Title Goes Here

Rafael Ortiz <rortiz12@jhu.edu>
Ramon Benitez-Pagan <ramon.benitez@jhu.edu>
Stephen Scally <sscally@jhu.edu> Cyrus Bulsara <cbulsar1@jhu.edu>

Abstract

Many papers focus on creating covert channels for the purpose of data exfiltration. That is, they attempt to remove some information from a protected network. Less common appears to be the concept of data *infiltration*, where a covert channel is established to secretly move information *inside* a protected network. Many exfiltration oriented channels make assumptions about extant arbitrary infiltration channels being available for loading the tooling necessary to establish the outbound channel. We are proposing studying and implementing an NTP-based covert channel for infiltrating unauthorized information into a secure network.

Introduction

Covert channels are traditionally classified as either storage or timing channels. These classifications came about from definitions outlined in the Pink book. These definitions are listed below:

Definition 1:

A communication channel is covert if it was neither designed or intended to transfer information at all. [Pink book section 2.1]

TODO: convert these to proper bibtex citations

Definition 2:

A communications channel is covert if it is based on "transmission by storage into variables that describe resource states." [Pink book section 2.1]

Definition 3:

Covert channels "will be defined as those channels that are a result of resource allocation policies and resource management implementation." [Pink book section 2.1]

Definition 4:

Covert channels are those that "use entities not normally viewed as data objects to transfer information from one subject to another." [Pink book section 2.1]

Definition 5:

Given a non-discretionary security policy model M and its interpretation I(M) in an operating system, any potential communication between two subjects I(S_h) and I(S_i) of I(M) is covert if and only if any communication between the corresponding subjects S_h and S_i of the model M is illegal in M [Pink book section 2.1]

After understanding the criteria of what makes a covert channel our goal within this paper is to utilize the network time protocol otherwise known as NTP to create a covert channel for data infiltration.

There are a number of network architectures that enterprises and device manufacturers utilize in order to have network connected devices synchronize their system time. We reason that most security controls within these architectures focus on ensuring that access is only allowed to appropriate network time servers, over the protocol UDP and port 123, and that public network time servers are themselves clients of higher tiered stratum layer servers. Due to the default trust associated with NTP very few operators validate the public NTP pool servers they connect to or the time data that is retrieved and propagated throughout the network from servers to clients.

In order to address these security concerns as well as control NTP delivery, enterprises have deployed dedicated vendor appliances while device manufacturers have removed the ability to change time sources on certain devices. Furthermore, adding

to the lower scrutiny of NTP usage, CVEs and exploits are typically directed at public NTP instances acting as time sources, with attacks in the form of DDoS and reflective type attacks. While it can be beneficial for an adversary to attack an internal time source as these disruptions can affect remote logins, authentication tokens, and DHCP leasing most service logging indicates clearly when a time or date synchronizations are out of skew. For an adversary attempting to keep a low profile, internal NTP attacks may create too much noise for very little gain.

We intend to show that, with the above factors in mind, ingress and egress NTP communications that are not being analyzed for correctness leaves networks and devices open to covert channel utilziation. This paper is broken down into the following sections. In the background section we will will provide basic terminogly and workings of NTP and its communication structure. Related works will review past and current analysis related to NTP covert channels. The design section will explain our network architecture for implementation as well as review our expected throughput, robustness, and detection of this channel. The implementation will demonstrate our covert channel in our lab environment. Lastly, our conclusions and future work outlines further areas of expanding the NTP covert channels based on current standards specifications as well as observations during our implementation.

Background

NTP Modes

The network time protocol(NTP) operates in one of three modes.

Primary Server

The first mode is primary server which is directly synchronized from a reference clock. Reference clocks can come from multiple sources however the most common are satillite based from GPS[1], GLONASS[2], and Galileo[3] as well as regional radio based time signals provided by MSF[4] in the UK, DCF77[5] in Germany, and WWVB[6] within the United States. A primary server is utilized by secondary servers and clients. Since the primary server derives its time from a reference clock it is also categorized as a stratum 1 server. The stratum designation signifies two items, the first is the distance that the server is from a reference clock(stratum 0), in this case one hop, and that this server provides a high level of time accuracy.

Secondary Server

The second mode is being a secondary server. This mode operates as a client to upstream primary servers and as a server to downstream clients. There is a defined maximum of 16 stratum levels and each secondary server will reflect their level depending on how far they are from a primary server. Each increased stratum level indicates a decrease time accuracy from the the higher level time source. Any system reporting a stratum level of 16 is understood to be unsynchronized.

Client

Lastly the third mode is client to which most devices fall under. A client references time from multiple available time sources to synchronize its system time.

NTP Uses

// TODO: this section can be cut down if we need space

While accurate time is useful the question remains about why it is needed for devices such as computers, phones and within local area networks. Most if not all computing type devices contain some local clock or time keeping mechanisms. Typically they are adjusted during the setup or initialization phase to the current date and time. While the local system clock is able to keep track of time, through frequency ticks, it is subject to the same interrupt process delays as other system hardware and software. Over time the missed or delayed interrupts can cause the system to fall behind real-time causing the system time to become skewed[7]. Putting this into perspective it can be seen how each device, on the network, if left to its own time keeping, would eventually fall out of sync with other devices. For a single device this may not be concerning as the time can be adjusted, however with logging, security, and within the network this can have cascading problems.

The first of these issues is the ability to schedule tasks. Within current operating systems tasks can be scheduled to run at specific dates and

times in order to perform maintenance, updates, or monitoring. These tasks can have dependencies on other scheduled tasks running or checking dates and times for file access and updates. Externally to the system other devices may depend on the scheduled jobs placing needed files or updates within a specific time frame, as time drifts between the systems this can lead to failed job processing and race conditions.

This leads to the second issue, auditing and logging. When creating log events, troubleshooting issues or auditing security logins one of the many sources referenced is the system or application logs. When reviewing the timeline of events and seeing that the system clocks are not synchronized it places extra effort in determining when systems were accessed and at what time relative to each local system clock. Instead, if the systems were synchronized, dates and times would line up accordingly and administrators could efficiently determine root causes of errors. Other network devices such as firewalls can depend on rulesets being configured to allow or deny access based on the current date and time. Since these devices are regularly deployed in pairs security access controls would show inconsistencies as devices would allow access early or delayed depending on the firewalls system time. When combining multiple device logs such as firewalls, systems and applications to audit compliance, assurance and integrity can be brought into question as numerous time and date inconsistencies have to be re-evaluated for proper alignment. With centralized log aggregation of these entries can be dropped from processing as a previous time period has been marked as processed or because the current time is determined to be in the future.

The final item is security. Systems and services that require users to authenticate need accurate dates and times across numerous systems. With the Windows operating system, which utilizes the W32Time manager, any system looking to join the Active Directory domain or obtain needed authorization Kerberos tickets needs to have highly accurate and stable system time[8]. Accurate time reduces the ability of attackers to perform replay attacks with expired Kerberos tickets which utilize time stamps to limit the life of the ticket. Additionally, the timestamps inform applications when to request a token refresh[9]. For other web protocols, such as HTTPS, time and date settings are important for validating and generating certificates. For example each website that is secured using HTTPS presents a certificate which contains a *not valid before* and *not valid after* entries which contain a date, time and timezone as seen below:

```
Signature Algorithm: sha256WithRSAEncryption
  Issuer:
    C=US, O=DigiCert Inc,
    OU=www.digicert.com,
    CN=GeoTrust RSA CA 2018
  Validity
    Not Before: Jul  1 00:00:00 2021 GMT
    Not After : Mar  7 23:59:59 2022 GMT
```

Figure 1.: https://www.jhu.edu website certificate information

This allows systems to validate the security of the presented certificate from the webserver.

NTP Public Pool

Synchronized time utilizing NTP is depended upon by numerous services and security mechanisms within typical network architectures. To support this need for accurate time, public time sources are provided by the NTP pool project[10]. Servers are allocated by DNS round robin, where each client requests a forward lookup of one of the following:

- 0.pool.ntp.org
- 1.pool.ntp.org
- 2.pool.ntp.org
- 3.pool.ntp.org

The DNS based pool set of returned NTP servers IP addresses rotates every hour. Additionally, further continent zones and sub-zones exist to provide time service asa close as possible to the intended client as seen in table 1 below.

Area	Hostname
Asia	asia.pool.ntp.org
Europe	europe.pool.ntp.org
North America	north-america.pool.ntp.org
Oceania	oceania.pool.ntp.org
South America	south-america.pool.ntp.org

Table 1: Continet Zones

// TODO: what does “score of 10” mean?

The available servers in the pool are volunteered by research organizations, companies, and individuals helping to support the project. In order to add a server to the available pool you must have a static IP address and a stable internet connection. If that criteria is met, you can configure your NTP servers with a known good stratum 1 or 2 servers and make the appropriate firewall allowances. To add the severs to the available pool, account registration is needed at ntppool.org, and after a point score of 10 has been reached through monitoring, for the specific servers they will be added to the NTP time cluster.

Related Work

List prior works

Design

// TODO: I’ll rewrite this section with my implementation details

In typical enterprise threat scenarios, data infiltration is often achieved by way of phishing emails, TLS tunnels, or shell access. From there, exfiltration may be achieved by a number of covert channels. However, in certain locked-down networks, it may not be possible to send inbound email, establish a TLS tunnel, or directly access protected machines. On the other hand, administrators of these networks typically *do* want their machines clocks to be synchronized, since Bad Things^{TM} happen when clocks drift out of sync. This presents NTP as a potential channel whereby information from the Internet can slowly leak into the inside of a protected network. We will consider two scenarios: DMZ access and direct access.

DMZ Access

In DMZ access, devices in a secure network synchronize their clocks via an NTP server that is located in a controlled DMZ. That NTP server synchronizes its clock by connecting to one or more trusted NTP pools. Any information that can survive a stratum layer* may reach devices through the DMZ.

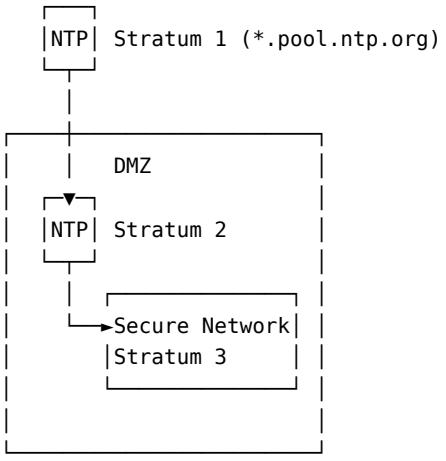

Figure 2. The DMZ Access NTP scenario, where an NTP server in a DMZ serves replies to clients in a secure inner network.

*In NTP, a stratum is a layer of devices that are the same distance from a reference clock. Reference clocks, considered a source of truth, are at stratum layer 0. Servers that rely on reference clocks are at stratum layer 1, and so on. In our DMZ scenario, if public pools are at stratum 1, then the DMZ NTP server is at stratum 2, and internal devices are at stratum 3.

Direct Access

In direct access, devices in a secure network synchronize their clocks directly via NTP to a trusted, publicly available, pool. Any information that the NTP server can pack into an NTP reply may reach devices inside this network.

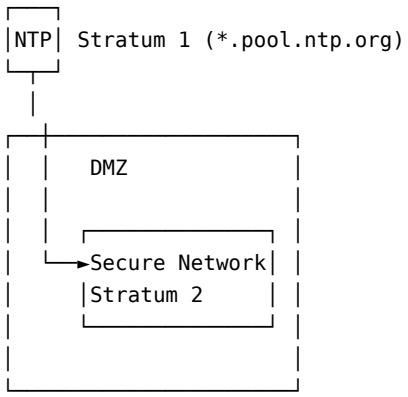

Figure 3. The Direct Access NTP scenario, where clients in a secure inner network are allowed direct access to NTP pool servers.

Throughput

Robustness

Detection

Implementation

How we implemented our test

Conclusions and Future Work

Deploying content to one service and exploiting automated idempotence to further the propagation of the exploit. You could use puppet or ansible which is configured to return a directory or file to a previously well-known hashedstate. The replacing of our covert file

References

TODO: format this with bibtex

[1] https://www.gps.gov/applications/timing/ [2] https://gssc.esa.int/navipedia/index.php/GLONASS_General_Introduction [3] https://gssc.esa.int/navipedia/index.php/Galileo_General_Introduction [4] https://www.npl.co.uk/msf-signal [5] https://www.ptb.de/cms/en/ptb/fachabteilungen/abt4/fb-44/ag-442/dissemination-of-legal-time/dcf77.html [6] https://www.nist.gov/pml/time-and-frequency-division/time-distribution/radio-station-wwvb [7] https://en.wikipedia.org/wiki/Clock_skew [8] https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/support-boundary [9] https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/jj852172(v=ws.11) [10] https://www.ntppool.org/en/