The impact of NTP security weaknesses on DNS(SEC)

Aanchal Malhotra¹
Willem Toorop² (presenter)
Benno Overeinder²
Sharon Goldberg¹

¹ Boston University
² NLnet Labs

RoN++ 15 December 2017
The impact of NTP security weaknesses on DNSSEC
Domain Name System

Working Group for Comments: 1034
: RFCS 882, 883, 973

DOMAIN NAMES - CONCEPTS AND FACILITIES

THIS MEMO

Introduction to the Domain Name System (DNS), and omits
which can be found in a companion RFC, "Domain Names -
Specification" (RFC 1035). That RFC assumes that the
function and data types constitute an official
protocol includes standard queries and
of the Internet class data formats (e.g., host
system is intentionally
implementing classes, functions, etc.)

P. Mockapetris
ISI
November 1987

root

. com

fr

dl

NL

NLNETLABS.NL

SURFNET.NL

The impact of NTP security weaknesses on DNS(SEC)
NTP Weaknesses

Attacking NTP’s Authenticated Broadcast

Aanchal Malhotra
Boston University
aanchal4@bu.edu

Sharon Goldberg
Boston University
goldbe@cs.bu.edu

ACT

Two attacks on the Network Time Protocol (NTP)’s
tightly-authenticated broadcast mode. First, we
describe a broadcast client to a specific time. Sec-
to a denial-of-service (DoS) attack that al-
attracker to prevent a broadcast client from
system clock: to do this, the attacker
single malformed broadcast packet per
an DoS attack also applies to all other
ephemeral’ or ‘preemptable’ (including
We then use network measure-
that NTP’s broadcast and other
open-source NTP implementation’s capability to provide
We present an on-path
broadcast mode (CVF)
unsuccessful
attacks on the
NTP client and one of its servers, and (2) off-path
where the attacker can be anywhere on the
access to
up to
and
questionable
attacks.
requests.
protocol
the
range
requests.
requests.
range
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
requests.
NTP Weaknesses


How does DNS depend on time?

TTL (TIME TO LIVE) = TIME SPAN
How do software implementations deal with time spans?

```c
struct RRset_t {
    uint8_t dname;
    uint16_t rrtype;
    uint16_t rrclass;
    struct timeval expiry;
    void *rdata[];
};

if (gettimeofday(&rrset->expiry, NULL))
    perror("Could not get time of day");
else
    rrset->expiry.tv_sec += ttl;
```
How do software implementations deal with time spans?

TIME SPAN TRANSLATED TO TIME STAMP
FROM SYSTEM TIME ➔ UPDATED BY NTP

```c
struct RRset_t {
    uint8_t dname;
    uint16_t rrtype;
    uint16_t rrclass;
    struct timeval expiry;
    void *rdata[];
};

if (gettimeofday(&rrset->expiry, NULL))
    perror("Could not get time of day");
else
    rrset->expiry.tv_sec += ttl;
```
Why is this bad?

TIME SPAN TRANSLATED TO TIME STAMP FROM SYSTEM TIME \(\rightarrow\) UPDATED BY NTP

- NTP vulnerabilities [1, 2, 3] can be leveraged for off-path attacks on DNS cache:
  - Cache-expiration attack (Time shifted forward)
  - Cache-sticking attack (Time shifted backwards)
Recommendation

- Not a protocol problem 😊
- Deal with implementations ONLY!

```c
struct RRset_t {
    uint8_t dname;
    uint16_t rrtype;
    uint16_t rrclass;
    struct timespec expiry;
    void *rdata[];
};

if (clock_gettime(CLOCK_MONOTONIC_RAW, &rrset->expiry))
    perror("Could not get time of day");
else
    rrset->expiry.tv_sec += ttl;
```

● Unspecified starting point
● Monotonically increasing
● not subject to NTP adjustments
● or by adjustments from adjtime
Recommendation

- Not a protocol problem 😊
- Deal with implementations ONLY!

Terminology

A **CLOCK** IS A FUNCTION THAT MAPS TIME TO A **CLOCK TIME VALUE**

Use **RAW CLOCK TIME STAMP** INSTEAD OF **REAL CLOCK TIME STAMP**

draft-aanchal-time-implementation-guidance

Willem Toorop (NLnet Labs)  The impact of NTP security weaknesses on DNS(SEC) 11/22
Certificate of Authenticity

Let it be known that this guitar was handmade by Lynn Ellsworth and Patrick Coleman of Lynn Ellsworth Guitars LLC in 2014.

This particular guitar is the very first Les Paul style guitar from the Relic Master Series.

The single cutaway Les Paul style body is made from a weathered Redwood. The hardware is black with a Gold bridge. The neck pickup is a Seymour Duncan Quarter Pound, the bridge pickup is a DiMarzio Red Dragon.

Gibson with a Rosewood fingerboard.
How does DNSSEC depend on time?

Expiration & Inception as Wall Clock Time Stamps
How do software implementations deal with wall clock time stamps?

```c
struct timeval now;

if (gettimeofday(&now, NULL))
    perror("Could not get time of day");

else if (now < rrset->rrsig.inception)
    verify_error("Not yet valid");

else if (now > rrset->rrsig.expiration)
    verify_error("Not valid anymore");
```
Recommendation

- Fundamental problem with the protocol 😞
- Have to use real clock time (i.e. system time)

The only solution

- Fix Network Time Protocols 😊

draft-aanchal-time-implementation-guidance
Recommendation

- Fundamental problem with the protocol 😞
- Have to use real clock time (i.e. system time)

The only solution

- Fix Network Time Protocols 😊

Impact?

- Denial of Service attacks
- Disable DNSSEC by shifting before 15 July 2010

draft-aanchal-time-implementation-guidance
Measure the attack surface
RIPE ATLAS

- Which resolvers run NTP?
  Target probe’s resolvers (DHCP?)
Measure the attack surface
RIPE ATLAS

• Which resolvers run NTP?
Measure the attack surface
RIPE ATLAS

- Which resolvers run NTP?
  Target probe’s resolvers (DHCP?)
- Target resolvers with public IPs +
- Try to discover IPs

```
$ dig o-o.myaddr.l.google.com. TXT +short
62.216.31.207

$ dig whoami.akamai.net A +short
194.109.133.206
```

Willem Toorop (NLnet Labs)
Measure the attack surface
RIPE ATLAS

Measurements done from 21 till 27 October 2017

<table>
<thead>
<tr>
<th></th>
<th># resolvers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>+- 18500</td>
</tr>
<tr>
<td>With public IP resolvers</td>
<td>8244</td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Answering NTP time queries</td>
<td>2021 (24.5%)</td>
<td></td>
</tr>
<tr>
<td>Answering NTP control queries from public internet</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>Answering NTP control queries from NLNOG RING node from same ASN</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td><strong>Total answering NTP control queries</strong></td>
<td><strong>101 (1.23%)</strong></td>
<td></td>
</tr>
</tbody>
</table>
# Measure the attack surface

## Open Resolvers

From August 2017 list of the Open Resolver Project

<table>
<thead>
<tr>
<th># resolvers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
</tr>
<tr>
<td>Targeted</td>
</tr>
<tr>
<td>Still answering DNS queries (Nov 2017)</td>
</tr>
<tr>
<td>Answered REFUSED (authoritatives)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th># resolvers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total 16.5M</td>
</tr>
<tr>
<td>Targeted 6.5M</td>
</tr>
<tr>
<td>Still answering DNS queries (Nov 2017) 2.3M</td>
</tr>
<tr>
<td>Answered REFUSED (authoritatives) 1.7M</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Open resolvers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answering NTP queries</td>
</tr>
<tr>
<td>Answering NTP control queries</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Open resolvers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answering NTP queries 600K</td>
</tr>
<tr>
<td>Answering NTP control queries 0.93%</td>
</tr>
<tr>
<td>Answering NTP control queries 1.23% on ATLAS</td>
</tr>
</tbody>
</table>

Willem Toorop (NLnet Labs)
De impact of NTP security weaknesses on DNS(SEC)

- Sophisticated attacks possible
- Script-kiddie attacks less so (DOS of DNSSEC resolvers)
- Attack surface around 1% of resolvers
- Software takes a common approach towards (wall/real) clock time stamps and time spans
- Not just RRset TTLs (also network timeouts etc.)