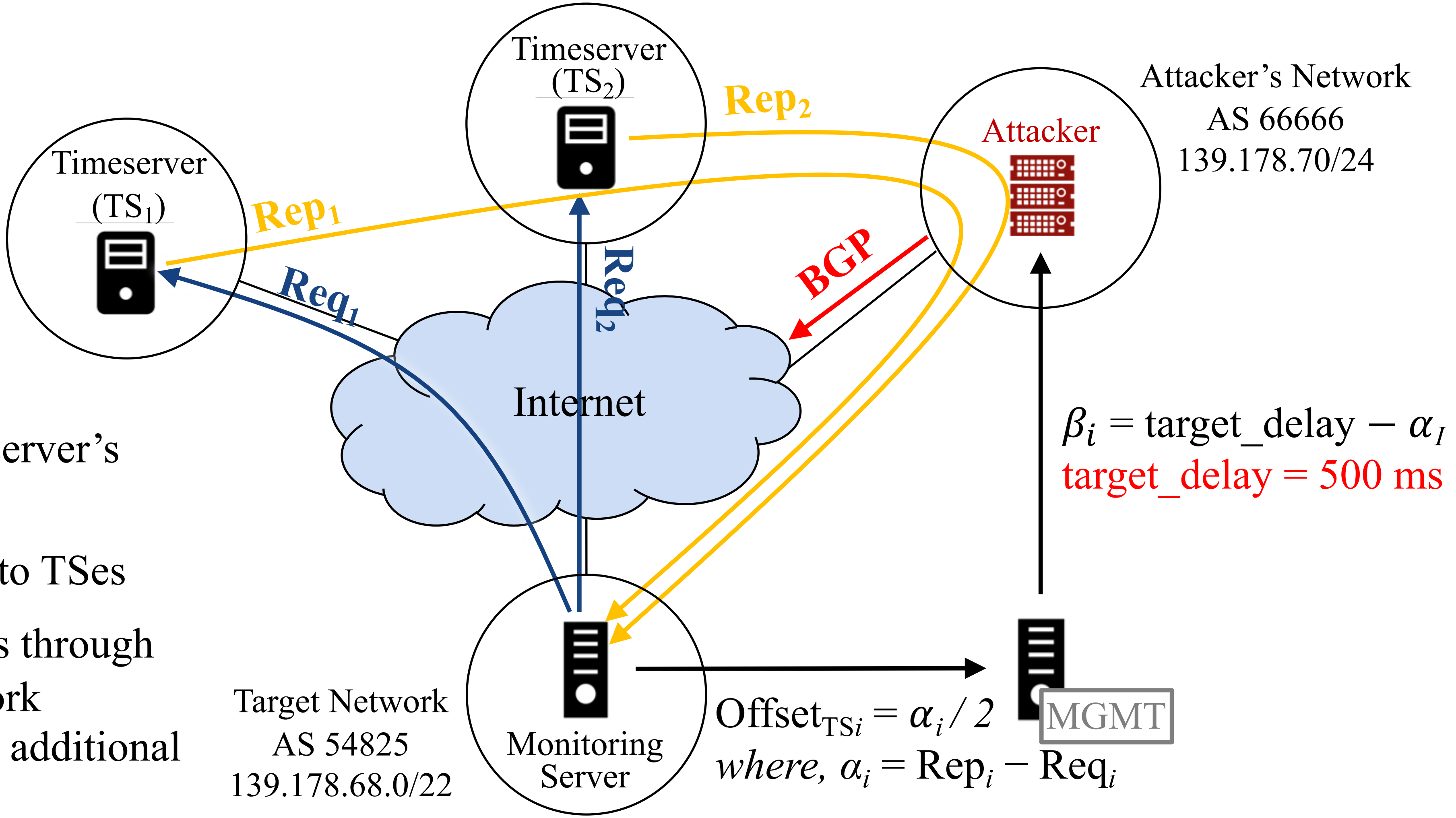


① Hijack monitoring server's IP prefix

② Send NTP requests to TSes

③ Reroute NTP replies through the attacker's network

Injecting Asymmetric Delays to Monitoring Packets



β_i = target_delay − α_I

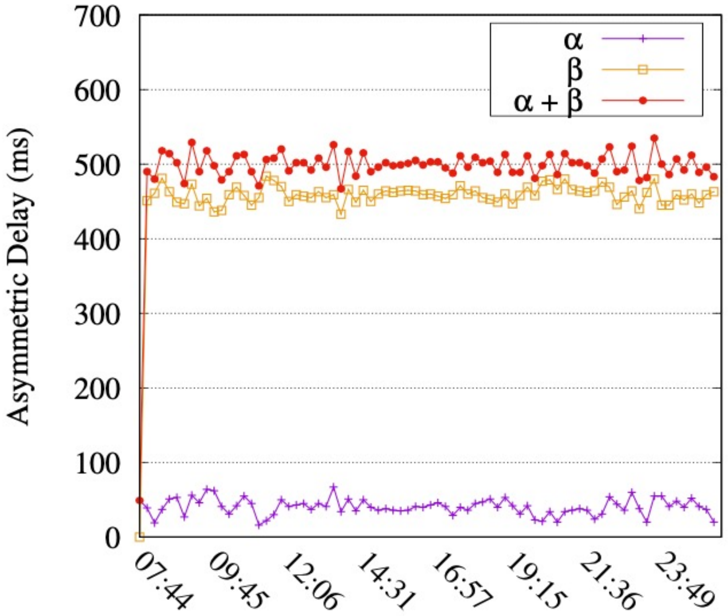
target_delay = 500 ms

Offset$_{TSi}$ = α_i / 2

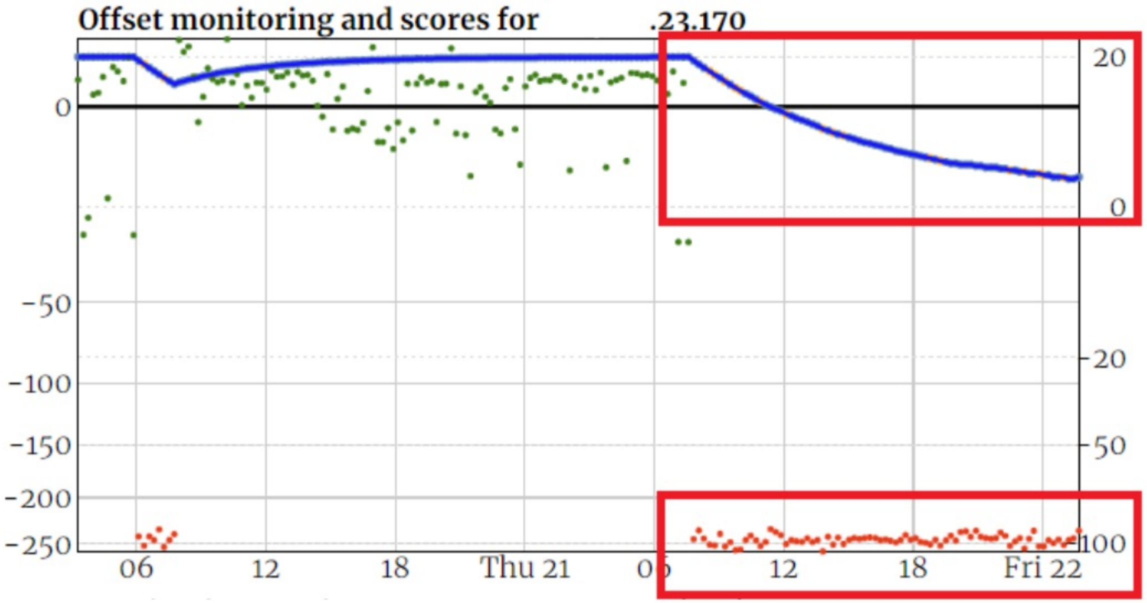
where, α_i = Rep$_i$ − Req$_i$

1. Hijack monitoring server's IP prefix
2. Send NTP requests to TSes
3. Reroute NTP replies through the attacker's network
4. Calculate and inject additional delays

Impact of Adding 500 ms of Asymmetric Delay



Achieved target delay (red line)



Logged offsets (red dots) and corresponding score drops (blue line)

More in the Paper

Did the Shark Eat the Watchdog in the NTP Pool? Deceiving the NTP Pool's Monitoring System

Jonghoon Kwon *ETH Zürich* Jeonggyu Song *Korea University* Junbeom Hur *Korea University* Adrian Perrig *ETH Zürich*

Abstract

The NTP pool has become a critical infrastructure for modern Internet services and applications. With voluntarily joined thousands of timeservers, it supplies millions of distributed (heterogeneous) systems with time. While numerous efforts have been made to enhance NTP's accuracy, reliability, and security, unfortunately, the NTP pool attracts relatively little attention. In this paper, we provide a comprehensive analysis of NTP pool security, in particular the NTP pool monitoring system, which oversees the correctness and responsiveness of the participating servers. We first investigate strategic attacks that deceive the pool's health-check system to remove legitimate timeservers from the pool. Then, through empirical analysis using monitoring servers and timeservers injected into the pool, we demonstrate the feasibility of our approaches, show their effectiveness, and debate the implications. Finally, we discuss designing a new pool monitoring system to mitigate these attacks.

1 Introduction

Time synchronization across distributed systems is essential in modern Internet services and applications, for instance in the validation of certificates [14, 16]. Accurate time is vital also for network infrastructure and its control and data plane operations, e.g., updating routing tables with a precise clock would enable dynamic congestion control and avoid routing loops [2, 36]. The Network Time Protocol (NTP) is

packet manipulation and hijacking. Nonetheless, delay attacks and compromised timeservers remain effective. Thus, redesigning NTP received attention to achieve Byzantine robustness even in the presence of adversarial timeservers [8,32]. While numerous efforts to secure NTP communication have been made, only a limited number of studies have focused on the NTP ecosystem [24, 35].

The NTP Pool Project [30], the biggest NTP ecosystem, bundles thousands of public timeservers into regional or vendor-specific domains, and provides NTP clients across the globe centralized access via the domain name servic (DNS). With the NTP pool, NTP clients enjoy reliable a available time sources. Indeed, millions of networked dev including routers, IoT devices, and Android mobile de rely on the NTP pool. Given this critical infrastructur esting research questions arise: "*What if a determined takes control over the pool?*", "*What if an attacke to remove the majority of legitimate timeservers f while keeping malicious timeservers as only sources?*". To answer these questions, we co analysis of the current NTP pool architectu vulnerabilities in its centralized manage

We explore strategic attack approa pool's health-check system [29]. Th ing system that inspects the stat It frequently sends NTP ch checks their clock accura with an incorrect time removed from the

Case Study

NTP Pool architecture
Scoring mechanism
Impact of network delay
New monitoring system

Attack Analysis

Integrity of the monitor clock
Injected monitor
Avoiding notification system

Mitigation

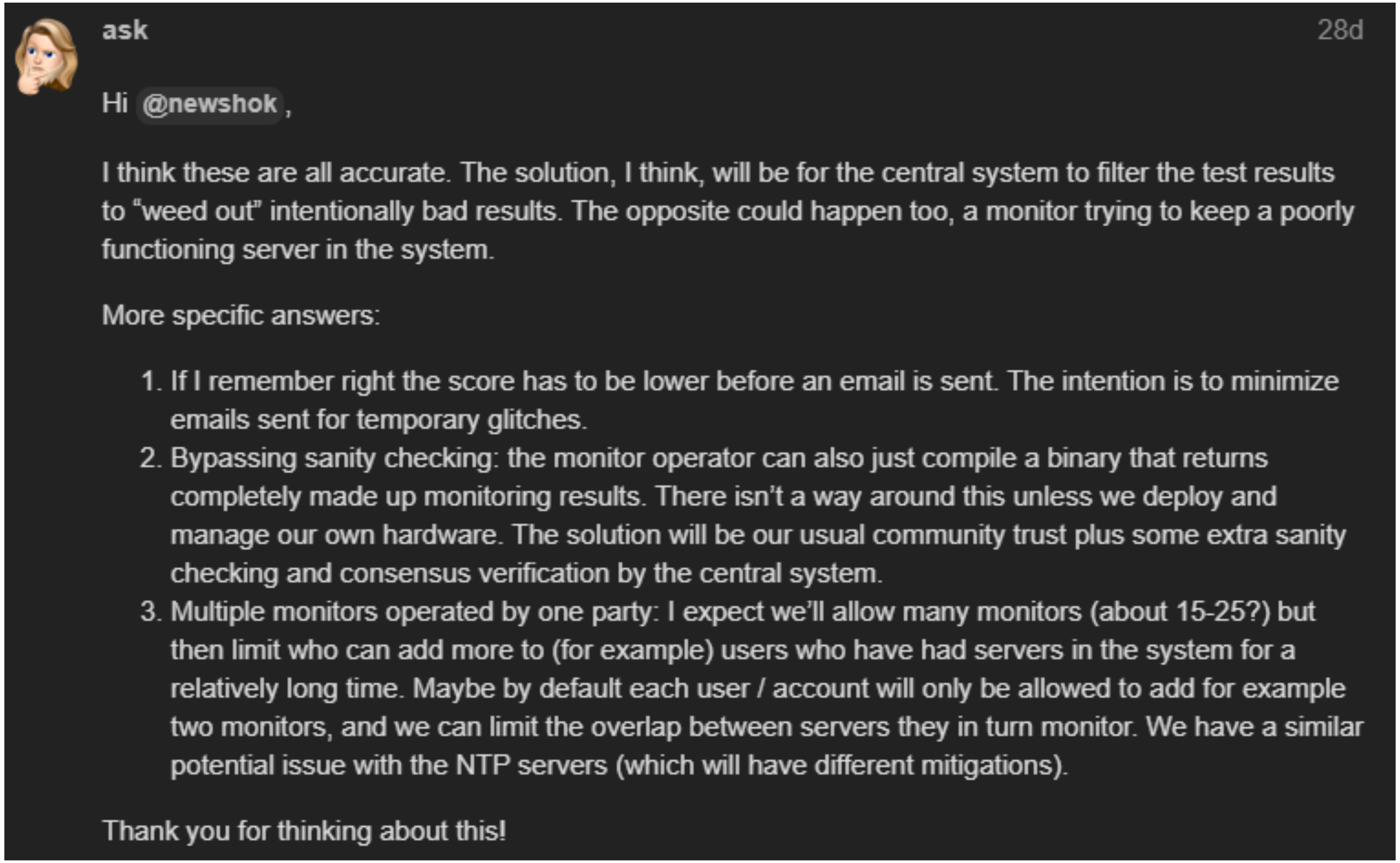
Robust reference clock
RTT-offset correlation
New scoring algorithm

Summary

- The paper provides a comprehensive analysis of the NTP pool and discloses vulnerabilities in its watchdog system
- We introduce strategic attacks exploiting the vulnerabilities and demonstrate their feasibility
- We present possible mitigations and discussion on securing the NTP pool monitoring system

Responsible Disclosure

https://community.ntppool.org/t/about-spoiling-ntp-pool-monitoring-system-report/2514



ask 28d

Hi @newshok,

I think these are all accurate. The solution, I think, will be for the central system to filter the test results to "weed out" intentionally bad results. The opposite could happen too, a monitor trying to keep a poorly functioning server in the system.

More specific answers:

1. If I remember right the score has to be lower before an email is sent. The intention is to minimize emails sent for temporary glitches.
2. Bypassing sanity checking: the monitor operator can also just compile a binary that returns completely made up monitoring results. There isn't a way around this unless we deploy and manage our own hardware. The solution will be our usual community trust plus some extra sanity checking and consensus verification by the central system.
3. Multiple monitors operated by one party: I expect we'll allow many monitors (about 15-25?) but then limit who can add more to (for example) users who have had servers in the system for a relatively long time. Maybe by default each user / account will only be allowed to add for example two monitors, and we can limit the overlap between servers they in turn monitor. We have a similar potential issue with the NTP servers (which will have different mitigations).

Thank you for thinking about this!

05.08.2022

Q&A

Jonghoon Kwon

Network Security Group

jong.kwon@inf.ethz.ch

CAB F83

Universitätstrasse 6

8092 Zürich, Switzerland

https://netsec.ethz.ch