TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

NAT64/DNS64 in the Networks with DNSSEC

Dissertation thesis summary

Study programme: P2612 – Electrical Engineering and Informatics
Study branch: 2612V045 – Technical cybernetics

Author: **Ing. Bc. Martin Huněk, M.Eng.**
Supervisor: prof. Ing. Zdeněk Plíva, Ph.D.

Liberec 2021

NAT64/DNS64 in the Networks with DNSSEC

Abstract

The rising number of DNS-over-HTTPS capable resolvers and applications results in the higher use of third-party DNS resolvers by clients. Because of that, the currently most deployed method of the NAT64 prefix detection, the RFC7050[1], fails to detect the NAT64 prefix. As a result, clients using either NAT64/DNS64 or 464XLAT transition mechanisms fail to detect the NAT64 prefix properly, making the IPv4-only resources inaccessible. The aim of this thesis is to develop a new DNS-based detection method that would work with foreign DNS and utilize added security by the DNS security extension, the DNSSEC. The thesis describes current methods of the NAT64 prefix detection, their underlying protocols, and their limitations in their coexistence with other network protocols. The developed method uses the SRV record type to transmit the NAT64 prefix in the global DNS tree. Because the proposed method uses already existing protocols and record types, the method is easily deployable without any modification of the server or the transport infrastructure. Due to the global DNS tree usage, the developed method can utilize the security provided by the DNSSEC and therefore shows better security characteristics than previous methods. This thesis forms the basis for standardization effort in the IETF.

Keywords: IPv6, IPv4aaS, NAT64/DNS64, 464XLAT, DNSSEC, DoH

NAT64/DNS64 v sítích s DNSSEC

Abstrakt

Zvyšující se podíl resolverů a aplikací používající DNS-over-HTTPS vede k vyšímu podílu klientů používajících DNS resolvery třetích stran. Kvůli tomu ovšem selhává nejpoužívanější NAT64 detekční metoda RFC7050[1], což vede u klientů používajících přechodové mechanismy NAT64/DNS64 nebo 464XLAT vede chybné detekci, a tím k nedostupnosti obsahu dostupného pouze po IPv4. Cílem této práce je navrhnout novou detekční metodu postavenou na DNS, která bude pracovat i s resolvery třetích stran, a bude schopná využít zabezpečení DNS dat pomocí technologie DNSSEC. Práce popisuje aktuálně standardizované metody, protokoly na kterých závisí, jejich omezení a interakce s ostatními metodami. Navrhovaná metoda používá SRV záznamy k přenosu informace o použitém NAT64 prefixu v globálním DNS stromu. Protože navržená metoda používá již standardizované protokoly a typy záznamů, je snadno nasaditelná bez nutnosti modifikovat jak DNS server, tak síťovou infrastrukturu. Protože metoda používá k distribuci informace o použitém prefixu globální DNS strom, umožňuje to metodě použít k zabezpečení technologii DNSSEC. To této metodě dává lepší bezpečnostní vlastnosti než jaké vykazují předchozí metody. Tato práce současně vytváří základ pro standardizaci této metody v rámci IETF.

Klíčová slova: IPv6, IPv4aaS, NAT64/DNS64, 464XLAT, DNSSEC, DoH

Contents

Acronyms

API Application Programming Interface. 21, 29

CLAT Customer-side Translator in 464XLAT. 13, 14, 21, 29
CPE Customer Premises Equipment. 13, 20

DHCPv6 Dynamic Host Configuration Protocol version 6. 21, 23, 26
DNS Domain Name System. 11, 13, 15–24, 26, 27, 29
DNS64 Domain Name System 6-to-4. 13–25, 28, 29
DNSSEC Domain Name System Security. 11, 15, 18, 19, 23–26, 28, 29
DNSSL Domain Name System Search List. 23, 29
DoH DNS over HTTPS. 15, 16, 20, 26, 28
DoS Denial of Service. 19
DoT DNS over TLS. 15, 26

FQDN Fully Qualified Domain Name. 24

IETF Internet Engineering Task Force. 20, 22, 29
IP Internet Protocol. 12, 14, 15
IPv4 Internet Protocol version 4. 13–15, 28
IPv4aaS Internet Protocol version 4 as a Service. 13
IPv6 Internet Protocol version 6. 13–15, 23, 27–29
ISP Internet Service Provider. 13, 16

MitM Man in the Middle. 19

NAT Network Address Translation. 12
NAT44 Network Address Translation 4-to-4. 14
NAT64 Network Address Translation 6-to-4. 12–17, 19–28
NSP Network Specific Prefix. 17, 18

PCP Port Control Protocol. 21, 26
PLAT Provider-side Translator in 464XLAT. 13, 14, 28

RA Router Advertisement. 21, 23, 27, 29

TCP Transmission Control Protocol. 25

WKA Well-Known IPv4 Address. 17
WKN Well-Known IPv4-only Name. 17, 19, 20, 28
WKP Well-known Prefix. 17, 18

1 Introduction

One of the first things network administrators would hear, as a network operator trying to deploy IPv6 translation mechanisms, would probably be to never mix it with Domain Name System Security (DNSSEC) validating resolver. This was also my case when I attended an IPv6 course of RIPE NCC in 2017. Back then, I was also considering deployment of NAT64/DNS64 translation mechanism in the network of one Internet Service Provider I am volunteering for, so I started working on the deployment scenario and was investigating related issues.

I eventually discovered a solution by the RFC7050[1], so the issue seemed to be solved for the time being. Even though this RFC7050[1] does not solve all the issues related to discovering a NAT64 prefix, it worked in most common cases and without the need for modifying underlying protocols or significant administrative effort. The only major issue left on a device capable of using Domain Name System (DNS) was an issue of foreign DNS, but back then, it was only caused by user settings, so it was only self-inflicted. A share of users with such custom settings was supposed to be marginal, and the issue caused by this could be solved by the helpdesk of ISP by the simple statement "Use DNS provided by us."

However, in 2017, the standardization of RFC8484[2] (a DNS-over-HTTPS) has begun, and in 2018 it was declared a Proposed Standard. As this method of transporting DNS queries does not have a working detection mechanism, it introduces foreign DNS to every user of such program which uses this transport method. This makes an RFC7050[1] unusable and sends us back to round one of NAT64/DNS64 detection.

In this thesis, I will try to explain the need for a reliable way to securely detect the NAT64/DNS64 translation mechanism as well as a proposed solution for this detection.

Table 1.1: Terminology used in this thesis

Word	Meaning
Client	A network device or network capable software consuming network service.
Node	A network device or network capable software.
Operator	A physical or legal entity providing Internet access to nodes and users.
Server	A network device or network capable software providing network service.
User	A physical or legal entity other than operator

2 Theoretical Background

This chapter is a shortened version of the Theoretical Background chapter of the dissertation. It includes only a brief description of relevant technologies and problems. For a full explanation, please see the full text and its sources.

2.1 IPv6 to IPv4 NAT

Even that Network Address Translation 6-to-4 (NAT64) uses the same principle as other types of Network Address Translation (NAT), but its purpose is different. Its purpose is not to conserve resources or to translate one address pool into another. It is transition mechanism between protocols. It allows communication between two hosts where both are using a different version of Internet Protocol (IP) protocol.

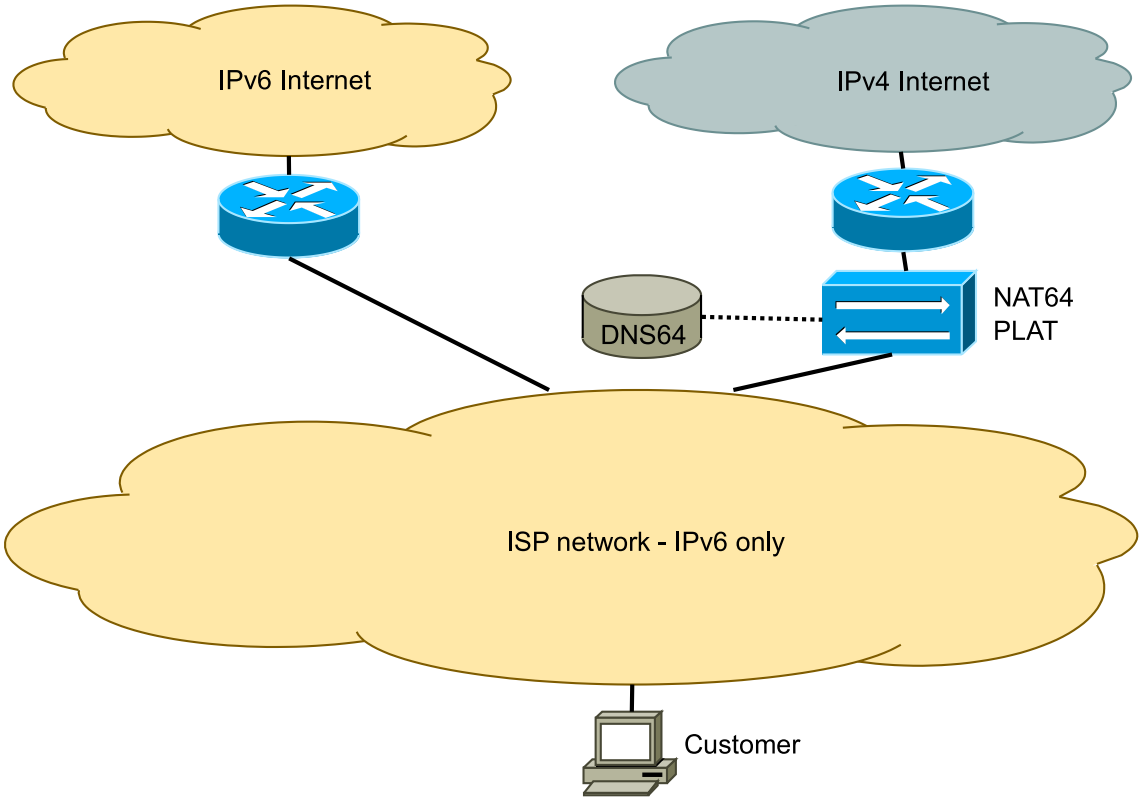

Figure 2.1: NAT64/DNS64 network with IPv6 only customers

The typical network topology for Internet Service Provider (ISP) is shown in figure 2.1. At the top of this figure, there is the Internet in both versions. Below it, there are two ISP's routers which may or may not be a single device. However, two routers represent two router daemon instances which are usually needed. Then on the Internet Protocol version 4 (IPv4) part, two services are needed for translation - the NAT64 and the Domain Name System 6-to-4 (DNS64). The DNS64 service works as a pointer that the requested name is reachable through the NAT64 translation. It is described in detail in section 2.3, and without it, a node would not even try to use the NAT64 service. After the NAT64 box, the whole infrastructure is Internet Protocol version 6 (IPv6)-only, so a customer at the bottom of the picture does not have native IPv4 connectivity.

As long as a customer is using domain names only, instead of hard-coded IPv4 addresses, and using a recursive DNS server provided by ISP, a customer would not notice that IPv4 is not provided natively but only in the mode of so-called Internet Protocol version 4 as a Service (IPv4aaS). However, if any of the prerequisites are not fulfilled, the customer may notice some services not responding as they should. An extension has been developed for these instances that add tunneled IPv4 connections through an IPv6-only network called *464XLAT*.

2.2 464XLAT

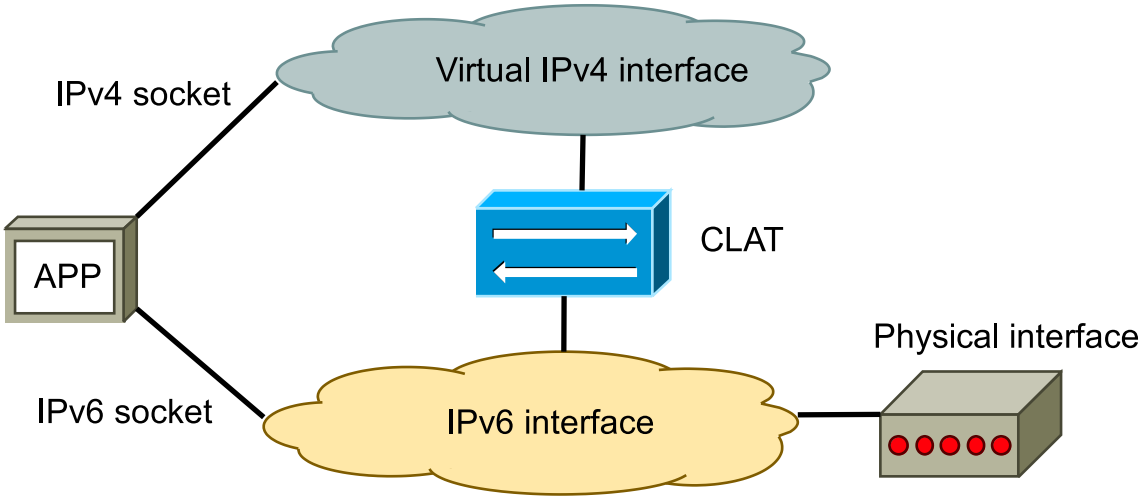

Figure 2.2: Client portion of 464XLAT (CLAT)

In the case of *464XLAT* provider end is the same as in the case of NAT64. The only difference is that there is a different name for the NAT64 box, which is called Provider-side Translator in 464XLAT (PLAT). What *464XLAT* adds is a service called Customer-side Translator in 464XLAT (CLAT). It can be located inside the Customer Premises Equipment (CPE) router or even inside the end-host (for example, an Android phone). The later location is depicted in figure 2.2. However, CLAT

in router would look similar only with physical interfaces on both ends instead of application and IP packets instead of sockets.

CLAT adds stateless "*1:1*" translation in the reverse direction to PLAT. This way, it is capable of creating IPv4 connections over the IPv6-only network. Because an IPv4 source address of the client can be incorporated inside IPv6 address, it is translated into it. The translation could be entirely stateless on the CLAT side. This makes CLAT part transparent for the client/application and the PLAT part behaving in a similar matter as Network Address Translation 4-to-4 (NAT44) from an application perspective. *464XLAT* then resembles the behavior of an IPv4 tunnel inside IPv6 packets from an application point of view. However, it is actually a double translation mechanism rather than an encapsulation one.

With *464XLAT*, applications do have what appears to be a dual-stack internet connection with IPv4 address from the IPv4-to-IPv6 address space. *464XLAT* has got a positive effect on broken applications that require an IPv4 connection to work correctly. This is why an Android supports the CLAT part of the *464XLAT* since the version of 4.3 [3].

2.3 DNS64

The DNS64 is one of two parts constructing the NAT64/DNS64 transition mechanism. Its job is to perform the record synthesis to provide AAAA records for services with only the A records. These synthesized AAAA records point to the NAT64 prefix, effectively routing traffic through the NAT64 box. This way, the node with IPv6-only connectivity would be able to access service with IPv4-only connectivity.

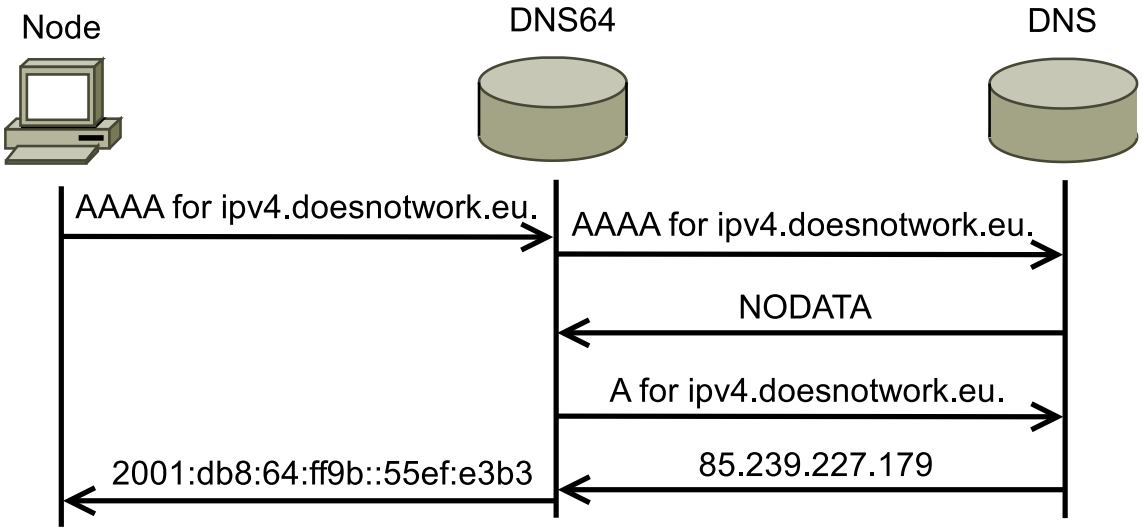

Figure 2.3: DNS64 principle of operation

Figure 2.3 shows steps that DNS64 capable resolver takes in order to perform this service. In the first step, the node asks for an AAAA for a domain name that only runs on IPv4. In this example, the requested domain name is *ipv4.doesnotwork.eu.*

which is part of IPv6 testing web[4] that runs only over the legacy protocol. Node does not know that the requested domain name is IPv4-only.

The DNS64 capable resolver starts its process by trying to resolve the AAAA record first. The DNS64 resolver could be a simple forwarding resolver – then it would just forward the query to its upstream resolver, or it could be a recursive resolver that will try to resolve a node's query by itself. Regardless of the resolution process taken, the DNS64 capable resolver receives a *NODATA* reply. By this reply, the resolver knows that the given domain name exists, but it does not have an AAAA record associated with it. Up until this point, the resolver is doing the same things that it would do without the DNS64 function.

When a node asks for an AAAA record, and there is some record associated with the requested name, there is a reasonable assumption that it would be an A record. So the DNS64 service generates a subsequent query for the A record. The resolver then receives a reply (in this example, *85.239.227.179*). This reply is then embedded into the NAT64 prefix configured at resolver. In this example, the NAT64 prefix is *2001:db8:64:ff9b::/96*, so the resulting address would be *2001:db8:64:ff9b::85.239.227.179*, equal to *2001:db8:64:ff9b::55ef:e3b3*. This synthesized record is then transmitted to the node as a reply to its query. Then the node can connect to the server through the translator.

Suppose a node would not be provided with the DNS64 service, either on a different device or locally, it would not be able to use NAT64 service, so it would not be able to access IPv4-only services. Then the DNS64 had to be considered an integral part of the NAT64/DNS64 transition mechanism whenever domain names are used.

2.4 DNS over HTTPS

Defined by RFC8484[2], the DNS over HTTPS (DoH) tries to solve the different issues of the DNS than the DNSSEC. The DNSSEC provides the authenticity of the received data, while the DoH (and DNS over TLS (DoT)) tries to provide a secure channel between a client and server. While plain-text DNS protocol uses unencrypted and unauthenticated channel through port 53, both DoT and DoH use an encrypted channel with server-side authentication. Both encrypted means of DNS transport uses different encapsulation; the DoT uses a simple TLS/DTLS layer on port 853 while the DoH uses encapsulation in HTTPS protocol. Apart from the difference in the outer protocol, the DoH also requires changes to DNS discovery processes, and the whole concept of using URI instead of IP address also brings new challenges.

At the time of writing this thesis, there was no autoconfiguration method for distributing DoH URI. Distributing just an IP address might not be enough because the path in URI where the DoH API is located is not standardized either. In the RFC8484[2], there is a listed path of */dns-query*, but the DoH operator may use a different path. The DoH client can try to use this path with available DNS servers' IP addresses. However, this may fail just because of different locations.

The lack of a DoH detection method could even be intentional. As one of RFC8484[2] the authors declared to work for Mozilla, and Mozilla as one of the first implementing the DoH in their browser, it is safe to say that implementation in Firefox could be viewed as a reference how this standard was meant to work. However, implementation in Firefox is one of the most controversial.

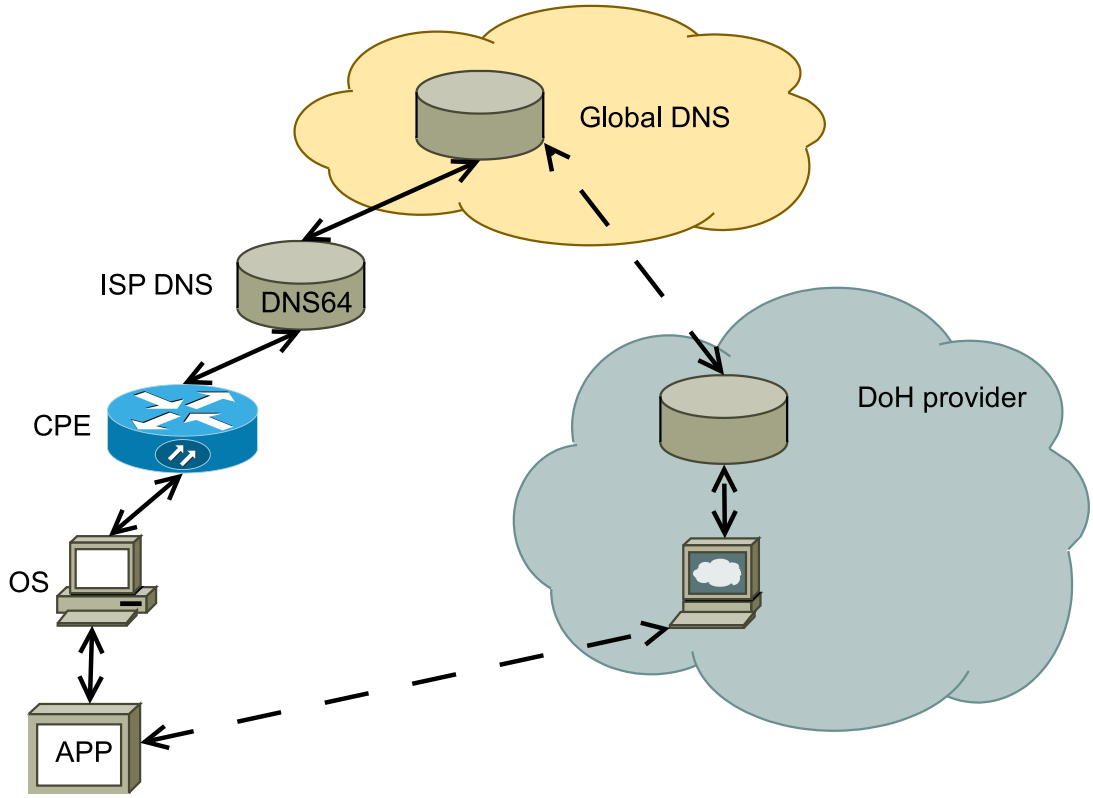

Figure 2.4: Comparison of conventional DNS and DoH query paths

Figure 2.4 shows the usual path for conventional DNS (solid line) and the DoH path implemented in Firefox (dashed line). In the conventional case, the application asks the operating system stub resolver for name resolution. If the stub resolver does not have the record in its cache, it asks forwarding resolver (usually CPE). It can ask recursive resolver provided by ISP that makes the recursive resolution for a client application.

In the case of DoH, at least in Firefox, the application asks directly some DoH resolver it knows. The application is completely ignoring stub resolver in the operating system and its settings and cache. It also does not use cache in forwarding resolver and recursive resolver. The DoH resolver then sends the query to its recursive resolver to provide a client application with an answer.

By using a third-party resolver, the application bypasses the recursive DNS server providing the DNS64 service. This renders the NAT64 service unusable for such application because it has no way to detect the NAT64 prefix used for the translation process.

3 Current Solutions

3.1 RFC7050

The RFC7050[1] is the current standard for getting the Network Specific Prefix (NSP) or the Well-known Prefix (WKP) and also the presence of NAT64/DNS64. For some time, this was sufficient, as a foreign DNS was rare, so it does not have to be solved. In this section, this method would be explained as well as its design features, which lead to current problems with a foreign DNS.

3.1.1 Message Flow

This section shows the message flow as described by the RFC7050[1]. In the first figure 3.1, there is a detection phase of this standard in its most usual deployment scenario. It starts with a query for Well-Known IPv4-only Name (WKN), this is then processed by DNS64 capable server, which would then issue a subsequent query to *arpa.* root server. It would be either AAAA query first then followed by A query. Alternatively, if it is aware of this name's special meaning, it would directly ask for an A record.

When a server receives a response with the Well-Known IPv4 Address (WKA), it will perform address synthesis according to local settings (prefix, prefix length, and encoding scheme) and sends a synthesized reply to a client.

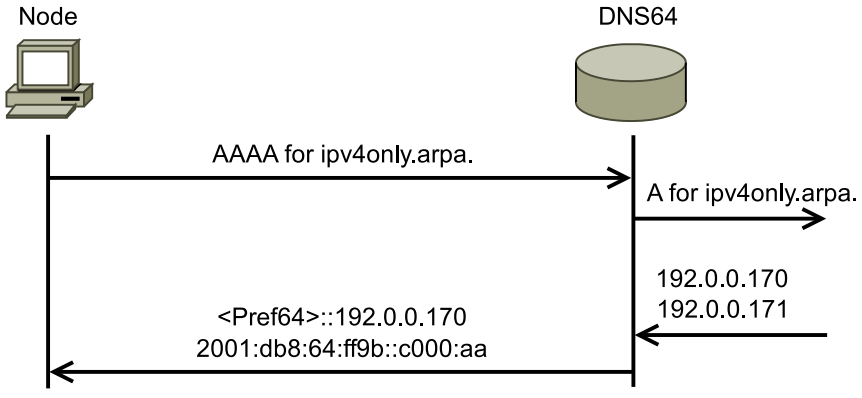

Figure 3.1: Detection of NAT64 prefix according to RFC 7050[1]

With up mentioned steps, the detection phase ends. Non-validating node is allowed to end the detection process here. It will start using those prefixes either for

its own address synthesis or by using a DNS64 server for it. Validating node must continue via process shown in the figure 3.2. There is just one exception when WKP is detected, as it is not possible to validate this prefix. A node should then continue without validation as well.

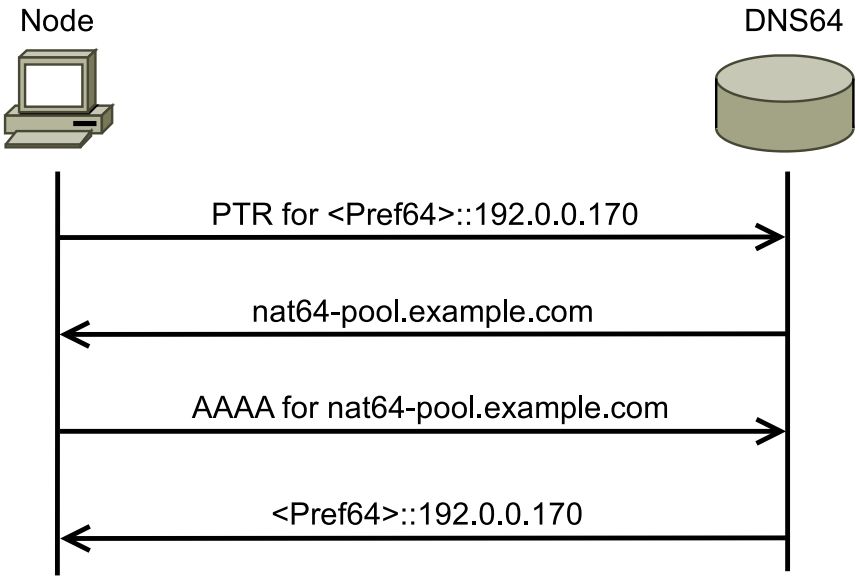

Figure 3.2: Validation of NAT64 prefix according to RFC 7050[1]

The validation phase starts with a client querying for a PTR record for an address that it receives in the detection phase. As a NSP should fall into a zone under control for either network operator or its contractual party, it should be able to provide reverse record back to its forward zone. So a DNS server provides a client with a response of hostname under the provider domain.

When a client receives a reply, it must compare the domain name in the reply with the list of the trusted domains. If this domain is not found in the list, a node must not use such a detected prefix. If it is found, then it would continue with the validation process.

The next step would be asking for an AAAA record of the domain name received in the previous step. Server wound reply with associated prefixes. A client will then check if prefixes received in this step are the same as ones from the detection phase. If an additional record is found, it would be silently discarded.

In the final validation phase, replies received on an AAAA query are validated by the DNSSEC. If all checks out, a prefix then turns into use.

3.1.2 Security Implications

Starting with the implication stated directly in the RFC7050[1], authors realize that it allows the same sort of attacks, as if the DNS64 server was under the control of an attacker. The document further states that replies generated by the DNS64 server could not be validated by the DNSSEC, as it is valid for step one of prefix discovery

proposed by this standard; it is not generally true. Standard is mentioning the same types of attacks as the RFC6147[5]. These include Denial of Service (DoS), Man in the Middle (MitM) and flooding attack, where an attacker is forcing packets on victim flooding its network interface.

Standard is further mentioning securing AAAA and A records with DNSSEC. It states that it is required to secure AAAA resource records, as unsecured records may be forged. It is also suggesting that the *arpa.* zone would also sign *ipv4only.arpa.* A record. This has one side effect as it is also producing an NSEC record (proof of non-existence) for AAAA record of the same WKN. This might be viewed as there is no AAAA for that name (which is right) or as there should be no such record present. However, the standard is using the word "SHOULD" not "MUST" for the requirement of an access network to sign NAT64 resource record.

The requirement of the trusted domain list states that implementations should not ask a user if a discovered domain should be trusted. It is also not providing a way to obtain such a list, but it says that if an implementation does not have a way to obtain such a list dynamically, prefix validation should fail.

Now for the problems not directly stated in the standard. The biggest concern of this standard is the detection phase. In this phase, a client is not capable of validating that it is receiving genuine information. This cannot be ensured because a resource record of *ipv4only.arpa.* is outside of a network operator zone, so it cannot provide a valid signature for this synthesized record.

Standard is trying to solve this insecure record query interception by the requirement of a secure channel between a node and a DNS64 server. This requirement is, however, hardly achievable, as it would require configuration effort, prior established trust relationship between a client and a server (requirement of trust anchor), and this would either require provisioning or encrypted and authenticated transport channel, which is not provided by traditional DNS.

The second way how the standard is trying to mitigate this issue is the trusted domain list. Forming such a list requires detecting locally used domains, node provisioning, or implicit trust to all domains, which would go against standard.

In the validation of detected prefixes, the PTR records are not required to be signed, and a node is not being recommended to validate its signature. This would not be an issue if the forward record is signed and a node is implementing a domain whitelist correctly. Otherwise, it introduces the second breach point to this method's security.

So in order for this method to be secure following must be true:

1. Node must have secure channel for transporting DNS data directly to DNS64 server.

2. Node must be validating.

3. Note must know the domain used for NAT64 resource prior to detection, and this domain must be trusted.

4. Resource record must be signed with a valid signature.

Without the trusted domain list, an attacker could push forged prefix in a detection phase or the first step of the validation phase. A forged prefix could have a valid and signed PTR AAAA pair to pass the validation process.

Without a signed zone, in which resource record is, an attacker can forge any address, even when the trusted domain list is present. With a signed zone attacker is limited by existing records in the zone.

3.1.3 Why RFC7050 Would not Work Now?

The main problem with this method is the detection part as a client is asking for WKN and is expecting to get back a modified record with locally used NAT64 prefix. Then a server, which it is asking, had to be presented with that information somehow. This essentially means that the NAT64 gateway and the DNS64 server, had to be under the control of the same subject – a network operator.

As it has been mentioned before, some technologies like a DoH or operating systems like Android may introduce a foreign DNS, which gets a priority over a network operator's infrastructure. This way, it is not possible to provide such client DNS64 service with this method, as the WKN is resolved only locally, and NAT64 prefixes are unknown to the public DNS infrastructure.

One solution for this limitation discussed at the Internet Engineering Task Force (IETF) is to make query for the WKN resolve locally, not by DoH. This may solve the issue for most of the users as the detection process would work for every client, which is not set up to use foreign DNS in its system stub resolver. From those, which are set up that way, the detection method would still not work, but this might be acceptable for a network operator, as it would not be working because of some settings someone else made to a client.

This concept has been later standardized as RFC8880[6]. This standard adds a requirement for having tight binding between recursive DNS resolver received by autoconfiguration methods with an interface from which it has been received. This is needed only for a query for *ipv4only.arpa.* name as this must be sent to network provider operated resolver. Standard further requires that static configuration not be used - even user-specified resolver cannot be used. This, however, disqualifies any static configuration as this is a user-specified configuration. Regardless of huge architecture changes needed for introduction resolver-interface binding, this standard does not fix other design flaws of RFC7050[1], and it would not work in all cases.

There is also a problem with the trusted domain list. Not all the CPE are custom-made for every network operator. In fact, smaller operators, like local Wi-Fi operators, are using CPEs from an open market, which does not need to be even installed by the operator's technician. This way, CPE is not pre-provisioned, so the trusted domain list could not be provided in advance of connecting it to the operator's network. When not provisioned, the CPE could not establish a secure channel to the provider's infrastructure by pre-loading operator's certificates or pre-shared keys.

3.2 Other Methods not Based on DNS

There are also other detection methods not based on the DNS. These methods are Port Control Protocol (PCP) based RFC7225[7], Dynamic Host Configuration Protocol version 6 (DHCPv6) based RFC8115[8], and Router Advertisement (RA) based RFC8781[9]. All of these methods share some common characteristic that makes their usage in user-space limited.

The first characteristic is connected with the underlying protocols. Both DHCPv6 and RA are typically implemented as a part of the system network stack. However, these protocols are not implemented in the user-space applications, and their access to these protocols could even be blocked by mandatory access control systems. Because of that, applications like web browsers or CLAT clients outside the system network stack would not be able to use those.

The second common characteristic is an expectation of either local DNS64 synthesis or CLAT presence. However, this is not always the case. Some nodes do not have enough resources to do DNS64 synthesis, and some platforms do not integrate CLAT in their network stack.

Those solutions would be a viable option for other platforms that include system support of the CLAT. An example of such a platform would be the Android operating system. It includes CLAT support and it supports RFC8781[9]. However, for now, the RFC8781[9] is not supported in the network equipment.

What is important to note is that DNS-based methods can easily be used for kernel-space implementation. On the other hand, non-DNS-based methods for their implementation require an application to support the client part of the underlying protocol or require system Application Programming Interface (API) change to receive the NAT64 prefix from a system. For this reason, the user-space implementation of the non-DNS-based method is harder on platforms that are not so tightly integrated. Because of that, the non-DNS-based method cannot easily replace the DNS-based one.

4 Proposed Solution

In this section, the thesis is describing a solution of NAT64/DNS64 invented by the author as it is in the process of standardization at IETF in the "v6ops" working group. It has been presented and discussed at IETF 104 meeting, and it is available from Tracker[10] with an intended status of Standard track.

4.1 Design Goals

When thinking about a solution, I had to think about achieving a NAT64 detection with the lowest number of alterations to existing protocols, device implementations and utilizing as much already present information about a network as possible. Another important goal was to maintain network security by not introducing new holes and, if possible, patching existing ones. This is the complete list of design goals:

Goal 1 No new protocol or alteration of an existing one.

Goal 2 Utilize widely supported protocols.

Goal 3 Utilize information already provided by a network.

Goal 4 Must work with foreign DNS.

Goal 5 Must not require DNS64 synthesis on a host.

Goal 6 Must not require prior provisioning.

Goal 7 Must provide secure detection over an insecure channel.

Goal 8 Must be able to run in user-space.

4.2 Node Behavior

In contrast to other methods, this method proposes three stages to the detection method instead of the usual two. The usual steps would be detection and validation. An additional step here is called "Information Gathering" because in this step proposed method does not produce any traffic going out of a node. It is just processing the information presented to a node by network autoconfiguration or other protocols and services.

4.2.1 Information Gathering

In this stage, the main goal of a node is to obtain information about a network to which it has been connected. Information, which is needed, is a list of domain names used by a network operator. This information is later used in the discovery phase. Possible sources of such information are:

1. Domain Name System Search List (DNSSL) from RA,
2. A DHCPv6 options,
3. PTR record for node address,
4. A client hostname.

A PTR would be the preffered source, which node must support. Resolving a PTR record would be the safest method, as validating a node would have the possibility of check the whole chain of trust; from a record of its IPv6 address to root zone, as the root signing key is known to a node. In order for this method to work, an operator must provide DNSSEC signed reverse zone with proper delegation, provide dynamic PTR records to every node, and this zone must have online signing deployed. It may be seen as too many requirements. However, some e-mail servers require valid PTR records from their clients to accept messages from them even on IPv6. This means that at least the requirement of every node having PTR record would be at least in some networks already fulfilled. The signing of a reverse zone is also a straightforward process, and online signing is also doable. As a result, an operator can get a secure and reliable way to detect NAT64/DNS64 over an almost infinite number of devices that do not have to know any configuration connected to them. This is the only proposed method that is producing additional DNS overhead as nodes need to actively ask for PTR record in contrast to other information-gathering methods that either passively gets required data from other configuration protocols or in which node is pre-configured.

4.2.2 Interactions with Other Methods

Even if the SRV method had been standardized, there would still be other methods, and there is no intention to obsolete them. Because of multiple existent methods, it is essential to produce predictable behavior when multiple methods are used in conjunction.

The author of this thesis belive that the network administrator should be the authority that decides which method should be preferred. Even as it is not possible for the network administrator to choose priorities between other methods, it is possible to choose the priority of the SRV method in relation to others. The suggested order of methods is shown in table 4.1 and follows the recommendation published in RFC8781[9].

When a node capable of using the SRV method is also capable of NAT64 detection via other methods, it should run these detection mechanisms and incorporate results

Table 4.1: Recommended priorities of NAT64 prefix detection methods

Standard	Protocol	Priority
RFC8115	DHCPv6	100
RFC7225	PCP	150
RFC8781	RA	200
RFC7050	DNS	250

into those provided by the SRV method. The missing priority field should be filled from table 4.1, and the missing weight field should be set to zero. As a lower priority field value means a higher priority, the fixed list of methods from the most preferred to the least is RFC8115[8], RFC7225[7], RFC8781[9], and RFC7050[1].

The position of the SRV method is then chosen by the network administrator. When the priority of the SRV records is lower than a hundred, the SRV method would be the most preferred one. If the higher priority number is chosen, the SRV method could be set as a backup for other, more preferred methods. The network administrator may even choose to publish multiple SRV records. In such a case, it is recommended that more client-specific records would have a lower priority number, and more general records would be published with a higher priority number. Nevertheless, it is recommended that the priority field of every SRV record published by the network operator should be less than 250 as the RFC7050[1] should be treated as a backup solution only as it is without proper provisioning considered to be the least secure solution.

4.2.3 Detection of a Local Domain via PTR

Client asks for its Fully Qualified Domain Name (FQDN) via PTR record in DNS. It receives its FQDN, records it, and makes a list of records starting with its FQDN and then omitting the first part ending with a dot. It will then record a result as well and continues this process until it reaches the root zone. This list is then subsequently used in the discovery phase.

Figure 4.1 depicts how local domain could be obtained from DNS and how the answer would look like when using synthetic records generated *synthrecord* module of the *knot* resolver.

4.2.4 Discovery Phase

Client asks for every domain a SRV record by prepending "*_nat64._ipv6.*" for NAT64 and optionally "*_dns64._<proto>.*" for DNS64. Answers to those queries are then validated by DNSSEC and grouped into lists of detected NAT64 prefixes and DNS64 servers. If answers to SRV queries did not include AAAA records in additional section, then subsequent queries are made. Priority and weight of SRV record must be recorded and associated with detected prefix and/or server as they are used for selection process.

Figure 4.2 shows the sequence of messages needed in order to discover a NAT64

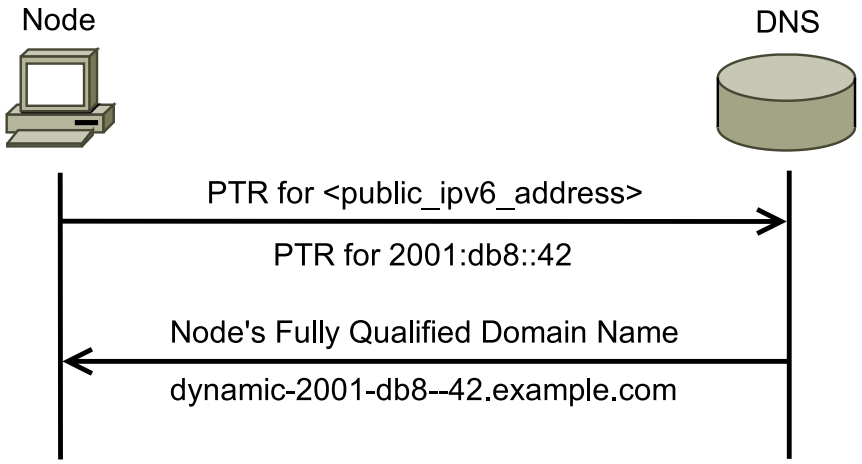

Figure 4.1: PTR query for node's FQDN

```
$ORIGIN example.net

% NAT64 records
_nat64._ipv6     IN SRV  5 10 9632 nat64-pool.example.net
nat64-pool       IN AAAA 2001:db8:64:ff9b::c000:aa
nat64-pool       IN A    192.0.2.64

% DNS64 records - stating priorities
_dns64._tcp      IN SRV  5  10 53  dns64.example.net
_dns64._udp      IN SRV  10 10 53  dns64.example.net
_dns64._tcp      IN SRV  20 10 443 dns64.example.net
dns64            IN AAAA 2001:db8::53
```

Listing 4.1: Example of NAT64/DNS64 records in operator zone

prefix via SRV record. This example shows a situation where the AAAA record was not included in an additional section of the server reply. Figure 4.3 then shows the discovery of DNS64 service on Transmission Control Protocol (TCP) protocol. In this example, a server included AAAA record - shown in brackets. If it would not, the subsequent query like in figure 4.2 would need to be made. Both of these examples represent data shown in listing 4.1.

4.3 Comparison with Other Solutions

For extensive comparison, please see the full text. In the thesis, there is a full review of all currently standardized methods, both according to RFC7051[11] and according to the design goals of the proposed method.

The main points of the comparison with the currently most used method RFC7050[1] are: proposed method works with third-party resolvers, it utilizes DNSSEC, specifies NAT64 priorities, and allows load sharing - the RFC7050[1] does not. Furthermore, the RFC7050[1] requires a secure channel and trusted domain list while the proposed method does not.

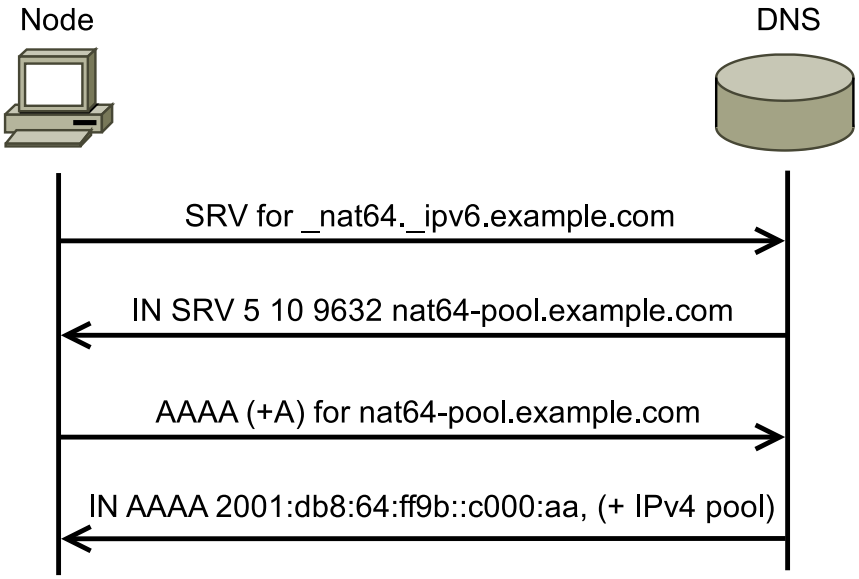

Figure 4.2: SRV query for NAT64 prefix

4.4 Security Considerations

The security of the SRV method relies heavily on the security provided by the DNSSEC. The network operator utilizing this method has to secure at least one of its forward domains with the DNSSEC, and such domain has to be announced to its clients, and the reverse domain for addresses used by clients has to be secured too. Only if both forward and reverse domains are fully secured and securely delegated can a client be sure that the replies have not been modified.

It is important to note that only DNSSEC validating clients are fully protected against DNS record modification. The not DNSSEC validating clients are not protected entirely and should ideally utilize a protocol that provides a secure channel between a client and its closest validating resolver. Minimization of the distance between a client and validating resolver would also be advised.

The author's recommendation would be to deploy at least validating caching resolver inside the local area network, reducing DNS attack surface outside the local network. This can be further aided by the usage of DoT against such resolver, reducing internal threats. The author would not recommend using a third-party DoH resolver. Even that the security provided by such a solution against the NAT64 detection method would be sufficient, the privacy implication of using a third-party provider may not be justified.

Overall, the SRV method has at least the same level of security as the previous methods. It allows the detection from DNS like RFC7050[1], but unlike RFC7050[1], it does not depend on provisioning, secure channel, and particular resolver.

Furthermore, the SRV method does not add dependencies on other protocols that might not be already present in the network, like PCP in the case of RFC7225[7] or the DHCPv6 in the case of RFC8115[8]. When a new protocol that was not needed is added to the network, the attack surface against such a network gets bigger. The

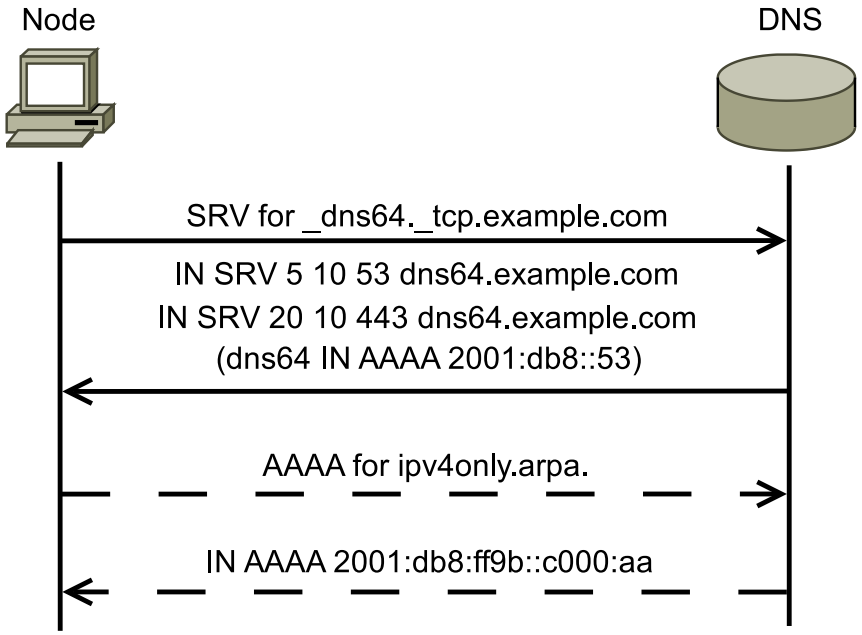

Figure 4.3: SRV query for DNS64 service

SRV method utilizes only the DNS that would be present in most networks.

Lastly, the SRV method allows a client to detect manipulation with the NAT64 prefix that is not provided by any other method. When using the PTR record to detect local domain, this verification method would fail only if the attacker could insert a fake RA and control the appropriate IPv6 zone to distribute arbitrary prefixes to a client. Other methods do not provide such verification.

4.5 Deployment of the SRV Method

When all the prerequisites of the SRV method are fulfilled (dynamic records and online signing), the deployment of this method is reduced into the simple step of adding appropriate SRV records into the operator's forward zone. An example of these records is shown in listing 4.1. By this simple step, the deployment of this method is finished on the network operator's end.

5 Conclusion

Due to the slower than suspected adoption of the IPv6, some services are still reachable only over the IPv4, while services reachable only over the IPv6 are quite seldom. Meaning, the IPv6-only connection without any transition mechanism would not be viewed as a complete service, while the IPv4-only service would look like to be unrestricted when viewed by a customer. Therefore, providing access to the IPv4 Internet is, and for some time, will be a necessity for every Internet provider.

One of the solutions would be to run both protocols in parallel, in so-called Dual-Stack mode, and many operators would start with this architecture. Then the operator realizes that it is doing everything twice. It has to configure addresses for both protocols, set up firewalls, traffic shaping, run dynamic routing for both protocols. Also, when running both protocols, the network operator has to secure both protocols as the network can be attacked on both protocols. By the L3 view, the operator runs two separate networks, meaning twice as many administrative tasks, security threats, and higher operational costs. There comes a time when the operator starts to think of shutting down the legacy network.

In order for the operator to shut down the IPv4 while keeping the IPv4 Internet reachable for its clients, a transition mechanism must be used. Today's two most used transition mechanisms are the NAT64/DNS64 and the 464XLAT. Both of those algorithms share a common component on the operator end - the NAT64, called PLAT in the 464XLAT. For those transition mechanisms to work, the NAT64 prefix has to be reliably and securely detected.

For years the reliable detection was provided by the RFC7050[1] method. The method has been designed to be reasonably secure when strict prerequisites have been met. This method requires a trusted domain list on clients and a secure channel between a client and resolver. However, there are implementations using this method that do not follow those requirements making this method a security threat. This method also requires the DNSSEC to be switched off on the validating client as this detection method would cause DNSSEC validation to fail. Furthermore, after the standardization of DoH, clients started to use third-party resolvers, rendering RFC7050[1] unusable.

The method was later patched by the RFC8880[6], specifying the resolvers that should be used to resolve WKN and adding yet more prerequisites. Theoretically, this fixed a problem with the DoH resolver. However, it also added even more prerequisites to those that have not been honored. Because of its prerequisites, the RFC7050[1] is easy to implement incorrectly and incredibly hard to implement correctly. It served well, but it was designed for circumstances that are no longer

true, and therefore it has to be replaced.

In the pursuit of this replacement, three other methods were standardized. All of those methods use different protocols than the DNS to avoid design limitations of the RFC7050[1], and all of them need to modify protocols that they are using for transport, making support of the new feature required on all fronts (client, server, and transport). Furthermore, two of those methods require a non-essential protocol to run.

The SRV method presented in this thesis is different. It has been designed not to require any change to any protocol it uses, so the support of the new feature is needed only on the client that requires it. It has been designed to use only the protocol present in most networks (the DNS). It has been designed to work with foreign DNS and with DNSSEC while not requiring any new functionality to be moved to the client (DNS64 or CLAT), like in the case of other methods.

The SRV method has also been designed in mind of one not-honored prerequisite of the RFC7050[1], the prior provisioning. Therefore, the SRV method does not require that as well as the other not honored prerequisite, the secure channel requirement. All of that is in the form of easily accessible information presented to an application by the DNS protocol they are used to run without needing a new platform-specific API. This way, any application can utilize this method in user-space, without administrative privileges, without the need to implement a new protocol or use a new API provided by the operating system or its network stack. As a bonus, this method can be enabled in the whole operator's network by a configuration change at a single point - the master authoritative DNS.

The contribution of this thesis is the replacement of the method that may not work in current conditions with a new method that is better suited for the current Internet, more secure while not sacrificing ease of use. This new method fulfills the design goals presented in this thesis. It provides at least the same level of security of detection process as previous methods, but in contrast to them, it provides additional verification of received data by using the DNSSEC - the same extension of the DNS the original detection method, the RFC7050[1], struggled to cooperate with.

Although this new method does not need any changes to protocols to be used in any network right away, the standardization of this method was attempted inside the IETF. After the first attempt for the standardization, the method was improved into the version presented in this thesis. The first version utilized the DNSSL option of the RA packet. However, a different approach was taken, as this option could not be validated, and it is not transitive through the routers. The current version instead uses the PTR record for the client's IPv6 address. While this adds the requirement for the operator to provide dynamic records with an online signing, this method of the local domain detection can be verified by the DNSSEC, eliminating the loophole in the detection process, and it is transitive through the network regardless of the number of routers in the path.

The future work on this topic will focus on finishing the standardization process of this method and providing an actual implementation of this method in the CLAT daemon.

Bibliography

1. SAVOLAINEN, Teemu; KORHONEN, Jouni; WING, Dan. *Discovery of the IPv6 Prefix Used for IPv6 Address Synthesis* [RFC 7050]. RFC Editor, 2013. Request for Comments, no. 7050. ISSN 2070-1721. Available from DOI: 10.17487/RFC7050.
2. HOFFMAN, Paul E.; MCMANUS, Patrick. *DNS Queries over HTTPS (DoH)* [RFC 8484]. RFC Editor, 2018. Request for Comments, no. 8484. ISSN 2070-1721. Available from DOI: 10.17487/RFC8484.
3. ŽORŽ, Jan. *Skype On Android Works Over IPv6 On Mobile Networks Using 464XLAT* [online]. Reston, VA, USA: Internet Society, 2013 [visited on 2021-09-11]. Available from: https://www.internetsociety.org/blog/2013/11/skype-on-android-works-over-ipv6-on-mobile-networks-using-464xlat/.
4. KRČMÁŘ, Petr; CALETKA, Ondřej. *DoesNotWork.eu* [online]. Czech Republic: Krčmář, Caletka, 2016 [visited on 2021-09-26]. Available from: https://www.doesnotwork.eu/.
5. MATTHEWS, Philip; SULLIVAN, Andrew; BEIJNUM, Iljitsch van; BAGNULO, Marcelo. *DNS64: DNS Extensions for Network Address Translation from IPv6 Clients to IPv4 Servers* [RFC 6147]. RFC Editor, 2011. Request for Comments, no. 6147. ISSN 2070-1721. Available from DOI: 10.17487/RFC6147.
6. CHESHIRE, Stuart; SCHINAZI, David. *Special Use Domain Name 'ipv4only.arpa'* [RFC 8880]. RFC Editor, 2020. Request for Comments, no. 8880. Available from DOI: 10.17487/RFC8880.
7. BOUCADAIR, Mohamed. *Discovering NAT64 IPv6 Prefixes Using the Port Control Protocol (PCP)* [RFC 7225]. RFC Editor, 2014. Request for Comments, no. 7225. ISSN 2070-1721. Available from DOI: 10.17487/RFC7225.
8. BOUCADAIR, Mohamed; QIN, Jacni; TSOU, Tina; DENG, Xiaohong. *DHCPv6 Option for IPv4-Embedded Multicast and Unicast IPv6 Prefixes* [RFC 8115]. RFC Editor, 2017. Request for Comments, no. 8115. ISSN 2070-1721. Available from DOI: 10.17487/RFC8115.
9. COLITTI, Lorenzo; LINKOVA, Jen. *Discovering PREF64 in Router Advertisements* [RFC 8781]. RFC Editor, 2020. Request for Comments, no. 8781. Available from DOI: 10.17487/RFC8781.

10. HUNEK, Martin. *NAT64/DNS64 detection via SRV Records*. Internet Engineering Task Force, 2021. Available also from: https://datatracker.ietf.org/doc/html/draft-hunek-v6ops-nat64-srv-00. Internet-Draft. Internet Engineering Task Force. Work in Progress.

11. KORHONEN, Jouni; SAVOLAINEN, Teemu. *Analysis of Solution Proposals for Hosts to Learn NAT64 Prefix* [RFC 7051]. RFC Editor, 2013. Request for Comments, no. 7051. ISSN 2070-1721. Available from DOI: 10.17487/RFC7051.

12. ALSADEH, Ahmad; MEINEL, Christoph. Secure Neighbor Discovery: Review, Challenges, Perspectives, and Recommendations. *Security & Privacy, IEEE*. 2012, vol. 10, pp. 26–34. Available from DOI: 10.1109/MSP.2012.27.

13. *DNS over TLS support in Android P Developer Preview* [online]. Mountain View, CA, USA: Google [visited on 2021-11-28]. Available from: https://security.googleblog.com/2018/04/dns-over-tls-support-in-android-p.html.

14. *DNS over HTTPS (aka DoH)* [online]. Mountain View, CA, USA: The Chromium Projects, 2021 [visited on 2021-09-25]. Available from: https://www.chromium.org/developers/dns-over-https.

15. COLITTI, Lorenzo. *Listen for pref64 RA attributes in IpClientLinkObserver*. [online]. Mountain View, CA, USA: Google, 2020 [visited on 2021-10-06]. Available from: https://cs.android.com/android/_/android/platform/packages/modules/NetworkStack/+/70d7ffa59f1f4ac242d84091431084660251 02c7.

16. *DNSSEC* [online]. Praha: CZ.NIC, 2021 [visited on 2021-09-22]. Available from: https://stats.adam.nic.cz/dashboard/en/DNSSEC.html.

17. *IP Address Sharing in Large Scale Networks: DNS64/NAT64 (BIG-IP v10: LTM)* [online]. Seattle, WA, USA: F5, 2016 [visited on 2021-11-28]. Available from: https://www.f5.com/services/resources/deployment-guides/ip-address-sharing-in-large-scale-networks-dns64na.

18. ARIYAPPERUMA, Suranjith; MITCHELL, Chris J. Security vulnerabilities in DNS and DNSSEC. In: *The Second International Conference on Availability, Reliability and Security (ARES'07)*. 2007, pp. 335–342. Available from DOI: 10.1109/ARES.2007.139.

19. SURÝ, Ondřej. *DNS flag day 2020* [online]. Czech Republic, 2020 [visited on 2021-09-15]. Available from: https://dnsflagday.net/2020/.

20. *Jool SIIT & NAT64: Home* [online]. Col. Altavista Monterrey, Mexico: NIC MÉXICO, 2021 [visited on 2021-08-29]. Available from: https://www.jool.mx/en/index.html.

21. *Port Control Protocol: Adaptive Services Interfaces User Guide for Routing Devices* [online]. Sunnyvale, CA, USA: Juniper Networks, 2021 [visited on 2021-10-01]. Available from: https://www.juniper.net/documentation/us/en/software/junos/interfaces-adaptive-services/topics/topic-map/port-control-protocol.html.

22. *The Domain Name Industry Brief* [online]. Reston, VA, USA: Verisign, 2021 [visited on 2021-09-12]. Available from: https://www.verisign.com/en_US/domain-names/dnib/index.xhtml.

23. *DNSSEC Scoreboard: .com and .net Domain Names with DS Records* [online]. Reston, VA, USA: Verisign, 2021 [visited on 2021-09-23]. Available from: https://www.verisign.com/en_US/company-information/verisign-labs/internet-security-tools/dnssec-scoreboard/index.xhtml.

24. *DNS Clients* [online]. Redmond, Washington, USA: Microsoft Corporation, 2016 [visited on 2021-09-21]. Available from: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn593685(v=ws.11).

25. WIKIPEDIA CONTRIBUTORS. *OSI model* [online]. Wikipedia, The Free Encyclopedia, 2019 [visited on 2019-05-27]. Available from: https://en.wikipedia.org/w/index.php?title=OSI_model&oldid=898908861.

26. POSTEL, J. *Assigned numbers* [RFC 790]. RFC Editor, 1981. Request for Comments, no. 790. ISSN 2070-1721. Available from DOI: 10.17487/RFC0790.

27. POSTEL, Jon (ed.). *Internet Protocol* [RFC 791]. RFC Editor, 1981. Request for Comments, no. 791. ISSN 2070-1721. Available from DOI: 10.17487/RFC0791.

28. POSTEL, Jon. *NCP/TCP transition plan* [RFC 801]. RFC Editor, 1981. Request for Comments, no. 801. ISSN 2070-1721. Available from DOI: 10.17487/RFC0801.

29. MOCKAPETRIS, P. *Domain names: Concepts and facilities* [RFC 882]. RFC Editor, 1983. Request for Comments, no. 882. ISSN 2070-1721. Available from DOI: 10.17487/RFC0882.

30. MOCKAPETRIS, P. *Domain names: Implementation specification* [RFC 883]. RFC Editor, 1983. Request for Comments, no. 883. ISSN 2070-1721. Available from DOI: 10.17487/RFC0883.

31. *DoD Internet host table specification* [RFC 952]. RFC Editor, 1985. Request for Comments, no. 952. Available from DOI: 10.17487/RFC0952.

32. *Domain names - implementation and specification* [RFC 1035]. RFC Editor, 1987. Request for Comments, no. 1035. ISSN 2070-1721. Available from DOI: 10.17487/RFC1035.

33. YU, Jessica; LI, Tony; VARADHAN, Kannan; FULLER, Vince. *Supernetting: an Address Assignment and Aggregation Strategy* [RFC 1338]. RFC Editor, 1992. Request for Comments, no. 1338. Available from DOI: 10.17487/RFC1338.

34. FULLER, Vince; LI, Tony; VARADHAN, Kannan; YU, Jessica. *Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy* [RFC 1519]. RFC Editor, 1993. Request for Comments, no. 1519. Available from DOI: 10.17487/RFC1519.

35. DROMS, Ralph. *Dynamic Host Configuration Protocol* [RFC 1531]. RFC Editor, 1993. Request for Comments, no. 1531. ISSN 2070-1721. Available from DOI: 10.17487/RFC1531.

36. EGEVANG, Kjeld Borch; FRANCIS, Paul. *The IP Network Address Translator (NAT)* [RFC 1631]. RFC Editor, 1994. Request for Comments, no. 1631. Available from DOI: 10.17487/RFC1631.

37. MOSKOWITZ, Robert; KARRENBERG, Daniel; REKHTER, Yakov; LEAR, Eliot; GROOT, Geert Jan de. *Address Allocation for Private Internets* [RFC 1918]. RFC Editor, 1996. Request for Comments, no. 1918. Available from DOI: 10.17487/RFC1918.

38. CARREL, David; EVARTS, Jeff; LIDL, Kurt; MAMAKOS, Louis A.; SIMONE, Dan; WHEELER, Ross. *A Method for Transmitting PPP Over Ethernet (PPPoE)* [RFC 2516]. RFC Editor, 1999. Request for Comments, no. 2516. ISSN 2070-1721. Available from DOI: 10.17487/RFC2516.

39. VIXIE, Paul A. *Extension Mechanisms for DNS (EDNS0)* [RFC 2671]. RFC Editor, 1999. Request for Comments, no. 2671. ISSN 2070-1721. Available from DOI: 10.17487/RFC2671.

40. HAIN, Tony L. *Architectural Implications of NAT* [RFC 2993]. RFC Editor, 2000. Request for Comments, no. 2993. Available from DOI: 10.17487/RFC2993.

41. EGEVANG, Kjeld Borch; SRISURESH, Pyda. *Traditional IP Network Address Translator (Traditional NAT)* [RFC 3022]. RFC Editor, 2001. Request for Comments, no. 3022. ISSN 2070-1721. Available from DOI: 10.17487/RFC3022.

42. CONRAD, David R. *Indicating Resolver Support of DNSSEC* [RFC 3225]. RFC Editor, 2001. Request for Comments, no. 3225. ISSN 2070-1721. Available from DOI: 10.17487/RFC3225.

43. BELLOVIN, Steven. *The Security Flag in the IPv4 Header* [RFC 3514]. RFC Editor, 2003. Request for Comments, no. 3514. ISSN 2070-1721. Available from DOI: 10.17487/RFC3514.

44. DROMS, Ralph. *DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* [RFC 3646]. RFC Editor, 2003. Request for Comments, no. 3646. ISSN 2070-1721. Available from DOI: 10.17487/RFC3646.

45. VOLLBRECHT, John; CARLSON, James D.; BLUNK, Larry; PH.D., Dr. Bernard D. Aboba; LEVKOWETZ, Henrik. *Extensible Authentication Protocol (EAP)* [RFC 3748]. RFC Editor, 2004. Request for Comments, no. 3748. ISSN 2070-1721. Available from DOI: 10.17487/RFC3748.

46. KEMPF, James; ARKKO, Jari; ZILL, Brian; NIKANDER, Pekka. *SEcure Neighbor Discovery (SEND)* [RFC 3971]. RFC Editor, 2005. Request for Comments, no. 3971. ISSN 2070-1721. Available from DOI: 10.17487/RFC3971.

47. ROSE, Scott; LARSON, Matt; MASSEY, Dan; AUSTEIN, Rob; ARENDS, Roy. *DNS Security Introduction and Requirements* [RFC 4033]. RFC Editor, 2005. Request for Comments, no. 4033. ISSN 2070-1721. Available from DOI: 10.17487/RFC4033.

48. ROSE, Scott; LARSON, Matt; MASSEY, Dan; AUSTEIN, Rob; ARENDS, Roy. *Resource Records for the DNS Security Extensions* [RFC 4034]. RFC Editor, 2005. Request for Comments, no. 4034. ISSN 2070-1721. Available from DOI: 10.17487/RFC4034.

49. ROSE, Scott; LARSON, Matt; MASSEY, Dan; AUSTEIN, Rob; ARENDS, Roy. *Protocol Modifications for the DNS Security Extensions* [RFC 4035]. RFC Editor, 2005. Request for Comments, no. 4035. ISSN 2070-1721. Available from DOI: 10.17487/RFC4035.

50. GILLIGAN, Robert E.; NORDMARK, Erik. *Basic Transition Mechanisms for IPv6 Hosts and Routers* [RFC 4213]. RFC Editor, 2005. Request for Comments, no. 4213. Available from DOI: 10.17487/RFC4213.

51. DEERING, Dr. Steve E.; HINDEN, Bob. *IP Version 6 Addressing Architecture* [RFC 4291]. RFC Editor, 2006. Request for Comments, no. 4291. Available from DOI: 10.17487/RFC4291.

52. HUITEMA, Christian. *Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)* [RFC 4380]. RFC Editor, 2006. Request for Comments, no. 4380. Available from DOI: 10.17487/RFC4380.

53. VOLZ, Bernie. *The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Client Fully Qualified Domain Name (FQDN) Option* [RFC 4704]. RFC Editor, 2006. Request for Comments, no. 4704. ISSN 2070-1721. Available from DOI: 10.17487/RFC4704.

54. DAIGLE, Leslie. *Domain-Based Application Service Location Using URIs and the Dynamic Delegation Discovery Service (DDDS)* [RFC 4848]. RFC Editor, 2007. Request for Comments, no. 4848. ISSN 2070-1721. Available from DOI: 10.17487/RFC4848.

55. ZHANG, Lixia; THALER, Dave; LEBOVITZ, Gregory M. *IAB Thoughts on IPv6 Network Address Translation* [RFC 5902]. RFC Editor, 2010. Request for Comments, no. 5902. Available from DOI: 10.17487/RFC5902.

56. IAB; KOCH, Peter; FÄLTSTRÖM, Patrik; AUSTEIN, Rob. *Design Choices When Expanding the DNS* [RFC 5507]. RFC Editor, 2009. Request for Comments, no. 5507. ISSN 2070-1721. Available from DOI: 10.17487/RFC5507.

57. TOWNSLEY, Mark; TRØAN, Ole. *IPv6 Rapid Deployment on IPv4 Infrastructures (6rd) – Protocol Specification* [RFC 5969]. RFC Editor, 2010. Request for Comments, no. 5969. Available from DOI: 10.17487/RFC5969.

58. THOMSON, Martin; WINTERBOTTOM, James. *Discovering the Local Location Information Server (LIS)* [RFC 5986]. RFC Editor, 2010. Request for Comments, no. 5986. ISSN 2070-1721. Available from DOI: 10.17487/RFC5986.

59. LI, Xing; BOUCADAIR, Mohamed; HUITEMA, Christian; BAGNULO, Marcelo; BAO, Congxiao. *IPv6 Addressing of IPv4/IPv6 Translators* [RFC 6052]. RFC Editor, 2010. Request for Comments, no. 6052. ISSN 2070-1721. Available from DOI: 10.17487/RFC6052.

60. MATTHEWS, Philip; BEIJNUM, Iljitsch van; BAGNULO, Marcelo. *Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers* [RFC 6146]. RFC Editor, 2011. Request for Comments, no. 6146. ISSN 2070-1721. Available from DOI: 10.17487/RFC6146.

61. WEIL, Jason; KUARSINGH, Victor; DONLEY, Chris; LILJENSTOLPE, Christopher; AZINGER, Marla. *IANA-Reserved IPv4 Prefix for Shared Address Space* [RFC 6598]. RFC Editor, 2012. Request for Comments, no. 6598. Available from DOI: 10.17487/RFC6598.

62. SAVOLAINEN, Teemu; KATO, Jun-ya; LEMON, Ted. *Improved Recursive DNS Server Selection for Multi-Interfaced Nodes* [RFC 6731]. RFC Editor, 2012. Request for Comments, no. 6731. ISSN 2070-1721. Available from DOI: 10.17487/RFC6731.

63. TOUCH, Dr. Joseph D. *Updated Specification of the IPv4 ID Field* [RFC 6864]. RFC Editor, 2013. Request for Comments, no. 6864. ISSN 2070-1721. Available from DOI: 10.17487/RFC6864.

64. MAWATARI, Masataka; KAWASHIMA, Masanobu; BYRNE, Cameron. *464XLAT: Combination of Stateful and Stateless Translation* [RFC 6877]. RFC Editor, 2013. Request for Comments, no. 6877. ISSN 2070-1721. Available from DOI: 10.17487/RFC6877.

65. WING, Dan; CHESHIRE, Stuart; BOUCADAIR, Mohamed; PENNO, Reinaldo; SELKIRK, Paul. *Port Control Protocol (PCP)* [RFC 6887]. RFC Editor, 2013. Request for Comments, no. 6887. ISSN 2070-1721. Available from DOI: 10.17487/RFC6887.

66. GONT, Fernando. *Implementation Advice for IPv6 Router Advertisement Guard (RA-Guard)* [RFC 7113]. RFC Editor, 2014. Request for Comments, no. 7113. Available from DOI: 10.17487/RFC7113.

67. DUKHOVNI, Viktor; HARDAKER, Wes. *The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance* [RFC 7671]. RFC Editor, 2015. Request for Comments, no. 7671. Available from DOI: 10.17487/RFC7671.

68. BORTZMEYER, Stéphane. *DNS Query Name Minimisation to Improve Privacy* [RFC 7816]. RFC Editor, 2016. Request for Comments, no. 7816. Available from DOI: 10.17487/RFC7816.

69. COLITTI, Lorenzo; CERF, Dr. Vinton G.; CHESHIRE, Stuart; SCHINAZI, David. *Host Address Availability Recommendations* [RFC 7934]. RFC Editor, 2016. Request for Comments, no. 7934. Available from DOI: 10.17487/RFC7934.

70. MRUGALSKI, Tomek; KINNEAR, Kim. *DHCPv6 Failover Protocol* [RFC 8156]. RFC Editor, 2017. Request for Comments, no. 8156. ISSN 2070-1721. Available from DOI: 10.17487/RFC8156.

71. DEERING, Dr. Steve E.; HINDEN, Bob. *Internet Protocol, Version 6 (IPv6) Specification* [RFC 8200]. RFC Editor, 2017. Request for Comments, no. 8200. ISSN 2070-1721. Available from DOI: 10.17487/RFC8200.

72. MRUGALSKI, Tomek; SIODELSKI, Marcin; VOLZ, Bernie; YOURTCHENKO, Andrew; RICHARDSON, Michael; JIANG, Sheng; LEMON, Ted; WINTERS, Timothy. *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* [RFC 8415]. RFC Editor, 2018. Request for Comments, no. 8415. Available from DOI: 10.17487/RFC8415.

73. MARTINEZ, Jordi Palet; LIU, Hans M.-H.; KAWASHIMA, Masanobu. *Requirements for IPv6 Customer Edge Routers to Support IPv4-as-a-Service* [RFC 8585]. RFC Editor, 2019. Request for Comments, no. 8585. Available from DOI: 10.17487/RFC8585.

74. FARRER, Ian; KOTTAPALLI, Naveen; HUNEK, Martin; PATTERSON, Richard. *DHCPv6 Prefix Delegating Relay Requirements* [RFC 8987]. RFC Editor, 2021. Request for Comments, no. 8987. Available from DOI: 10.17487/RFC8987.

75. POSTEL, Jon. *Comments on Internet Protocol and TCP* [Internet Experiment Note]. RFC Editor, 1977. Available also from: https://www.rfc-editor.org/ien/ien2.txt. IEN. RFC Editor.

76. CERF, Vint. *A Proposed New Internet Header Format* [Internet Experiment Note]. RFC Editor, 1978. Available also from: https://www.rfc-editor.org/ien/ien26.pdf. IEN. RFC Editor.

77. POSTEL, Jonathan B. *Draft Internetwork Protocol Specification Version 2* [Internet Experiment Note]. RFC Editor, 1978. Available also from: https://www.rfc-editor.org/ien/ien28.pdf. IEN. RFC Editor.

78. POSTEL, Jonathan B. *Internetwork Protocol Specification Version 4* [Internet Experiment Note]. RFC Editor, 1978. Available also from: https://www.rfc-editor.org/ien/ien41.pdf. IEN. RFC Editor.

79. POSTEL, Jonathan B. *Latest Header Formats* [Internet Experiment Note]. RFC Editor, 1978. Available also from: https://www.rfc-editor.org/ien/ien44.pdf. IEN. RFC Editor.

80. POSTEL, Jonathan B. *Internetwork Protocol Specification Version 4* [Internet Experiment Note]. RFC Editor, 1978. Available also from: https://www.rfc-editor.org/ien/ien54.pdf. IEN. RFC Editor.

81. BOUCADAIR, Mohamed; BURGEY, Eric. *A64: DNS Resource Record for IPv4-Embedded IPv6 Address*. Internet Engineering Task Force, 2010. Available also from: https://datatracker.ietf.org/doc/html/draft-boucadair-behave-dns-a64-02. Internet-Draft. Internet Engineering Task Force. Work in Progress.

82. KORHONEN, Jouni; SAVOLAINEN, Teemu. *EDNS0 Option for Indicating AAAA Record Synthesis and Format*. Internet Engineering Task Force, 2011. Available also from: https://datatracker.ietf.org/doc/html/draft-korhonen-edns0-synthesis-flag-02. Internet-Draft. Internet Engineering Task Force. Work in Progress.

83. WING, Dan. *Learning the IPv6 Prefix of a Network's IPv6/IPv4 Translator*. Internet Engineering Task Force, 2009. Available also from: https://datatracker.ietf.org/doc/html/draft-wing-behave-learn-prefix-04. Internet-Draft. Internet Engineering Task Force. Work in Progress.

84. BOUCADAIR, Mohamed; LEVIS, Pierre; GRIMAULT, Jean-Luc; SAVOLAINEN, Teemu; BAJKO, Gabor. *Dynamic Host Configuration Protocol (DHCPv6) Options for Shared IP Addresses Solutions*. Internet Engineering Task Force, 2009. Available also from: https://datatracker.ietf.org/doc/html/draft-boucadair-dhcpv6-shared-address-option-01. Internet-Draft. Internet Engineering Task Force. Work in Progress.

85. CERF, V.; KAHN, R. A Protocol for Packet Network Intercommunication. *IEEE Transactions on Communications*. 1974, vol. 22, no. 5, pp. 637–648. ISSN 0090-6778. Available from DOI: 10.1109/TCOM.1974.1092259.

86. *Dynamic Host Configuration Protocol for IPv6 (DHCPv6)* [online]. Los Angeles, US, 2019 [visited on 2019-06-18]. Available from: https://www.iana.org/assignments/dhcpv6-parameters/dhcpv6-parameters.xhtml.

87. *Domain Name System (DNS) Parameters* [online]. Los Angeles, US, 2021 [visited on 2021-09-13]. Available from: https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml.

88. *Internet Protocol Version 6 Address Space* [online]. Los Angeles, US, 2021 [visited on 2021-01-20]. Available from: https://www.iana.org/assignments/ipv6-address-space/ipv6-address-space.xhtml.

89. *ICAAN Research: TLD DNSSEC Report* [online]. Los Angeles, California, USA: Internet Corporation For Assigned Names and Numbers, 2019 [visited on 2021-09-21]. Available from: http://stats.research.icann.org/dns/tld_report/.

90. *Available Pool of Unallocated IPv4 Internet Addresses Now Completely Emptied: The Future Rests with IPv6* [online]. Internet Corporation For Assigned Names and Numbers, 2011 [visited on 2021-03-16]. Available from: https://itp.cdn.icann.org/en/files/announcements/release-03feb11-en.pdf.

91. *Information technology – Open Systems Interconnection – Basic Reference Model: The Besic Model*. Geneva, CH, 1994. Standard. International Organization for Standardization.

92. *Data networks and open system communications, Open systems interconnection – Model and Notation*. Geneva, CH, 1994. Standard. ITU Telecomunication Standardization Sector.

93. IEEE Standard for Local and metropolitan area networks–Port-Based Network Access Control. *IEEE Std 802.1X-2010 (Revision of IEEE Std 802.1X-2004)*. 2010, pp. 1–205. Available from DOI: 10.1109/IEEESTD.2010.5409813.

Europass životopis

Osobní údaje

Jméno / Příjmení	**Ing. Bc. Martin Huněk, M.Eng.**
Státní příslušnost	česká

Vzdělání

Roky	2006 – 2010
Škola	Střední průmyslová škola strojní a elektrotechnická a Vyšší odborná škola Liberec
Zakončení	maturita s vyznamenáním
Roky	2010 – 2013
Škola	Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, obor Elektronické informační a řídící systémy
Zakončení	červený diplom, titul bakalář
Roky	2012 – 2014
Škola	Technická univerzita v Liberci, Ústav zdravotnických studií, obor Biomedicínská technika
Zakončení	červený diplom, titul bakalář
Roky	2013 – 2015
Škola	Hochschule Zittau/Görlitz, Fakultät Elektrotechnik und Informatik, obor Mechatronics
Zakončení	diplom, titul Master of Engineering
Roky	2013 – 2016
Škola	Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, obor Mechatronics
Zakončení	červený diplom, titul inženýr
Roky	2015 – dosud
Škola	Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, obor Technická kybernetika
Zakončení	předpokládané ukončení: 2022 (udělovaný titul: Doktor)

Kurzy, semináře a konference

Rok	2015
Akce	Technická univerzita v Liberci, Konference: Ošetřovatelství bez hranic
Typ účasti	aktivní, příspěvek: „Bezpečnost zdravotnických informací a bezpečné chování v počítačových sítích“
Rok	2015
Akce	Brandenburg University of Technology Cottbus-Senftenberg, DCPS – Dependable Cyber Physical Systems
Typ účasti	aktivní, příspěvek: „Anthropometric Measurements of Hollow Bone Structures based upon Computer Assisted Tomography“
Rok	2017
Akce	2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their application to Mechatronics (ECMSM)
Typ účasti	aktivní, příspěvek: „Design and optimisation of NiTi pressure gauge“
Rok	2017
Akce	RIPE NCC Academy, IPv6 Basic Training + IPv6 Advanced Training
Typ účasti	pasivní
Rok	2017
Akce	CESNET: Seminář IPv6 2017
Typ účasti	aktivní, příspěvek: „Implementace IPv6 v síti Freenet Liberec“
Rok	2018
Akce	2018 International Conference on Applied Electronics (AE)
Typ účasti	aktivní, příspěvek: „DNSSEC in the networks with a NAT64/DNS64“
Rok	2019
Akce	IETF 104
Typ účasti	aktivní, příspěvek: „NAT64/DNS64 detection via SRV Records“
Rok	2019
Akce	CESNET: Seminář IPv6 – 2019
Typ účasti	aktivní, příspěvek: „Automatické objevování prefixů pro NAT64“
Rok	2020
Akce	Computer Networks 2020
Typ účasti	aktivní, příspěvek: „Detection of NAT64/DNS64 by SRV Records“
Rok	2021
Akce	CESNET: On-line seminář nejen o IPv6
Typ účasti	aktivní, příspěvek: „Autokonfigurace tisíckrát jinak aneb jak můj router získá adresy?“

Zaměstnání a odborné praxe

Období	2007 – 2015
Zaměstnavatel	Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií, Ústav informačních technologií a elektrotechniky, laboratoř PCBLab
Pozice	laborant
Období	2013, 3 týdny
Zaměstnavatel	Institut klinické a experimentální medicíny, Praha
Pozice	povinná studentská praxe
Období	2014 – 2015
Zaměstnavatel	Technická univerzita v Liberci, Ústav pro nanomateriály, pokročilé technologie a inovace, Edutech
Pozice	Lektor junior
Období	2014 – dosud
Zaměstnavatel	Freenet Liberec, z.s.
Pozice	Správce sítě, dobrovolník
Období	2016 – dosud
Zaměstnavatel	Technická univerzita v Liberci, Správa sítě
Pozice	Správce Linuxových serverů
Období	2017 – 2017
Zaměstnavatel	České vysoké učení technické v Praze, Fakulta elektrotechnická
Pozice	Stáž
Období	2018 – 2019
Zaměstnavatel	Střední průmyslová škola Strojní a Elektrotechnická a Vyšší odborná škola
Pozice	Učitel odborných předmětů
Období	2020 – dosud
Zaměstnavatel	Cerberos s.r.o.
Pozice	Správce sítě - L3 specialista

Pedagogická praxe:

Předmět	Lektor odborných předmětů (Edutech) - elektronika a programování
Délka	2 roky
Předmět	Elektronika (cvičení)
Délka	2 semestry
Předmět	Elektronická dokumentace (cvičení)
Délka	4 semestry
Předměty	Informační a telekomunikační technika, hardware, sítě a operační systémy - střední škola
Délka	1 rok
Předměty	Letecké právo a postupy, Letové přístroje, Komunikace - Instruktor pozemní přípravy
Délka	3 roky

Účasti na grantech a projektech

Reg. číslo	MPO: FR-TI3/751
Název	Biometrické signály – jejich snímání, vyhodnocování a přenos ve zdravotnickém a pečovatelském prostředí
Role v projektu	výroba desek plošných spojů pro akustické signály (služby)
Reg. číslo	CZ.1.07/2.3.00/45.0011
Název	VZDĚLÁVÁNÍ PRO EFEKTIVNÍ TRANSFER TECHNOLOGIÍ A ZNALOSTÍ V PŘÍRODOVĚDNÝCH A TECHNICKÝCH OBORECH
Role v projektu	zajišťování kurzů
Reg. číslo	TAČR: TA04011720
Název	Zlomeniny pánve
Role v projektu	vyhodnocování obrazových dat (služby)
Reg. číslo	MPO 56925/18/61200/3576
Název	Integrace mikropočítačů do osvětlovacích systémů
Role v projektu	návrh a programování řídících systémů (služby)

Oponované práce

HUNĚK, Martin. *Řízení a automatizace technologické linky výroby DPS*. Liberec, 2012. Bakalářský projekt. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií. Vedoucí práce Ing. Leoš Petržílka.

HUNĚK, Martin. *Návrh a realizace řídící jednotky výrobní linky povrchových úprav plošných spojů*. Liberec, 2013. Bakalářská práce. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií. Vedoucí práce Ing. Leoš Petržílka.

HUNĚK, Martin. *Realizace programu pro optimalizace tvaru a velikosti vnitřních pánevních dlah z řezů CT*. Liberec, 2014. Bakalářská práce. Technická univerzita v Liberci, Ústav zdravotnických studií. Vedoucí práce MUDr. Jaroslav Šrám.

HUNĚK, Martin. *Statistical Data Processing of Inner Pelvis Anatomy Structure from CT Images*. Liberec, 2014. Magisterský projekt. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií. Vedoucí práce Ing. Martin Kysela.

HUNĚK, Martin. *Optimization of the Ni-Ti pressure sensors*. Liberec, 2015. Master thesis. Hochschule Zittau/Görlitz, Fakultät Elektrotechnik und Informatik. Vedoucí práce Ing. Martin Kysela.

HUNĚK, Martin. *Optimization of the Ni-Ti pressure sensors*. Liberec, 2016. Master thesis. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií. Vedoucí práce Ing. Martin Kysela.

Publikace

Příspěvek ve sborníku	HUNĚK, Martin. Bezpečnost zdravotnických informací a bezpečné chování v počítačových sítích. In: *Ošetřovatelství bez hranic*. Liberec: Technická univerzita v Liberci, 2015, s. 18-22. 55-015-15. ISBN 978-80-7494-193-1.
Příspěvek ve sborníku	HUNĚK, Martin. Anthropometric Measurements of Hollow Bone Structures based upon Computer Assisted Tomography. In: *DCPS – Dependable Cyber Physical Systems*. Cottbus: Brandenburg University of Technology Cottbus-Senftenberg, 2015. S. 48 – 49.
Příspěvek ve sborníku	KANIOVÁ, E. a HUNĚK, M. [EL] – variable light. In: *Workshop for Ph.D. students of fakulty of textile ingeneering and faculty of mechanical ingeneering TUL*. Liberec: Technical university of Liberec, 2015. S. 87 – 91. ISBN 978-80-7494-229-7.
Příspěvek ve sborníku	HUNĚK, M. Zpracování signálů EEG na obvodech FPGA. *Počítačové architektury & diagnostika*. Brno: Vysoké učení technické v Brně, 2016. S. 22 – 24. ISBN 9788021453760.
Příspěvek ve sborníku	HUNĚK, M. a Z. PLÍVA. Design and optimisation of NiTi pressure gauge. In: *2017 IEEE International Workshop of Electronics, Control, Measurement, Signals and their Application to Mechatronics (ECMSM)*. San Sebastian: IEEE, 2017. S. 1 – 4.DOI: 10.1109/-ECMSM.2017.7945902. ISBN 978-1-5090-5582-1.
Příspěvek ve sborníku	HUNĚK, M. a Z. PLÍVA. DNSSEC in the networks with a NAT64/DNS64. *2018 International Conference on Applied Electronics (AE)*. 1. vyd. Plzeň: University of West Bohemia, Pilsen, Czech Republic, 2018. S. 51 – 54. ISBN 978-802610721-7, ISSN 1803-7232.
Příspěvek ve sborníku	HUNĚK, M. a Z. PLÍVA, *Detection of NAT64/DNS64 by SRV Records: Detection Using Global DNS Tree in the World Beyond Plain-Text DNS*, Communications in Computer and Information Science, Springer Nature, 1, ISBN: 978-303050718-3, p. 27-40, 14 pages, ISSN: 1865-0929, n. 1231, DOI: 10.1007/978-3-030-50719-0_3, 2020
Internetový standard	FARRER, I., N. KOTTAPALLI, M. HUNĚK, R. PATTERSON, *RFC 8987 DHCPv6 Prefix Delegating Relay Requirements*, Request for Comments, RFC Editor, 1, p. 1-11, 11 pages, DOI: 10.17487/RFC8987. ISSN: 2070-1721, 2021 Available also from: https://rfc-editor.org/rfc/rfc8987.txt. Internet Engineering Task Force.
Návrh standardu	HUNEK, Martin. *NAT64/DNS64 detection via SRV Records*. Internet Engineering Task Force, 2021. Available also from: https://datatracker.ietf.org/doc/html/draft-hunek-v6ops-nat64-srv-00. Internet-Draft. Internet Engineering Task Force. Work in Progress.

Znalost jazyků

Mateřský jazyk(y) **čeština**

Sebehodnocení
Evropská úroveň(*)

	Porozumění			**Mluvení**				**Psaní**	
	Poslech		Čtení	Ústní interakce		Samostatný ústní projev			
angličtina	C1	Zkušený uživatel	C1	Zkušený uživatel	C1	Zkušený uživatel	C1	Zkušený uživatel	C1	Zkušený uživatel

(*) *Společný evropský referenční rámec pro jazyky*

Technické schopnosti a dovednosti

Programovací jazyky	C, C++, C++/Qt, MySQL, Bash + základy LabVIEW, Lua a MATLAB
Další znalosti	Dynamické routování BGP a OSPF
	Správa e-mailových serverů, DNS resolverů a Web serverů
	Návrh a provozování sítí IPv6
	Správa počítačových sítí a systémů GNU/Linux
	Aktivní člen RIPE APWG
	Aktivní člen IETF pracovních skupin 6man a v6ops
	Návrh DPS
	Vyhodnocování obrazových dat
	Uživatelská znalost LaTeXu

Další kvalifikace

Pilotní kvalifikace: SPL, CPL(A), SEP land, ICAO+IFR English, ATPL(A) teorie
Řidičský průkaz skupin: A, B, BE, C, CE
Vyhláška 50/1978 Sb., §6, §7, §8, §10 a §11
Oprávnění k výkonu zdravotnického povolání dle zákona 96/2004 Sb., §20