
Open Access

*Open Journal of Web Technologies (OJWT)*
*Volume 5, Issue 1, 2018*

http://www.ronpub.com/ojwt
**ISSN 2199-188X**

# Hijacking DNS Subdomains via Subzone Registration: A Case for Signed Zones

Peter Thomassen, Jan Benninger, Marian Margraf

Freie Universität Berlin, Takustr. 9, 14195 Berlin, Germany
{peter.thomassen, jan.benninger, marian.margraf}@fu-berlin.de

## Abstract

*We investigate how the widespread absence of signatures in DNS (Domain Name System) delegations, in combination with a common misunderstanding with regards to the DNS specification, has led to insecure deployments of authoritative DNS servers which allow for hijacking of subdomains without the domain owner's consent. This, in turn, enables the attacker to perform effective man-in-the-middle attacks on the victim's online services, including TLS (Transport Layer Security) secured connections, without having to touch the victim's DNS zone or leaving a trace on the machine providing the compromised service, such as the web or mail server. Following the practice of responsible disclosure, we present examples of such insecure deployments and suggest remedies for the problem. Most prominently, DNSSEC (Domain Name System Security Extensions) can be used to turn the problem from an integrity breach into a denial-of-service issue, while more thorough user management resolves the issue completely.*

## Type of Paper and Keywords

Regular research paper: *DNS, security, domain, subdomain, zone, man in the middle, TLS certificate, ACME DNS*

## 1 Introduction

Before a connection to a named Internet host (e. g. *www.fu-berlin.de*) can be established, it is necessary to determine the IP address associated with the host name. This lookup is done using the Domain Name System (DNS) which for this reason underlies almost every Internet connection today. The lookup process itself, however, is rather complicated in practice, as the DNS is a decentralized storage system with many independent parties maintaining authoritative information for disjunct subtrees ("zones") within the global name space, and with a myriad of Internet access providers maintaining their own caches. Thus, the correct operation of an authoritative DNS service is a non-trivial task.

Furthermore, while being initially intended and still primarily used for IP lookups, the DNS has been seeing growing use for other domain-related purposes [10]. A common use case is requesting a TLS certificate as is necessary in order to provide trustworthy transport security for a web or mail server running on that particular named host. The certificate is tied to the host name, and in order to establish trust, proof of ownership of that name is required before a certificate will be issued. There are various mechanisms to prove domain ownership one of which is to provide a challenge in the DNS zone containing the domain in question [14]. Other use cases include anti-spam policies, certificate pinning, and key exchange mechanisms [4].

This paper is accepted at the *International Workshop on Web Data Processing & Reasoning (WDPAR 2018)* in conjunction with the 41st German Conference on Artificial Intelligence (KI) in Berlin, Germany. The proceedings of WDPAR@KI 2018 are published in the Open Journal of Web Technologies (OJWT) as special issue.

With the DNS playing such a crucial role in the orderly functioning of the Internet, it is important that the integrity of DNS information be ensured. Failing that, communication partners cannot trust even the most basic expected properties of online communication, such as talking to the intended recipient.

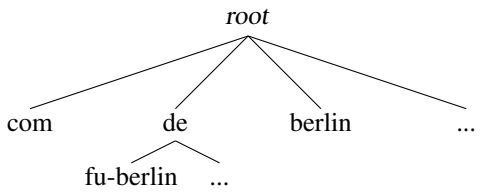


**Figure 1: Tree structure of the DNS**

## 2 Related Work

Having originally been designed as an unauthenticated plaintext protocol, the DNS has offered numerous opportunities for interception and manipulation since its inception [3]. In particular, an attacker who is in control of the transport layer —such as a Wifi operator— can replace DNS response packets at will, leading the client computer to accept spoofed information. *DNS response spoofing* can be achieved even if the attacker cannot intercept and drop the original packet, provided that the fake packet can be injected such that it arrives at the client before the original packet [21].

Clients usually do not query an authoritative DNS server directly. Instead, queries are channeled through "resolver" servers that try to answer the question based on cached information, and only perform full DNS resolution if the answer to the question is unknown or has expired. When performed between the DNS resolver and the authoritative server, the attack described in the previous paragraph can be used to poison resolver caches. Forging the expiration time permits long-term *cache poisoning* [20].

Other DNS abuse scenarios include sending queries with fake IP addresses in the UDP header such that the response, often considerably larger than the question, is directed to the specified third party. Especially when performed in a distributed manner, such large-scale *amplification attacks* quickly lead to resource depletion or bandwidth exhaustion at the receiver, followed by denial of service [1].

Further, DNS integrity can be compromised by inducing irregular behavior in other infrastructure components whose correct operation is taken for granted by DNS resolvers. Only recently, a large DNS service provider suffered from forged BGP route announcements, leading to the redirection of DNS queries to attacker-controlled servers, enabling them to take control of all zones managed by the provider's DNS servers [19, 24].

A similar idea is to take possession of expired domain names which are referenced elsewhere in a security-related context, so that the attacker can act stealthily on the victim's behalf [15, 11].

Various remedies have been proposed to deal with the issues described above, including source port randomization (against cache poisoning), digital signing of DNS information using the Domain Name System Security Extensions (DNSSEC) (against spoofing), BGP signing (against routing attacks), and DNS transport encryption using TLS (against interception) [8, 2, 9, 23, 7].

## 3 Background

### 3.1 DNS Zones vs. Domain Names

The DNS organizes name-related information using a tree structure whose root is managed centrally by so-called DNS root servers (see Fig. 1). By means of DNS delegation records, responsibility for parts of the tree can be transferred to other DNS servers. On the root level, this usually means that responsibility for top-level subtrees such as *.de* is delegated to an independent entity (e. g. DENIC, the *.de* registry).

The delegation cascade starts at the root level and defines the responsibility for each subtree. A server at the end of the delegation chain is said to be an authoritative DNS server.

In DNS terminology, delegated subtrees are called "zones". Each zone corresponds to a distinct data structure on the responsible authoritative DNS server containing DNS records for the names which are part of the zone. For example, the zone *fu-berlin.de* may contain an IP address for this name, but also additional records for other names such as *www.fu-berlin.de*. Note that the latter name is part of the *fu-berlin.de* zone unless a delegation record exists for it. Only by the delegation record's presence, the associated subtree is turned into a zone on its own. The root of a zone is also called the "zone apex".

Valid node (and leaf) names of the name space tree are called domain names. (Validity requirements primarily pertain to length and character set restrictions.) A zone thus may contain information both for the domain name representing the apex and for domain names that have additional prefixes ("subdomains"). Likewise, for a given domain name, there is not necessarily a zone of the same name, as the domain may be a subdomain of the domain at the zone apex.

### 3.2 DNS Resolution

The process of obtaining pieces of information from the DNS is called DNS resolution. It consists of two steps [12]:

1. The client sends a query to a DNS server. The query specifies the domain name (QNAME) and the record type (QTYPE) of interest. Additional flags may be set for extra control over the resolution.

2. The server sends a response to the client. If the desired information was retrieved without error, it is encoded in the response along with various metadata; otherwise, response codes signify the error condition.

Servers can be configured to recursively resolve DNS queries if they find themselves not to be authoritative for the queried QNAME. In this case, the resolution process will follow the delegation chain so that the query will eventually reach an authoritative server. (This assumes that the query has the "recursion desired" flag set.)

RFC 1034 Sec. 4.3.2 specifies the algorithm by which DNS resolution is performed [12]. In particular, it defines how a server determines which data structure to use for assembling the response. Specifically, the server will

> *search the available zones for the zone which is the nearest ancestor to QNAME. If such a zone is found, go to step 3 [...]*

where "step 3" details how to fetch the information associated with the requested QTYPE from the data structure representing the zone that was found.

Note that this part of the algorithm contains a conflict resolution mechanism for the case of ambiguous zone information being present on the server: By referring to the nearest ancestor, the algorithm prescribes that if the server has two (or more) distinct zones configured spanning QNAME —such that one apex domain presents itself as a subdomain of the other—, the information stored in the subdomain zone ("subzone") is decisive for answering the query. Any information that the parent zone may have stored for the same QNAME must not be used in the response.

To better appreciate this algorithmic choice, it is worth illustrating the decision context a bit further. First of all, note that the mechanism is consistent with the notion that the parent zone is no longer authoritative for a subtree that has been delegated someplace else. Still, when both zones are configured on the same server, the nearest ancestor rule applies regardless of the presence of a delegation record in the parent zone. In fact, no check for a delegation record in the parent zone is performed.

Such a check though would actually be spurious: As the server only knows the contents of its own zones, it does not know whether there exists a continuous delegation chain from the root zone to the parent zone in question, and therefore it cannot assume that it is authoritative for the parent zone. As a consequence, it would be ill-founded to honor any delegation (or other) records in the parent zone without starting another recursive resolution process in order to validate the delegation chain. Recursive resolution, however, adding significant latency and requiring an uplink Internet connection, is considered out of scope for an authoritative DNS server.

## 4 Subdomain Hijacking Attack

### 4.1 Scheme

The nearest ancestor rule described in the previous section dictates that if an attacker successfully configures a zone whose apex is a subdomain of another zone that is configured on the same server, the server will exclusively use information from the attacker-controlled subzone when answering queries for QNAMEs covered by that subzone. If the server is also authoritative for the parent zone, this allows the attacker to exert full control over the public DNS subtree corresponding to that subzone.

The following conditions are sufficient for an attacker to take control of a subdomain:

1. The user management policies of the DNS server operator allow the creation of zones even if there is a parent zone defined on the same server but owned by a different user.

2. The server is authoritative for that parent zone.

Under these circumstances, a malicious user can launch a subdomain hijacking attack against zones owned by other users. The result of the attack is equivalent to delegating the hijacked subdomain away from the parent zone, without however having to create a delegation record within the parent zone.

### 4.2 Analysis

The attack brings about several dangerous consequences:

**Traffic redirection:** Typically, zones contain records with IP addresses for prominent subdomains such a *www*. With full control over arbitrary subzones, the attacker can easily redirect traffic to an attacker-controlled machine by creating `A` or `AAAA` records containing IPv4 and IPv6 addresses, respectively. Similarly, a man-in-the-middle attack can be conducted against email traffic by creating an `MX` record.

**Full-nameserver hijacking:** As an extension of the preceding point, a particularly peculiar situation arises if the names of the authoritative nameservers (as listed in the parent zone's delegation records) are subdomains of a zone that is hosted on those same nameservers. This is in fact often the case, a typical examples being the zone *example.com* having `NS` records *ns1.example.com* and *ns2.example.com* in the parent zone *com*.

In this case, an attacker who successfully creates subzones for those names can set `A` and `AAAA` records for the nameserver subdomains and thus — under reasonable assumptions[1]— intercept general nameserver traffic, hijacking zones and replacing their contents at will. This scenario does not only allow for manipulation of subzones, but also of entire zones owned by different users.

Furthermore, this situation allows the attacker to monitor and analyze traffic that is intended to arrive at the hijacked nameservers, enabling the attacker to learn about and potentially compromise other parts of the DNS provider's infrastructure.

**TLS certificate forgery:** By means of the ACME DNS challenge method, domain owners can request TLS certificates and prove domain ownership by placing a challenge inside a `TXT` record at the *_acme-challenge* subdomain [14]. With control over all subdomains, an attacker can not only obtain forged TLS certificates for subdomains like *www* (challenge record at *_acme-challenge.www*), but also for the parent zone apex, as the ownership proof only involves the *_acme-challenge* subdomain and not the apex itself. Having obtained a valid certificate, the attacker can extend the previously described man-in-the-middle attack to encrypted connections.

Both traffic redirection as well as certificate forgery are achieved without touching or leaving any trace on the servers whose services are being compromised (e. g. DNS, web, or mail). Moreover, because the attack does not require the modification of any zone information (but merely the creation of a subzone), the attacker does not need to circumvent any access protections that may be in place for zones configured on the server. As a result, the attack is very unlikely to set off any alarm notification systems if performed properly and carefully (e. g. relaying intercepted traffic on to the intended recipient).

Further harm can be done if a DNS server operator automatically adds delegation records to the parent zone upon creation of a subzone. While a useful feature for users who create subzones under their own zones —especially when using DNSSEC which requires several delegation records in order to establish the cryptographic chain of trust—, it has a pernicious effect when employed with bogus subzones, as the DNSSEC delegation will cryptographically legitimize the delegation so that resolvers are unable to detect the attack.

## 5 Affected DNS Service Providers

To illustrate that the vulnerability is not merely a theoretical issue, we attempted to identify a few DNS service providers affected by the vulnerability. As we indeed found several vulnerable deployments, we notified the affected providers and assisted them with the adoption of appropriate countermeasures (see Sec. 6) before publication of this result. To our knowledge, the operators listed below are no longer vulnerable.

The investigations covered 22 providers and were conducted using two test accounts each, one of which we used to attempt hijacking a subzone belonging to the other. To this end, we tried creating DNS records for the same subdomains in both accounts and, when that worked, checked whether the the provider's servers indeed served the value from the hijacked subzone. As expected (see Sec. 3.2), this was always the case when the fraudulent record creation attempt was not rejected.

In case of success, we also checked if the nameservers were authoritative for their own names, and if so, attempted hijacking a sister subzone of the nameserver domain(s). As to not interfere with any existing configuration, we made sure that these sister names previously had no DNS records set. All hijacked subzones were removed after the tests were finished.

We found the following providers to be vulnerable:

**Hurricane Electric:** This provider runs one of the largest IPv4 and IPv6 transit networks in the world, measured by the number of peerings. Nameservers operated under *ns[1..5].he.net* host various zones, including *he.net* [17]. Other subdomains in this

[1] In the case at hand (with the nameserver names being part of the zone itself), a correctly configured parent zone (e. g. *com*) will have so-called glue records defined. These records, albeit unauthoritative, point resolvers to the nameservers' IP addresses. Without such glue records, resolvers would be stuck in a chicken-and-egg problem when trying to resolve, for instance, *example.com* via *ns1.example.com*. While glue records themselves are not affected by the subzone hijacking attack, resolvers may, as suggested by RFC 2181 Sec. 5.4.1 [6], cross-check glue data with an authoritative nameserver and honor the records found there, even exclusively (replacing the cached information extracted from glue records). As the authoritative data may be part of a hijacked subzone, this especially diligent behavior enables the possibility to hijack entire nameservers. Still, in most configurations, it is likely that at least a small fraction of DNS traffic would still reach the servers specified in the glue records.

zone relate to Hurricane Electric's global network infrastructure. DNSSEC is not used. Zones on these nameservers can be created for free at `https://dns.he.net/`.

We succeeded in hijacking subzones from other users' zones. As these nameservers are authoritative for many high-profile zones which make attractive targets, this is a significant result.

We also found that a restriction was in effect for some zones including *he.net* so that subzone hijacking was not possible. Without this precaution, a full-nameserver hijacking attack could have been performed, the likely effect of which would be a denial of service condition and/or a compromise of significant parts of the Hurricane Electric infrastructure.

**GeoScaling:** This is a small DNS provider whose nameservers *ns[1..5].geoscaling.com* host various zones belonging to sites worldwide [16]. Until recently, the *geoscaling.com* zone itself was also hosted on these nameservers. DNSSEC is not used. Accounts can be set up for free on `http://www.geoscaling.com/dns2/`. (We note that the site does not use transport encryption.)

We succeeded in hijacking subzones from other accounts. Furthermore, no restriction regarding the nameserver's own zone was in place. As the nameservers in question are no longer authoritative for the *geoscaling.com* zone, this is not an issue any longer. However, while the nameservers were still authoritative for that zone, we were able to obtain an illegitimate TLS certificate for *wwww.geoscaling.com*, by hijacking the *_acme-challenge.wwww* subdomain [5]. Other zones managed by GeoScaling remain vulnerable to TLS certificate forgery.

It appears that by the same token, it would have been possible to hijack the nameservers' subdomains, taking extensive control of GeoScaling's DNS operations.

**Vautron AG:** It hosts a significant number of zones in Germany [18]. Their nameservers are, amongst others, *ns{1,2}.ns-serve.net*; these are the ones that were associated with our test accounts. They are also authoritative for their own hostnames. DNSSEC is not used, and account setup is not free.

As a domain reselling provider, Vautron does not allow the creation of arbitrary zones on their nameservers. Instead, zones are automatically created and associated with the current user in the run-up to a domain registration attempt. Interestingly, having registered the domain *example.com* in one user account, it was possible to issue a domain registration order for *www.example.com* in another account. While the domain registration would, of course, be rejected either by the Vautron systems or by the higher-level registry, the corresponding subzone was nevertheless created and associated with the eliciting account. Thus, we succeeded in hijacking subzones belonging to other accounts.

Several weeks later, we attempted hijacking a subzone of *ns-serve.net*, without success. We then also found ourselves unable to perform the attack on other domains. It appears that the provider fixed the issue after more than 12 years, possibly alerted by the ominous domain registration failures.[2]

**Table 1: DNS providers probed for subzone hijacking**

| Name | self-hosted | DNSSEC |
|---|---|---|
| ***vulnerable*** *(yet fixed before publication of this result)* | | |
| Hurricane Electric | yes* | no |
| GeoScaling | yes† | no |
| Vautron‡ | yes | no |
| ***not vulnerable*** | | |
| Amazon Route 53 | GoDaddy | |
| Cloudflare | Internet.bs | |
| ClouDNS | Key-Systems | |
| deSEC | Moniker | |
| DNS Made Easy | Name.com | |
| Dynadot | Namecheap | |
| Dynu | NS1 | |
| easyDNS Technologies | Rackspace | |
| Enom | Safenames | |
| Fabulous.com | | |

* explicit protection for zone spanning nameserver domain

† nameserver zone moved to other server after end of study

‡ fixed vulnerability prior to our notice

As none of the providers found to be vulnerable supported DNSSEC, we cannot make any statements regarding whether the aggravating case of automatic DNSSEC delegation in combination with the vulnerability exists in the wild (see Sec. 4.2). We also probed several other providers and found that they were not vulnerable to the subzone hijacking attack. Table 1 gives an overview of the investigated providers.

[2] In 2005, one of the authors started using the systems that were now investigated. The systems had previously been run by Cronon AG, a subsidiary of Strato AG, and were transferred to Vautron AG in 2010. With nameservers, IP addresses, web interface, and staff remaining the same, it appears that the systems overall are the same as before the changeover.

## 6 COUNTERMEASURES

The vulnerability at hand is not merely a matter of lacking permission checks for access to existing zones. Rather, the issue is rooted in a misconception prompting a conceptually faulty user management policy.

In particular, it is not sufficient for the DNS server operator to impose zone access control restrictions, for the conflict resolution mechanism of RFC 1034 implicates that the policy must also demand verification of whether any zone which is about to be created would fall within the realm of another user's existing zone.

It is yet worth noting that the problem is not simply fixed by adding a check for the existence of parent zones owned by another user: If a DNS server operator naïvely sought to mitigate the hijacking attack in this manner, a malicious user could create a zone whose apex is one of the "public suffixes" from which subzones are routinely delegated to interested parties (such as *de* or *co.uk*), precluding all other users from setting up their legitimate subzones under these domains.

The situation actually presents itself as even more complicated: The list of public suffixes whose immediate subzones are available for interested third parties by means of delegation ("domain registration") is of considerable size ($\sim$ 8,000 entries) [13]. In addition to most top-level domains (such as *de*) and special-case second-level domains (such as *co.uk*), the list also contains even lower-level domains (such as *s3.amazonaws.com*) whose availability for public registration is not immediately obvious by inspection. For this reason, and because it is impossible to reliably identify public suffixes based on their name only, a centrally maintained Public Suffix List (PSL) [13] has been established.

With this in mind, it is clear that user management policy adjustments require a subtler strategy. The proper approach is to augment the parent zone existence check with an exception that permits subzone creation if the immediate parent domain is a public suffix. This approach requires querying the PSL remotely every time a zone is created, or maintaining a regularly updated local copy. Frequent updates are advised as the PSL changes several times per month.

Another solution consists of implementing the parent zone check, but allowing subzone creation if the parent zone contains valid delegation records on its authoritative server, pointing to the server on which the subzone is about to be created. In this case, the delegation could act as an authorization for the nameserver to create (and later answer queries based on) the subzone. However, the idea comes with two disadvantages: First, there is a race condition as a malicious user could create the subzone betwixt the delegation record creation and the legitimate user's attempt to create the subzone. Secondly, delegation records would have to be put in place before the subzone can be populated, potentially causing a downtime for the names in that subzone. Even after the subzone has been fully configured, the downtime duration may be prolonged unexpectedly by caching effects on resolving servers. The approach can be amended by some sort of challenge–response protocol, authorizing a specific user to create the subzone ahead of time; however, such a protocol is not covered by established DNS standards.

Zone signing is very useful to significantly lower the impact of subzone hijacking attacks. In particular, when using DNSSEC to sign DNS records in the parent zone, resolvers will consider invalid any records that stem from the attacker's subzone, for the parent zone does not refer to the subzone's public key as would be required for DNSSEC validation to succeed, and the attacker also cannot forge signatures using the parent zone's (private) key. Thus, the breach of DNS data integrity that is incurred without zone signing is turned into a denial-of-service condition by virtue of using DNSSEC. While not preventing the attack completely, DNSSEC ensures that its consequences are of a much less severe degree. (Note that even without DNSSEC, the attack can be used for denial-of-service purposes, for example by simply leaving the hijacked subzone empty.)

Another way of mitigating the subzone hijacking attack, including the denial-of-service scenario, is to adapt the algorithm in RFC 1034 Sec. 4.3.2 such that the server performs full DNS resolution in order to determine whether it is authoritative for the "zone which is the nearest ancestor to QNAME", and otherwise proceed to the next-nearest ancestor zone. As argued in Sec. 3.2, this would be a major modification to the DNS resolution procedure, and DNS providers trying to adopt it would face various operational challenges.

Lastly, we recall that there exist other attacks — especially those aimed at the transport layer— that open the door to DNS integrity violations (see Sec. 2). We would like to point out that DNSSEC, in addition to controlling the subzone hijacking attack as described before, is also a suitable antidote against all of those attacks, regardless of the reliability of the underlying transport mechanism. It is thus *always* advisable to deploy DNSSEC in order to obtain maximum integrity of DNS data. Formely made counter-arguments, especially regarding the prohibitive size of DNSSEC signatures, are no longer substantiated since the advent of elliptic-curve cryptography [22]. We therefore encourage DNS operators to seriously consider the deployment of DNSSEC.

## 7 Conclusion and Outlook

We investigated a serious vulnerability found in some deployments of authoritative DNS servers that allows for the hijacking of subdomains. By mounting this attack, unauthorized parties can perform full-scale man-in-the-middle attacks on the victim's online services, including TLS-secured connections, without touching the machines providing the compromised services such as web or email. We described that the origins of the vulnerability lie in a misconception about the DNS resolution algorithm defined in RFC 1034, leading to faulty user management concepts. After presenting real-world examples of the issue, we suggested remedies against the problem, including improved user management policies and the deployment of DNSSEC.

The prevalence of the vulnerability in the global DNS market remains unknown, and a systematic evaluation of the DNS service providers operating the most-queried zones in the world would be an interesting subject of further research. During our preliminary survey, we found that due to the inhomogeneity of the ownership landscape with respect to IP and domain owners as well as server operators, and due to further complications such as whitelabeling of nameserver hostnames, the attribution of domain names to DNS operators is rarely straightforward. As a result, a systematic prevalence study is likely to be a rather challenging endeavor.

## Acknowledgements

We thank Tobias Fiebig (TU Delft), Nils Wisiol (Freie Universität Berlin), and the PowerDNS developer team for valuable guidance and discussions on the subject.

## References

[1] M. Anagnostopoulos, G. Kambourakis, P. Kopanos, G. Louloudakis, and S. Gritzalis, "DNS amplification attack revisited," *Computers & Security*, vol. 39, pp. 475–485, 2013.

[2] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose, "Dns security introduction and requirements," Network Working Group, RFC 4033, March 2005. [Online]. Available: https://www.rfc-editor.org/rfc/rfc4033.txt

[3] D. Atkins and R. Austein, "Threat analysis of the domain name system (DNS)," Network Working Group, RFC 3833, August 2004. [Online]. Available: https://www.rfc-editor.org/rfc/rfc3833.txt

[4] R. Barnes, "Use cases and requirements for DNS-based authentication of named entities (DANE)," Internet Engineering Task Force, RFC 6394, October 2011. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6394.txt

[5] COMODO CA. crt.sh — 261191763. Last accessed July 20, 2018. [Online]. Available: https://crt.sh/?id=261191763

[6] R. Elz and R. Bush, "Clarifications to the DNS specification," Network Working Group, RFC 2181, July 1997. [Online]. Available: https://www.rfc-editor.org/rfc/rfc2181.txt

[7] Z. Hu, L. Zhu, J. Heidemann, A. Mankin, D. Wessels, and P. Hoffman, "Specification for DNS over transport layer security (TLS)," Internet Engineering Task Force, RFC 7858, May 2016. [Online]. Available: https://www.rfc-editor.org/rfc/rfc7858.txt

[8] A. Hubert and R. van Mook, "Measures for making dns more resilient against forged answers," Network Working Group, RFC 2452, January 2009. [Online]. Available: https://www.rfc-editor.org/rfc/rfc5452.txt

[9] G. Huston and G. Michaelson, "Validation of route origination using the resource certificate public key infrastructure (PKI) and route origin authorizations (ROAs)," Internet Engineering Task Force, RFC 6483, February 2012. [Online]. Available: https://www.rfc-editor.org/rfc/rfc6483.txt

[10] IANA, "Domain name system (DNS) parameters," IANA, Tech. Rep., January 2018. [Online]. Available: https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml

[11] D. Liu, S. Hao, and H. Wang, "All your dns records point to us: Understanding the security threats of dangling dns records," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, 2016, pp. 1414–1425.

[12] P. Mockapetris, "Domain names - concepts and facilities," Network Working Group, RFC 1034, November 1987. [Online]. Available: https://www.rfc-editor.org/rfc/rfc1034.txt

[13] Mozilla Foundation. Public suffix list. Last accessed July 20, 2018. [Online]. Available: https://publicsuffix.org/list/

[14] R. Barnes, J. Hoffman-Andrews, D. McCarney, J. Kasten, "Automatic certificate management environment (ACME)," IETF, Internet-Draft, July 2018. [Online]. Available: https://tools.ietf.org/html/draft-ietf-acme-acme-13

[15] J. Schlamp, J. Gustafsson, M. Wählisch, T. C. Schmidt, and G. Carle, "The abandoned side of

the internet: Hijacking internet resources when domain names expire," in *International Workshop on Traffic Monitoring and Analysis*. Springer, 2015, pp. 188–201.

[16] SecurityTrails. All domains hosted on NS ns1.geoscaling.com. Last accessed July 20, 2018. [Online]. Available: https://securitytrails.com/list/ns/ns1.geoscaling.com

[17] SecurityTrails. All domains hosted on NS ns1.he.net. Last accessed July 20, 2018. [Online]. Available: https://securitytrails.com/list/ns/ns1.he.net

[18] SecurityTrails. All domains hosted on NS ns1.ns-serve.net. Last accessed July 20, 2018. [Online]. Available: https://securitytrails.com/list/ns/ns1.ns-serve.net

[19] A. Siddiqui. What happened? the amazon route 53 bgp hijack to take over ethereum cryptocurrency wallets. 27 April 2018. [Online]. Available: https://www.internetsociety.org/blog/2018/04/amazons-route-53-bgp-hijack/

[20] S. Son and V. Shmatikov, "The hitchhiker's guide to DNS cache poisoning," in *Proceedings of the International Conference on Security and Privacy in Communication Systems*, 2010, pp. 466–483.

[21] U. Steinhoff, A. Wiesmaier, and R. Araújo, "The state of the art in DNS spoofing," in *the 4th Intl. Conf. Applied Cryptography and Network Security (ACNS)*, 2006.

[22] R. van Rijswijk-Deij, A. Sperotto, and A. Pras, "Making the case for elliptic curves in DNSSEC," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 5, pp. 13–19, 2015.

[23] M. Wählisch, O. Maennel, and T. C. Schmidt, "Towards detecting BGP route hijacking using the RPKI," in *Proceedings of the ACM SIGCOMM 2012 conference on applications, technologies, architectures, and protocols for computer communication*. ACM, 2012, pp. 103–104.

[24] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, and L. Zhang, "Protecting BGP routes to top-level DNS servers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 9, pp. 851–860, 2003.

## AUTHOR BIOGRAPHIES

**Peter Thomassen** received his Master's and Ph.D. degrees in particle physics from Rutgers, The State University of New Jersey. Having collected extensive practical experience in computing and his primary field of interest, IT security, he then joined Professor Margraf's Identity Management research group at Freie Universität Berlin. His work focuses on developing user-friendly real-world IT security solutions.

**Jan Benninger** is currently enrolled at Freie Universität Berlin where he is finishing his Bachelor's degree. His interests include computer security and privacy in the digital age. The latter will also be the topic of his Bachelor's thesis. Jan participated in a various CTF competitions to gain additional hands-on security experience.

**Marian Margraf** received the diploma degree in mathematics and the Ph.D. degree from the University of Kiel, Germany. He is currently Professor at the department of computer science at Freie Universität Berlin. His research interests include cryptographic protocols and cryptanalysis.