Domain name resolution with DNS over HTTPS

Secure Paths

Lead Image © Sergeiu Bertasius, 123RF.com

Article from ADMIN 53/2019

By Thorsten Scherf

The new DNS over HTTPS standard from the Internet Engineering Task Force is meant to eliminate some of the known vulnerabilities of the Domain Name System.

Domain name system security extensions (DNSSEC) was meant to solve many of the known security problems in the domain name system (DNS) protocol, but it has not really taken off and is rarely used in practice, not least because of the DNS extension's complexity. For example, if you look at an end user, a recursive DNS request is usually made to the DNS resolver at the user's Internet service provider (ISP), because the browser itself does not know the IP address of a particular website.

The information presumably is not already stored locally or in a cache ISP-side, so the ISP takes care of responding to the DNS request and forwards it through various other DNS servers until the request arrives at the server that has a corresponding entry in its own DNS zone file, which allows it to answer the request. The response is then returned to the requesting DNS server, where it is cached for a certain period of time for further requests and is also sent to the requesting client.

Listing 1 shows a simplified example of how a DNS request ultimately reaches the DNS server responsible for a particular domain. Because the requests reach the DNS resolver in plain text, the resolver can log this information for later use. Whether the information is sold to interested customers or simply discarded is up to the operator of the DNS resolver. The DNS resolver from Google (8.8.8.8 and 8.8.4.4), for example, logs various information either temporarily or permanently [1].

Listing 1

DNS Request

```
# dig +trace www.redhat.com
.           7743 IN NS h.root-servers.net.
.           7743 IN NS i.root-servers.net.
.           7743 IN NS j.root-servers.net.
com.                172800 IN NS l.gtld-servers.net.
com.                172800 IN NS j.gtld-servers.net.
com.                172800 IN NS b.gtld-servers.net.
redhat.com.         172800 IN NS ns2.redhat.com.
redhat.com.         172800 IN NS ns3.redhat.com.
redhat.com.         172800 IN NS ns1.redhat.com.
redhat.com.         172800 IN NS ns4.redhat.com.
www.redhat.com.     3600 IN CNAME ds-www.redhat.com.edgekey.net.
```

Encrypting the DNS request itself is also desirable so that not everyone can see the request that has just been made. Encryption is also important because the DNS resolver might have to forward the request, including the IP address from which it originally came, to various other DNS servers. However, because this process takes place in plain text, anyone can read the information and establish a relationship between the query and the source IP.

Internet giant Cloudflare offers what it refers to as a trusted recursive resolver (TRR) [2] service (1.1.1.1 and 1.0.0.1), which comes complete with a Privacy First guarantee. Cloudflare promises neither to sell the DNS data nor to log the IP addresses of the requests and offers a DNS over HTTPS (DoH) endpoint. All DNS requests are sent to a specific server by HTTPS – in this case, to Cloudflare's DNS resolver, which then takes care of responding to the request.

In these iterative DNS queries, the DNS server tries to reveal as little information as possible. For example, if the name *www.redhat.com* is to be resolved into an IP address, then it is sufficient for the DNS query to one of the DNS root servers merely to request information on the DNS server responsible for the *.com* top-level domain. The root server does not need to know for which second-level domain the original request was made. This technique, known as "DNS Query Name Minimisation to Improve Privacy," is specified in RFC 7816 [3].

Current Firefox and Google browsers have supported DoH for some time. In Firefox v62, DNS over HTTPS has to be enabled manually. Mozilla is currently running some tests with selected beta users, for whom DoH is already preset in the Firefox browser [4]. It remains to be seen whether this setting will be generally implemented in the future.

The required settings are made after entering *about:config* in the Firefox address bar. Decide for yourself whether you want to use Cloudflare as a TRR or trust another DNS server with a DoH endpoint instead. The `curl` GitHub site [5] has a list of public DNS servers that all support DoH.

If you want to send all DNS requests through a DoH endpoint at the operating system level, you need to install a proxy. I use the `cloudflared` daemon for this, which is available in various package formats for different platforms and, of course, as source code [6]. The first step is to install the RPM package. To install on Fedora 29, enter:

```
# dnf install https://bin.equinox.io/c/VdrWdbjqyF/cloudflared-stable-linux-amd64.rpm
```

Because you will want the proxy to run on a non-privileged port, you can create a separate service account, under which the proxy then runs:

```
# useradd -s /usr/sbin/nologin -r -M cloudflared
# chown cloudflared:cloudflared /usr/local/bin/cloudflared
```

For the service to mesh seamlessly with systemd, you need to create a suitable `/etc/systemd/system/cloudflared.service` Unit file (Listing 2). The `/etc/default/cloudflared` configuration file,

Listing 2

Systemd Unit

```
[Unit]
Description=Cloudflared DNS over HTTPS proxy
After=syslog.target network-online.target
[Service]
Type=simple
User=cloudflared
EnvironmentFile=/etc/default/cloudflared
ExecStart=/usr/local/bin/cloudflared $CLOUDFLARED_OPTS
Restart=on-failure
RestartSec=10
KillMode=process
[Install]
WantedBy=multi-user.target
```

```
CLOUDFLARED_OPTS="proxy-dns --upstream https://1.1.1.1/dns-query --upstream https://1.0.0.1/dns-query --port 5053"
```

points to the DoH endpoints offered by Cloudflare and specifies that the proxy will run on port 5053. If you prefer to use a different DoH server, enter one of the other public DoH servers from the list mentioned earlier [5]. The service can then be started easily with the `systemctl` command:

```
# systemcel enable --now cloudflared
```

The reason `cloudflared` runs on port 5053 is simply that I already run Dnsmasq on port 53 as a local DNS resolver and caching server. In the `/etc/dnsmasq.conf` configuration file, `cloudflared` is simply added as an upstream server:

```
# grep ^server' /etc/dnsmasq.conf
server=127.0.0.1#5053
```

All DNS queries at the operating system level are then sent to Dnsmasq first. Only if no entry exists in the local cache is the request forwarded to `cloudflared` on port 5053:

```
# dig www.redhat.com
ds-www.redhat.com.edgekey.net.
ds-www.redhat.com.edgekey.net.globalredir.akadns.net.
e3396.dscx.akamaiedge.net.
104.124.139.61
```

If you do not operate a local DNS caching server, you can of course also run `cloudflared` directly as a privileged service on port 53, but this means sacrificing the local DNS cache.

Conclusions

Even though some people might not relish the idea of sending DNS requests directly from Firefox to a central DoH endpoint, DoH is an improvement on the current situation. Mobile devices, which constantly use other DNS servers, can especially benefit from the use of a trusted DNS server and the ability to communicate with it in encrypted form.

The decision between the Cloudflare DNS server and some other offering is a question of personal taste; however, Cloudflare's privacy statement offers huge incentives for coming down in favor of this DNS resolver. For those of you who would like to look into the DNS over HTTPS protocol more closely, read the corresponding RFC [7].

Infos

1. Google DNS Privacy: https://developers.google.com/speed/public-dns/privacy
2. Cloudflare name server: https://1.1.1.1/dns/
3. DNS Query Name Minimisation to Improve Privacy RFC 7816: https://tools.ietf.org/html/rfc7816
4. DoH in Firefox: https://blog.mozilla.org/futurereleases/2018/09/13/dns-over-https-doh-testing-on-beta/
5. Public DoH endpoints: https://github.com/curl/curl/wiki/DNS-over-HTTPS#publicly-available-servers
6. cloudflared proxy: https://developers.cloudflare.com/argo-tunnel/downloads/
7. DNS over HTTPS (DoH) RFC 8484: https://tools.ietf.org/html/rfc8484

The Author

Thorsten Scherf is a Principal Consultant for Red Hat EMEA. You can meet him as a speaker at conferences. He is also a keen marathon runner whenever time and his family permit.

Buy this article as PDF

Express-Checkout as PDF		Price $2.95 (incl. VAT)	Check out with PayPal The safer, easier way to pay

Buy ADMIN Magazine

SINGLE ISSUES	Print Issues	Digital Issues	
SUBSCRIPTIONS	Print Subs	Digisubs	
TABLET & SMARTPHONE APPS	Download on the App Store	GET IT ON Google play US / Canada	GET IT ON Google play UK / Australia

Related content

Infrastructure as Code with Terraform

Application releases can take place several times a day. Terraform helps you roll out virtual machines automatically in your data center or in the cloud, and you adapt the manual only when it changes.

more »

DNS name resolution with HTTPS

Now that web content is encrypted by HTTPS, the underlying name resolution is often unprotected. We look at the classic DNS protocol and investigate whether DNS over HTTPS could be the solution to ensure the confidentiality of DNS requests.

more »

Distributed denial of service attacks from and against the cloud

In some particularly sophisticated DDoS attacks, the attackers rely on and target the cloud, allowing attackers to work around conventional defense mechanisms. We explain how a DDoS attack in the cloud works, and how you can defend against it.

more »

Solving the security problems of encrypted DNS

DNS encryption offers WiFi users good protection in public spaces; however, in the enterprise, it prevents the evaluation and filtering of name resolution.

more »

Denial of Service in the Cloud

In some particularly sophisticated DDoS attacks, the attackers rely on and target the cloud, which allows them to work around conventional defense mechanisms. We explain how a DDoS attack in the cloud works, and how you can defend against it.

more »

comments powered by Disqus

© 2022 Linux New Media USA, LLC – Legal Notice

Topics

12.04 LTS 16 cores 8 cores AI AMD AMD-V AMI Active Directory Administration Amazon AWS Amazon CloudFront Amazon Machine Images Anaconda Analytics Ansible Apache Apache Deltacloud Apache benchmarking tool ab acceleration acquisition admin tools agedu alert amazon analysis analysis anticipatory application performance