Classical internet applications
Lab session: DNS-1

Anatoly Tykushin Security and Network Engineering, a.tykushin@innopolis.ru

September 14, 2016

Contents

1 Why is it wise to use a signature to check your download?

A hash value processed on the downloaded file is a way to make sure that the content is transferred well and has not been damaged during the download process (authenticity and integrity).

2 Which kind of signature is the best one to use? Why?

One quick and easy way to verify the integrity of a downloaded file is to use various checksum tools (e.g., md5sum, sha256sum, cksum) to compute and compare checksums (e.g., MD5, SHA or CRC). However, checksums are vulnerable to collision attacks, and also cannot be used to verify the authenticity (i.e., owner) of a file.

Second way is to use asymmetric cryptography. By encrypting a message's hash value with the sender's private key generates a digital signature of a message. Now anyone can verify the signature by decrypting it using sender's public key. If this reveals the correct hash value for the message, so it is the only possible sender since he is the only one to have the private key that fits public key. Now anyone can find out about the encrypted message's hash value.

To sum it up, using combination of asymmetric cryptography with hash algorithms is the best way to support message authentication and integrity checking. One of the best example is PGP (pretty good privacy).

In appendix A you can see a list of commands and their output which was used to verify signatures of unbound sources.

3 Why are caching-only name servers still useful?

DNS cache servers fundamentally provide options described in table 1

Option	Value
Avaliabiity	Should only respond to queries that originate from a local user base
Types of query that it should answer	recursive queries
Design of a software implementation	Must maintain state because of recursive queries and an internal cache
Records that it should attempt to resolve	Should attempt to resolve any request

Table 1: DNS Cache servers fundemental providing abilities

Also DNS cache servers (e.g. Unbound) are:

- lightweight
- high performance
- adding of software diversity
- single-purpose server
- securable
- manageable and portable

To sum it up, caching-only nameservers are being used for some reasons:
Firstly, they store queries and prevent excessive traffic and reduces query time.
Secondly, for example, you have NAT over your subnet. Some free DNS caching services like OpenDNS or Google's public DNS normally limit the maximum number of connections allowed from specific IP address. In that case it would be useful to deploy own caching nameserver.
Also it would be useful to mention that size of cache in memory shouldn't be more than

$$V_{cache} \leq 1024 Mbytes \tag{1}$$

because it wouldn't provide efficiency due to CPU load.

4 Unbound configuration

Created configuration file of caching-only unbound dns-server is shown in the listing below.

```
server:
verbosity: 2 # sets verbosity level to show detailed information in logs
port: 53
interface: 188.130.155.38 # interface to listen

do-ip4: yes
do-ip6: no                 # allow ip version 4 not 6
do-udp: yes
do-tcp: yes                # allow both udp and tcp
access-control: 188.130.155.0/24 allow      # acl config
access-control: 127.0.0.0/8 allow           # acl for localhost
#access-control: 0.0.0.0/0 refuse
username: "unbound"                         # run unbound as user
directory: "/usr/local/etc/unbound"         # where to store files

logfile: "dns.log"                          # name of log file in "directory"
use-syslog: no
hide-version: yes # do not to show information of unbound's version

remote-control:
control-enable: yes

forward-zone: #forward all queries to google
```

```
name: "."
  forward-addr: 8.8.8.8 #google.com

forward-zone: # forward queries to the zone "st5.os3.su" to the
name: "st5.os3.su"
  forward-addr: 188.130.155.38@5355   # ip4 + port to the authoritative server
```

The action **allow** gives access to clients from subnet, written in access-control record. It gives only access for recursion clients (which is what almost all clients need). Nonrecursive queries are refused.
The **allow** action does allow nonrecursive queries to access the local-data that is configured. The reason is that this does not involve the unbound server recursive lookup algorithm, and static data is served in the reply. This supports normal operations where nonrecursive queries are made for the authoritative data. For nonrecursive queries any replies from the dynamic cache are refused.

So, yes, this configuration enables recursive queries for users from subnet 188.130.155.0/24.

5 Testing configuration

Why do the *unbound-checkconf* program returns a value?
This program returns a value to use it in script writing, or using in some other software as it provides good abilities for analysis.

But writing log to the screen of the terminal is very useful for users so it would be a wise decision to support both logging and return value abilities.

6 Changes you made to configuration to allow remote control

To use remote control as a tool to send commands to the server during runtime. To enable this commands we have to do the following steps:

Step 1: ***unbound-control-setup*** to generate the necessary TLS key files (they are put in the default install directory).

Step 2: add this option to configuration file
remote-control:
control-enable: yes
control-port: 34567 # but to change this you have to restart unbound

Also you can setup options like:

control-interface: ip address or path Give IPv4 or IPv6 addresses or local socket path to listen on for control commands. By default localhost (127.0.0.1 and ::1) is listened to. Use 0.0.0.0 and ::0 to listen to all interfaces. If you change this and permissions have been dropped, you must restart the server for the change to take effect.

control-port: port number The port number to listen on for IPv4 or IPv6 control interfaces, default is 8953. If you change this and permissions have been dropped, you must restart the server for the change to take effect.

other options: that enable you to use certificates

To see how it works

```
tyvision@st5: sudo netstat -tlpun
[sudo] password for tyvision:
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:34567         0.0.0.0:*               LISTEN      28103/unbound
tcp        0      0 188.130.155.38:5355     0.0.0.0:*               LISTEN      28103/unbound
```

7 What other commands/functions does unbound-control provide?

Unbound-control is a unbound remote server control utility. performs remote administration on the unbound DNS server. It reads the configuration file (but doesn't rewrite it), contacts the unbound server over SSL sends the command and displays the result.

It's more likely to list commands provided by **unbound-control** it the table. Table 2 contains most frequently used commands.

You can see example of usage below

Option	Description
start	Start the server. Simply execs unbound
stop	Stop the server. The server daemon exits.
reload	Reload the server. This flushes the cache and reads the config file fresh
verbosity	Change verbosity value for logging
local zone *name type*	Add new local zone with name and type
load cache	The contents of the cache is loaded from stdin
dump cache	The contents of the cache is printed in a text format to stdout
forward add [+i] *zone addr*	Add a new forward zone to running unbound
forward remove [+i] *zone*	Remove a forward zone from running unbound

Table 2: Unbound-control commands

```
tyvision@st5:/usr/local/etc/unbound$ sudo unbound-control verbosity 3
ok
```

8 What do you need to put in resolv.conf (and/or other files) to use your own name server?

As already mentioned file ***resolv.conf*** is automatically generated by resolv-conf utility. To add a record to the ***resolv.conf*** file you have to write **nameserver** ***[IP address of NSD]*** in a ***/etc/resolvconf/resolv.conf.d/base*** file.

Also, you should reconfigure network-manager. It was done by GUI interface (because it's easy enough). After configuration network-manager was reloaded (resolv.conf was changed, but record about nameserver was still there) and simple check was made:

```
    tyvision@st5:~/Labs/inno_labs_cia/Lab4_report$ nslookup google.com
Server: 188.130.155.38
Address: 188.130.155.38#53

Non-authoritative answer:
Name: google.com
Address: 83.169.197.246
Name: google.com
Address: 83.169.197.242
Name: google.com
Address: 83.169.197.245
Name: google.com
Address: 83.169.197.240
Name: google.com
Address: 83.169.197.247
Name: google.com
Address: 83.169.197.251
Name: google.com
Address: 83.169.197.244
Name: google.com
Address: 83.169.197.243
Name: google.com
Address: 83.169.197.248
Name: google.com
Address: 83.169.197.249
Name: google.com
Address: 83.169.197.250
Name: google.com
Address: 83.169.197.241

tyvision@st5:~/Labs/inno_labs_cia/Lab4_report$ drill google.com @188.130.155.38
;; ->>HEADER<<- opcode: QUERY, rcode: NOERROR, id: 51502
;; flags: qr rd ra ; QUERY: 1, ANSWER: 12, AUTHORITY: 0, ADDITIONAL: 0
;; QUESTION SECTION:
;; google.com. IN A

;; ANSWER SECTION:
google.com. 286 IN A 83.169.197.246
google.com. 286 IN A 83.169.197.242
google.com. 286 IN A 83.169.197.245
google.com. 286 IN A 83.169.197.240
google.com. 286 IN A 83.169.197.247
google.com. 286 IN A 83.169.197.251
google.com. 286 IN A 83.169.197.244
google.com. 286 IN A 83.169.197.243
google.com. 286 IN A 83.169.197.248
google.com. 286 IN A 83.169.197.249
google.com. 286 IN A 83.169.197.250
google.com. 286 IN A 83.169.197.241

;; AUTHORITY SECTION:

;; ADDITIONAL SECTION:

;; Query time: 0 msec
;; SERVER: 188.130.155.38
;; WHEN: Tue Sep 13 21:05:00 2016
;; MSG SIZE  rcvd: 220
```

9 Forward mapping zone

Log of the forward zone is shown in the list below:

```
================================== NSD zone file =================================
==================================================================================

$ORIGIN st5.os3.su.
$TTL 1800
@       IN      SOA     st5.os3.su.      godofzone.st5.os3.su. (
                        2014070312        ; serial number (was 201407201)
                        3600                    ; refresh
                        900                     ; retry
                        1209600                 ; expire
                        1800                    ; ttl
                        )
; Name servers
                    IN      NS      sne1.st5.os3.su.
                    IN      NS      sne2.st5.os3.su.
; Mail exchange
                    MX 10 mail1
                    MX 20  thegod
; A records for name servers
sne2                 IN      A       188.130.155.38
sne1                 IN      A       188.130.155.36
sne3  IN   A    188.130.155.46
@                    IN      A       188.130.155.38

; Canonical names
numbaone  CNAME  sne1
godmail  CNAME  thegod.st5.os3.su.

; Additional A records
```

10 Delegation

If Azat had not yet implemented the delegation, what information would you need to give him so that he can implement it?

I have to tell him this data:

domain: st5.os3.su

ip: 188.130.155.38

nameserver: ns1.os3.su ; the next name server points to ns1 in the os3.su zone above

11 Important requirements

What important requirement is not yet met for your subdomain?

In the lists below you can see what important requirements do we have for our subdomain. You need the following records in your hosts DNS:

- two Name Server (NS) records pointing to the authoritative name servers for your sub-domain
- Address (A) records for the sub-domain name servers

And you need to provide a pair of DNS name servers for your sub-domain. They need to serve the following records:

- a Start of Authority (SOA) record for the sub-domain
- two or more (NS) records
- (A) records for the sub-domain name servers
- All zones should have secondary DNS servers

Appendices

A Checking signatures

```
tyvision@st5:~/Downloads$ gpg  unbound-1.5.9.tar.gz.asc
gpg: assuming signed data in ‘unbound-1.5.9.tar.gz’
gpg: Signature made Thu 09 Jun 2016 03:11:06 PM MSK using RSA key ID 7E045F8D
gpg: Can’t check signature: public key not found


tyvision@st5:~/Downloads$ gpg -q --keyserver hkp://hkps.pool.sks-keyservers.net --recv-key 07E045F8D
gpg: requesting key 7E045F8D from hkp server hkps.pool.sks-keyservers.net
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u


gpg --edit-key wouter@nlnetlabs.nl trust
gpg (GnuPG) 1.4.20; Copyright (C) 2015 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.


This key was revoked on 2016-08-16 by RSA key 7E045F8D W.C.A. Wijngaards <wouter@nlnetlabs.nl>
pub  4096R/7E045F8D  created: 2014-06-16  revoked: 2016-08-16  usage: SCEA
                     trust: unknown       validity: revoked
[ revoked] (1). W.C.A. Wijngaards <wouter@nlnetlabs.nl>

This key was revoked on 2016-08-16 by RSA key 7E045F8D W.C.A. Wijngaards <wouter@nlnetlabs.nl>
pub  4096R/7E045F8D  created: 2014-06-16  revoked: 2016-08-16  usage: SCEA
                     trust: unknown       validity: revoked
[ revoked] (1). W.C.A. Wijngaards <wouter@nlnetlabs.nl>

Please decide how far you trust this user to correctly verify other users’ keys
(by looking at passports, checking fingerprints from different sources, etc.)

  1 = I don’t know or won’t say
  2 = I do NOT trust
  3 = I trust marginally
  4 = I trust fully
  5 = I trust ultimately
  m = back to the main menu

Your decision? 5
Do you really want to set this key to ultimate trust? (y/N) y

This key was revoked on 2016-08-16 by RSA key 7E045F8D W.C.A. Wijngaards <wouter@nlnetlabs.nl>
pub  4096R/7E045F8D  created: 2014-06-16  revoked: 2016-08-16  usage: SCEA
                     trust: ultimate      validity: revoked
[ revoked] (1). W.C.A. Wijngaards <wouter@nlnetlabs.nl>
Please note that the shown key validity is not necessarily correct
unless you restart the program.


gpg unbound-1.5.9.tar.gz.asc
gpg: assuming signed data in ‘unbound-1.5.9.tar.gz’
gpg: Signature made Thu 09 Jun 2016 03:11:06 PM MSK using RSA key ID 7E045F8D
gpg: checking the trustdb
gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model
gpg: depth: 0  valid:   2  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 2u
gpg: Good signature from "W.C.A. Wijngaards <wouter@nlnetlabs.nl>"
Primary key fingerprint: EDFA A3F2 CA4E 6EB0 5681  AF8E 9F6F 1C2D 7E04 5F8D
```