Network Working Group
Internet-Draft
Intended status: Informational
Expires: 3 January 2022

J. Arkko
J. Novotny
Ericsson

2 July 2021

Privacy Improvements for DNS Resolution with Confidential Computing
draft-arkko-dns-confidential-02

Abstract

Data leaks are a serious privacy problem for Internet users. Data in flight and at rest can be protected with traditional communications security and data encryption. Protecting data in use is more difficult. In addition, failure to protect data in use can lead to disclosing session or encryption keys needed for protecting data in flight or at rest.

This document discusses the use of Confidential Computing, to reduce the risk of leaks from data in use. Our example use case is in the context of DNS resolution services. The document looks at the operational implications of running services in a way that even the owner of the service or compute platform cannot access user-specific information produced by the resolution process.

Table of Contents

1. Introduction

DNS privacy has been a popular topic in the last few years, and continues to be. The issues with regards to privacy are first that domain name meta-data is visible on the wire, even when the actual communications are encryped. This is being addressed with better technology.

But even if the meta-data is hidden inside communications, any DNS resolvers still have the potential too see users' entire browsing history. This is particularly problematic, given that commonly used large public or operator resolver services are an obviously

attractive target, for both attacks and for commercial or other use of information visible to them.

A lot of work is ongoing in the industry and the IETF to address some of these issues:

* Work on encrypted DNS query protocols to hide the meta-data related to domain names.

* Discovery mechanisms. These may enable a bigger fraction of DNS query traffic to move to encrypted protocols, and may also help distributed queries to different parties to avoid concentrating all information in one place.

* Practices, expectations, contracts (e.g., [RFC8932], Mozilla's trusted recursive resolver requirements [MozTRR])

* Improvements outside DNS (e.g., encrypted Server Name Indication (eSNI) [I-D.ietf-tls-esni]).

* General technology developments (e.g., confidential computing, attestations, remote attestation work at the IETF RATS WG, and so on)

The goal of this document is to build on all that work - and assume all communications are or become encrypted, including the DNS traffic. Our question is what problems remain? Is there a next step?

Our worry is that resolvers can be a major remaining source of leaks, e.g., through accidents, attacks, commercial use, or requests from the authorities. We need to protect user's data in flight, at rest, or in use - we wanted to experiment with technology that could reduce leaks on the last two cases. Confidential Computing is one such potential technology, but it is important to talk about it and get broader feedback. The use of this technology does have some operational impacts.

Our primary conclusions are that data held by servers should receive at least as much security attention as communications do. The authors feel that this is particularly crucial for DNS, due to the potential to leak of users' browsing histories, but principles apply also to other services.

As a result, all applicable tools should be considered, including confidential computing that is discussed in this document. However, the operational and business implications of such tools should be considered. Feedback to us is very welcome. Are these approaches

feasible or infeasible? What aspects need to be taken into account to successfully apply them?

2. Background

Communications security has been at the center of many security improvements in the Internet. The goal has been to ensure that communications are protected against outside observers and attackers [RFC3552] [RFC7258]. Communications security is, however, not sufficient by itself, and continuing success in better protection of communications is highlighting the need to address other issues.

In particular, more attention needs to be paid to protecting data not just in flight but also at rest or in use. User data leaks that can occur from servers and other systems, through accidents, attacks, commercial use of data, and requests for information by authorities. Both data at rest and data in use needs to be protected. Being able to protect data in use provides also benefits to protecting keys used for protecting data in flight and at rest.

Data leaks are very common, and include highly publicized ones or ones with significant consequences, such as [Cambridge]. Data leaks are also not limited to traditional computer applications, but can also impact anything from private health data [Vastaamo] to children's toys [Toys] or smart TVs [SmartTV].

The general issue and possible solutions have been discussed extensively elsewhere, e.g., [Digging], [Mem], [Comparison], [Innovative], [AMD], [Efficient], [CCC-Deepdive], [CC], and so on. The Internet-relevant angle has also been discussed in few documents, e.g., [I-D.lazanski-smart-users-internet], [I-D.iab-dedr-report] [I-D.arkko-farrell-arch-model-t-redux], and so on. The topic is also related to best practices for protocol and network architecture design, and what information can be provided to what participants in a system, see, e.g. [RFC8558] [I-D.thomson-tmi] [I-D.arkko-arch-infrastructure-centralisation].

Data leaks can occur in user-visible services that user has chosen to use and agreed to provide information to (at least in theory [Unread]). But leaks can also occur in other types of services, that are part of the infrastructure, such as DNS resolution services or parts of the communication infrastructure.

This document looks at the possibility of using a specific technical solution, Confidential Computing [CCC-Deepdive], to reduce the risk of leaks from data in use. We consider the operational implications of running services in a way that even the owner of the service or

compute platform cannot access user-specific information that is produced as a side-effect of the service.

We explore the use of Confidential Computing in the context of DNS resolution services [RFC1035]. This is a nice and relatively simple example, but there are of course potential other applications as well.

DNS resolution services are of course also an important case where privacy matters a lot for the users. Threats against the resolution process could prevent the user from accessing services. Data leaks from the process have the potential to expose the user's entire browsing history.

The use of Confidential Computing in the DNS context has been also discussed in other documents, e.g., [PDoT] and [I-D.reddy-add-server-policy-selection].

The DNS privacy issues have been also discussed in multiple documents, such as [RFC7626] [RFC8324] and so on.

3. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

4. Prerequisities

The primary sources of leaks are as follows:

* Communications interception. This threat can be addressed by encrypted communications, such as the use of DNS-over-TLS (DoT) [RFC7858], DNS-over-HTTPS (DoH) [RFC8484], or DNS-over-QUIC (DoQ) [I-D.ietf-dprive-dnsoquic] instead of traditional DNS protocols.

* Data leakage from the server or service, either from data at rest or in use. This can be addressed by encrypting the data while at rest and employing the techniques discussed in this document for data in use.

The specific information that is privacy sensitive depends on the application. In DNS resolution application it is clear that the users' browsing histories, i.e., which users asked for what names is privacy sensitive, and protecting that information is the primary focus in this document. In contrast, the domains themselves or the

associated address information is in the general case public and not privacy sensitive. However, in some cases even this information may be sensitive, such as in the case of internal domains of a corporate network. Information not related to individuals may also be sensitive in some cases, e.g., the collective browsing destinations of an entire organization.

The above was also observed in [RFC7626] which stated the following:

"DNS data and the results of a DNS query are public [...], and may not have any confidentiality requirements. However, the same is not true of a single transaction or a sequence of transactions; that transaction is not / should not be public."

Nevertheless, it should be noted that technology can help only insofar as there is commercial willingness to provide the best possible service and to protect the users' information.

Similarly, the techniques discussed in this document are not the sole, or full answer to all problems. There are a lot of technical, operational, and governance issues that also matter and practices that help. A good compilation of some best practices can be found in [RFC8932], and particularly Section 5.2 that discusses data at rest.

5. Confidential Computing

Confidential Computing is about protecting data in use by performing computation in a hardware enforced Trusted Execution Environment (TEE) [CCC-Deepdive]. It addresses the need to protect data in use, which traditionally has been hard to achieve. It may also help improve the encryption of data in flight and at rest, by helping protect session keys and other security information used in that process.

For our purposes, we focus on Trusted Execution Environments that use computer hardware to provide the following characteristics:

* Attestability: The environment can provide verifiable evidence to others (such as client using services running on it) about the environment, its characteristics, and the software it runs.

* Code integrity: Unauthorized entities cannot modify software being run within the environment.

* Data confidentiality and integrity: Unauthorized entities cannot view or modify data while it is in use within the TEE.

These characteristics have been paraphrased from [CCC-Deepdive]. See also [I-D.ietf-rats-architecture] for details of attestation. There are additional characteristics that matter in some situations, but for our purposes the above ones are central.

Specific technologies to perform Confidential Computing or run TEEs are becoming common in CPUs, operating systems, and other supporting software. For instance, Intel's Software Guard Extension (SGX) [SGX] is one CPU manufacturer's approach to this technology. SGX allows application developers to run software protected in a secure enclave protected by the CPU, including for instance encrypting all memory accesses outside the CPU and being able to provide remote attestation to outsiders about which software image is being run. These secure enclaves are the SGX approach to providing a TEE.

Confidential Computing is also becoming available on commonly available cloud computing services. When a user employs these services, they have the ability to run software and process data that even the owner of the cloud system does not have access to.

Interestingly, that is quite a contrast to the worries expressed some years ago about Trusted Computing technology, when it was feared that it enabled running software in users' computers that could act against the interests of the user in some cases, such as when protecting media files [Stallman]. While those concerns may apply even today in some cases, it is clear that whe the user can get secure information about services running somewhere in the network, this is an advantage for the users.

Note that availability might be another desirable characteristic for Confidential Computing systems, but it is one that is not in any special way supported by current technology. Ultimately, the owner of the computer still has the ability to choose when to switch the computer off, for instance. There is also no particular hardware technology at this time to deal with Denial-of-Service attacks. Some of the software techniques related to dealing with Denial-of-Service attacks are discussed in the Security Considerations section.

6. Using Confidential Computing for DNS Resolution

Confidential Computing can be used to provide a privacy-friendly resolution service in a server.

The basic arrangement is two-fold:

* User's computer and the DNS resolution server communicate using an encrypted and integrity protected transport protocol, such as DoT or DoH [RFC7858] [RFC8484].

* The secure connection terminates inside a TEE running in the the DNS resolution server. This TEE performs all the necessary processing to respond to the user's query. The TEE will not provide any user-specific information outside of the TEE, such as logs of what names specific clients queried for.

  The TEE may need to contact other local servers or in the Internet to resolve a query that has no recently cached answer. We will discuss later how this can be done securely: it is necessary to prevent the linking any external actions such as receiving a client request and observing a query going out to other DNS servers in the Internet.

The arrangement is shown in Figure 1.

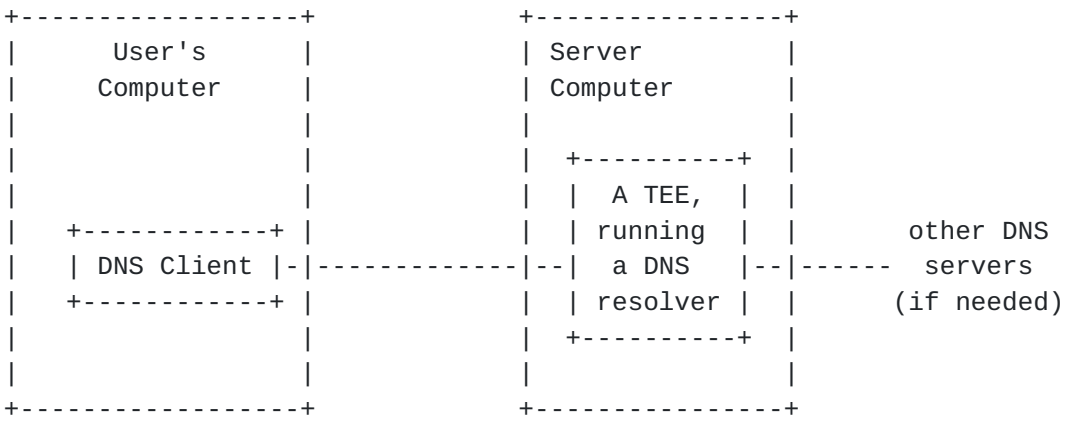

Figure 1: Confidential Computing for DNS Resoluton

In this application, we strive to have no data at rest at all, at least nothing that relates directly to users. Data in flight and data in use are both protected by encryption. As a result of running the resolution service in this manner, any user-specific information should remain within the TEE, and not exposed to outsiders or even the owner of the service or the compute platform where the service is running in.

The authors believe that this is a desirable property. However, it remains to assure users and clients that the service is actually run in this manner. This can be done in two ways:

* Through off-line reliance on a particular service, i.e., a human decision to use a particular system. Once there is a decision to use a particular system, cryptographic means such as public keys may be used to ensure that the client is indeed connected to the expected server. However, there is no guarantee that the human-

space statements about the practices used in running the server are valid.

* Cryptographic check that the service is actually running inside a valid TEE and that it runs the expected software. Such checks needs to rely on third parties. The attestation verification is performed by a verifier - that can be either user's computer or a designated verifier as discussed in [I-D.ietf-rats-architecture] and [I-D.voit-rats-attestation-results] The verifier checks that (a) the cryptographic attestation refers to a server machine that is acceptable to the user (e.g., manufactured by a manufacturer it trusts, CPU features considered secure are used, features considered insecure are turned off, etc.) (b) that the software image designated as being run in the attestation is a software image that the relying party (end user) is willing to use (e.g., has a hash that matches a known software that does not log user actions, or is vouched as trustworthy by another party that the relying party trusts).

7. Operational Considerations

This section discusses some aspects of the Confidential Computing arrangement for DNS, based on the authors' experience with these systems.

7.1. Operations

Given that the service executes confidentially, and is not observable even by the owner of the hardware, the operations model becomes different. Some different models may be applied:

* The service executes on a hardware platform (such as a commercial cloud service) that has no access to information, but there is some other management entity that does have access. The control functions of this entity can communicate with the service instances running in TEEs, and have access to the internal state and statistics of the service instances.

* Truly confidential operations where the service and hardware owners have decided to deploy a service that really does not expose private user information to anyone, including themselves.

It is not clear how the first model differs from currently deployed service models. It merely makes it possible to run a service without exposing information to, say, the cloud provider, but any data collection about user behaviours would still be possible for the service owner.

As a result, this document focuses mostly on the second model. For some functions, such as DNS resolution, it is possible to hide all user-related information, and our document argues that we should do so.

Of course, the owners of a service do need some information to run the service, from an efficiency, scaling, problem tracking, and security monitoring point of view. The service operator may even benefit from seeing some overall trend information about various queries and traffic. This does not have to mean exposing individual user behaviours, however.

The authors have worked with aggregate statistics to be able to provide load, performance, memory usage, cache statistics, error, and other information out of the confidential processes. This helps the operator understand the health and status of various service instances. Even with aggregate statistics, there are some danger of revealing private information. For instance, even a sum of counters across all clients can reveal counters associated with an individual user, if the aggregate counters can be sampled at any time with arbitrary precision. For instance, the actions of a single client can be determined by sampling the statistics before and after that client sent a message.

A simplistic approach to producing safer statistics in such cases is to truncate and/or obfuscate the least significant bits of the statistics. It is often necessary to tailor such truncation to the types of measurements, e.g., number of requests is typically a very large number while the number of specific errors is usually small. Truncation could of course be done dynamically. More generally, the set of information provided to the operator about the confidential process could be viewed in light of differential privacy.

Another complementary approach is to provide statistics only at set intervals, or after a sufficient amount of new traffic has been received.

Another complementary technique to monitor the health of confidential services is the use of probes to ensure that the services function correctly. Probes can also measure the performance of the services.

The case of excessive service conditions due to Denial-of-Service attacks is discussed further under the Security Considerations section.

7.2. Debugging

Various error conditions and software issues may occur, as is usual with any service. There is a need to monitor problems that occur inside the service or at the client. This can be done, for instance, with the help of various statistics discussed earlier.

Some of the monitored conditions should include:

* All major (or preferably even minor) error conditions should have an associated counter. This is necessary as no traditional logging can be reasonably provided that would otherwise have entries for, say, "client IP 203.0.113.0 sent a malformed request". While some errors can be expected at any time, a major increase in specific issues can indicate a problem. As a result, the counters need to be monitored and issues investigated as needed.

* Client connection failures, which might indicate software version, trust root or other configuration problems.

Of course, for dedicated software testing purposes (such as debugging interoperability problems), even confidential services need to be run in a mode that exposes everything. Actual clients and users MUST be able to ensure that they are connected to a production service instance. This can be be done by providing debugging status as part of the remote attestation, so that clients can verify it is off. Alternatively, testing versions of the service are simply not listed as trusted software versions.

7.3. Dependencies

The use of Confidential Computing introduces three additional dependencies to the system:

There is a need to be able to verify that the CPU executing the service is a legitimate CPU with the right hardware, and that the software being run for the service is acceptable. While this can be hard coded information in the service clients, in practice there is often a need to rely on other parties for scalability. As a result, there are two dependencies for legitimate CPU verification and for checking acceptable software versions. These are services that need to be run, and/or their use need to be agreed and possibly contracted for. The CPU manufacturer often plays a role in the CPU verification.

The third dependency is on the client. Depending on specific protocol arrangements, Confidential Computing services often can

serve unmodified clients, but for the full benefits and for validating attestations or software images, client changes are necessary. The necessary communications may happen as part of TLS negotiations or other general purpose protocols [I-D.mandyam-tokbind-attest], [I-D.ietf-rats-eat].

7.4. Additional services

Many services employ information that can be used to perform additional services beyond the basic task. For instance, knowledge about what the users requests or who the user is can be used for various optimizations or additional information that can be delivered to the user. Or the user can provide some additional information that is taken into account by the service.

One concern with these types of additional services is that the information used by them can be privacy sensitive. But Confidential Computing can assist in this as well, as long as the relevant information stays only within the TEE, it is better protected than by, e.g., providing that extra information to a regular service on the Internet.

Conversely, care needs to be taken whenever the service needs to relay some information outside the TEE. Some specific situations where this is needed with DNS are discussed in Section 7.1.

One example of additional services is that aggregate, privacy-sensitive data may be produced about trends in a confidentially run service, if it will not be possible to separate individual users from that data. For instance, it would be difficult sell information about individual users to help with targeted advertising, but the overall popularity of some websites could be measured.

7.5. Performance

Confidential Computing technology may impact performance. Nakatsuka et al. [PDoT] report on DNS resolution within a TEE where their solution could outperform the open source Unbound DNS server in certain scenarios, especially in situations where there are not a lot of DNS client connections. We concur their suggestion that at current stage of Confidential Computing technology, possible implementations may be more suited for local DNS resolution services rather than global scale implementation, where the performance hit would be much more significant. Nonetheless with Confidential Computing technology ever evolving we believe the low performance overhead solutions will be possible in foreseeable future.

Other things being equal there's likely some performance hit, as current Confidential Computing technology typically involves separating a server into two parts, the trusted and untrusted parts. In practice, all communications need to go through both, and the communication between the two parts consumes some cycles. There are also current limitations on amount of memory or threads supported by these technologies. However, newer virtualization-based confidential computing TEE approaches are likely going to improve these aspects.

Another performance hit comes from the overhead related to running the attestation process, and passing the necessary extra information in the communications protocols with the clients. In general, this works best when the cost of the setup is amortized over a long-lived session. Such sessions may exist between DoT/DoH-enabled clients and resolvers. Also, there are many possible arrangements and possible parties involved in attestation, see [I-D.ietf-rats-architecture].

8. Security Considerations

Security issues in this arrangement are discussed below.

8.1. Observations from outside the TEE

While a TEE is considered to be secure and not observable, there may be signs outside the TEE that can reveal information.

For instance, a server may receive a request from a client and immediately send out a question to a server in the Internet about a particular domain name. Observers - such as the owner of the server computer or the cloud farm - may be able to link incoming user queries to outgoing questions

Caching, randomly made other traffic, and timing obfuscation can deter such attacks, at least to an extent.

8.2. Trust Relationships

For scaling reasons, the arrangement typically depends on the ability to have trusted parties (a) for attesting the validity of a particular CPU being manufactured by a CPU manufacturer, and (b) for determining whether a particular software image hash is acceptable for the task it is advertising to do.

Such trusted parties need to be configured, which presents an additional operational burden. The information can of course be provided as part of a device manufacturer's or application's initial configuration, or be provided independently similar to how, for instance, certificate authorities are run.

It is important to recognize that mere use of technology is not sufficient to make the system secure. With communications, establishing a secure, encrypted channel is of no use if it is not with the intended party due to a certificate authority that proved to be untrustworthy. With confidential computing, the same applies: one has to have someone who can assert that a CPU is capable of performing the confidential computing task and that the indicated software is good for performing the task that the user expects it to perform. That being said, when such trusted parties can be found, the service performed by the server can become much more privacy friendly.

8.3. Denial-of-Service Attacks

To paraphrase an old philosophical question, "If an evil packet is sent behind the veil of encryption and no one is around to lift it, did an attack happen?" [Chautauquan]

Denial-of-Service attacks are a more serious form of the problems with operating services that the operator (intentionally) does not fully see. There needs to be means to deal with these attacks.

Attacks that can be identified by particularly high traffic flows from externally observable sources (e.g., source IP address) can of course still be dealt with in similar ways as we do in more open server designs.

But this is often not enough, and for this purpose some additional support is needed in the systems, for both detection of attacks and reacting to them.

One detection technique is to use the aggregate/truncated statistics to analyze anomalous behaviour. Another technique is to have the confidential part of the service produce extra information about events that cross a threshold. For instance, a particular error may occur exceptionally frequently, say among millions of requests, and this could warrant exposing either something about the request (e.g., the associated domain name) or something about the client (e.g., connection type, protocol details, or sender address).

The operator of the services needs to be able to react to possible attacks as well. One technique is to be able to provide instruction to the confidential part of the service to refuse service for specific requests (e.g., specific domain names) or for specific clients (e.g., coming from specific addresses). Alternatively, the service can also dynamically react to issues, e.g., by starting to reduce the amount of resources dedicated to some classes of requests that for some reason are starting to require exceptionally high

amount of resources. These techniques do not endanger user privacy, but may of course impact provided service.

8.4. Other vulnerabilities

Like all security mechanisms, this solution is not a panacea. It relies on the correct operation of a number of technologies and entities. For instance, CPU bugs or side channel vulnerabilities can cause information leaks to become possible. While confidential computing offers a layer of protection against attacks even from the owner of the computer hardware or the operating system, it is believed that this protection does not extend to sophisticated physical attacks, such being able to study chips with an electron microscope.

And as discussed above, it is also critical to check what software is being run, as otherwise any possible benefit would be negated by the possibly negligent or nefarious actions the unchecked software makes.

The mechanism does offer an additional layer of defense, however. It allows some of the trust that we place on our cloud platform owners, CPUs, and software applications to be verified and controlled with technical means. It may have some remaining vulnerabilities, but we obviously already depend on, for instance, the correct operation of our computing platforms. As such, Confidential Computing works to reduce some of the vulnerabilities in this area.

It should also be a desirable feature for users. A service that offers Confidential Computing-based protection of user data and can show that its software does not leak user-specific information is likely going to be more attractive to users than one that provides no such assurances. Of course, overall user choice depends on many factors beyond privacy, such as cost, ease of use, switching costs, and so on.

There is also a danger of attacks or pressure from intelligence agencies that could result in, e.g., the use of unpublicized vulnerabilities in an attempt to dwarf the protections in Confidential Computing. This could be used to perform pervasive monitoring, for instance [RFC7258]. Even so, it is always beneficial to push the costs and difficulty for attackers. Requiring parties who perform pervasive monitoring to employ complex technical attacks rather than being able to request logs from a service provider significantly increases the difficulty and risk associated with such monitoring.

9. Recommendations

Data held by servers SHOULD receive at least as much security attention as communications do.

The authors would like to draw attention to the problem of data leaks, particularly for data in use, and RECOMMEND the application of all available tools to prevent inappropriate access to users' information.

This is particularly crucial for DNS resolution services that have the potential to learn user's browsing histories. But the principles apply also to other services.

While using Confidential Computing without other modifications to the service in question is possible, real benefits can only be realized when the actual service is built for the purpose of avoiding data leaks or user data capture. Systems may need to be tuned or modified, for instance they MUST NOT produce logs that would negate purpose of running them inside a TEE to begin with. Mechanisms SHOULD be found to enable debugging and the detection of fault situations and attacks, again without exposing private information relating to individual users.

Some computing services can proceed on their own and require no interaction with the rest of the world. These are easier to secure. Even then, care SHOULD be taken to avoid request-response timing to provide information useful for side-channel attacks. If so, the owner of the server hardware can not determine much about what was going on.

However, other services may require interaction with other systems, such as is the case with a DNS resolver needing to find out a particular name that is not in a cache or whose cache entry has expired. This is because the resolution service is not a self-contained computation task but ultimately needs, at least in some cases, interaction with the rest of the world.

Consequently, the resolver needs to collaborate with other network nodes that are not even in the same administrative domain and cannot be guaranteed to subscribe to the same principles of protecting user's information. In this case, even if communications to other entities are encrypted, the potentially untrusted party at the other end of the communications may leak information.

In such communications, care SHOULD be taken to avoid exposing any information that would identify users, or allow fingerprinting the capabilities of those users' systems. Similarly, care SHOULD be

taken to avoid exposing any timing information that would allow the owner of the server hardware to determine what is going on, e.g., which users are asking for what names. Even so, vulnerabilities may appear if the attacker can force the system to behave in a particular way, by, e.g., forcing cache overflow, overloading it with traffic it knows about, etc.

The situation is slightly different when the interaction is with other systems that form a part of the same administrative domain. In particular, if those other systems employ similar confidential computing setup, and an encrypted channel is used, then some additional security can be provided compared to communicating with other entities in the Internet.

10. Acknowledgments

The authors would like to thank Juhani Kauppi, Jimmy Kjaellman, and Tero Kauppinen for their work on systems supporting some of the ideas discussed in this memo, and Dave Thaler, Daniel Migault, Karl Norrman, and Christian Schaefer for significant feedback on early version of this draft. The author would also like to thank Marcus Ihlar, Maria Luisa Mas, Miguel Angel Munos De La Torre Alonso, Jukka Ylitalo, Bengt Sahlin, Tomas Mecklin, Ben Smeets and many others for interesting discussions in this problem space.

11. References

11.1. Normative References

[RFC1035] Mockapetris, P.V., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

[AMD] Kaplan, D., Powell, J., and T. Woller, "AMD Memory Encryption", AMD White Paper , April 2016.

[Cambridge]
Isaak, J. and M. Hanna, "User Data Privacy: Facebook, Cambridge Analytica, and Privacy Protection", Computer 51.8 (2018): 56-59, https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8436400 , 2018.

[CC] Rashid, F.Y., "What Is Confidential Computing?", IEEE Spectrum, https://spectrum.ieee.org/computing/hardware/what-is-confidential-computing , May 2020.

[CCC-Deepdive]
Confidential Computing Consortium, ., "A Technical Analysis of Confidential Computing", https://confidentialcomputing.io/whitepaper-02-latest , January 2021.

[Chautauquan]
"The Chautauquan", Volume 3, Issue 9, p. 543 , June 1883.

[Comparison]
Mofrad, S., Zhang, F., Lu, S., and W. Shi, "A comparison study of intel SGX and AMD memory encryption technology", HASP '18, Proceedings of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy, Pages 1-8, https://doi.org/10.1145/3214292.3214301 , June 2018.

[Digging] Hammouchi, H., Cherqi, O., Mezzour, G., Ghogho, M., and M. El Koutbi, "Digging Deeper into Data Breaches: An Exploratory Data Analysis of Hacking Breaches Over Time", Procedia Computer Science, Volume 151, pp. 1004-1009, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2019.04.141, https://www.sciencedirect.com/science/article/pii/S1877050919306064 , 2019.

[Efficient]
Suh, G.E., Clarke, D., Gasend, B., van Dijk, M., and S. Devadas, "Efficient memory integrity verification and encryption for secure processors", Proceedings. 36th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO-36, San Diego, CA, USA, pp. 339-350, doi: 10.1109/MICRO.2003.1253207 , 2003.

[I-D.arkko-arch-infrastructure-centralisation]
Arkko, J., "Centralised Architectures in Internet Infrastructure", Work in Progress, Internet-Draft, draft-arkko-arch-infrastructure-centralisation-00, 4 November

2019, <https://www.ietf.org/archive/id/draft-arkko-arch-infrastructure-centralisation-00.txt>.

[I-D.arkko-farrell-arch-model-t-redux]
Arkko, J. and S. Farrell, "Internet Threat Model Evolution: Background and Principles", Work in Progress, Internet-Draft, draft-arkko-farrell-arch-model-t-redux-01, 22 February 2021, <https://www.ietf.org/archive/id/draft-arkko-farrell-arch-model-t-redux-01.txt>.

[I-D.iab-dedr-report]
Arkko, J. and T. Hardie, "Report from the IAB Workshop on Design Expectations vs. Deployment Reality in Protocol Development", Work in Progress, Internet-Draft, draft-iab-dedr-report-01, 2 November 2020, <https://www.ietf.org/archive/id/draft-iab-dedr-report-01.txt>.

[I-D.ietf-dprive-dnsoquic]
Huitema, C., Mankin, A., and S. Dickinson, "Specification of DNS over Dedicated QUIC Connections", Work in Progress, Internet-Draft, draft-ietf-dprive-dnsoquic-02, 22 February 2021, <https://www.ietf.org/archive/id/draft-ietf-dprive-dnsoquic-02.txt>.

[I-D.ietf-rats-architecture]
Birkholz, H., Thaler, D., Richardson, M., Smith, N., and W. Pan, "Remote Attestation Procedures Architecture", Work in Progress, Internet-Draft, draft-ietf-rats-architecture-12, 23 April 2021, <https://www.ietf.org/archive/id/draft-ietf-rats-architecture-12.txt>.

[I-D.ietf-rats-eat]
Mandyam, G., Lundblade, L., Ballesteros, M., and J. O'Donoghue, "The Entity Attestation Token (EAT)", Work in Progress, Internet-Draft, draft-ietf-rats-eat-10, 7 June 2021, <https://www.ietf.org/archive/id/draft-ietf-rats-eat-10.txt>.

[I-D.ietf-tls-esni]
Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", Work in Progress, Internet-Draft, draft-ietf-tls-esni-11, 14 June 2021, <https://www.ietf.org/archive/id/draft-ietf-tls-esni-11.txt>.

[I-D.lazanski-smart-users-internet]
Lazanski, D., "An Internet for Users Again", Work in

Progress, Internet-Draft, draft-lazanski-smart-users-internet-00, 8 July 2019, <https://www.ietf.org/archive/id/draft-lazanski-smart-users-internet-00.txt>.

[I-D.mandyam-tokbind-attest]
Mandyam, G., Lundblade, L., and J. Azen, "Attested TLS Token Binding", Work in Progress, Internet-Draft, draft-mandyam-tokbind-attest-07, 24 January 2019, <https://www.ietf.org/archive/id/draft-mandyam-tokbind-attest-07.txt>.

[I-D.reddy-add-server-policy-selection]
Reddy, T., Wing, D., Richardson, M. C., and M. Boucadair, "DNS Server Selection: DNS Server Information with Assertion Token", Work in Progress, Internet-Draft, draft-reddy-add-server-policy-selection-08, 29 March 2021, <https://www.ietf.org/archive/id/draft-reddy-add-server-policy-selection-08.txt>.

[I-D.thomson-tmi]
Thomson, M., "Principles for the Involvement of Intermediaries in Internet Protocols", Work in Progress, Internet-Draft, draft-thomson-tmi-01, 3 January 2021, <https://www.ietf.org/archive/id/draft-thomson-tmi-01.txt>.

[I-D.voit-rats-attestation-results]
Voit, E., Birkholz, H., Hardjono, T., Fossati, T., and V. Scarlata, "Attestation Results for Secure Interactions", Work in Progress, Internet-Draft, draft-voit-rats-attestation-results-01, 10 June 2021, <https://www.ietf.org/archive/id/draft-voit-rats-attestation-results-01.txt>.

[Innovative]
Ittai, A., Gueron, S., Johnson, S., and V. Scarlata, "Innovative Technology for CPU Based Attestation and Sealing", HASP'2013 , 2013.

[Mem] Henson, M. and S. Taylor, "Memory encryption: a survey of existing techniques", ACM Computing Surveys volume 46 issue 4 , 2014.

[MozTRR] Mozilla, ., "Security/DOH-resolver-policy", https://wiki.mozilla.org/Security/DOH-resolver-policy , 2019.

[PDoT] Nakatsuka, Y., Paverd, A., and G. Tsudik, "PDoT: Private DNS-over-TLS with TEE Support", Digit. Threat.: Res. Pract., Vol. 2, No. 1, Article 3, https://dl.acm.org/doi/fullHtml/10.1145/3431171 , February 2021.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, DOI 10.17487/RFC3552, July 2003, <https://www.rfc-editor.org/info/rfc3552>.

[RFC7258] Farrell, S. and H. Tschofenig, "Pervasive Monitoring Is an Attack", BCP 188, RFC 7258, DOI 10.17487/RFC7258, May 2014, <https://www.rfc-editor.org/info/rfc7258>.

[RFC7626] Bortzmeyer, S., "DNS Privacy Considerations", RFC 7626, DOI 10.17487/RFC7626, August 2015, <https://www.rfc-editor.org/info/rfc7626>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC8324] Klensin, J., "DNS Privacy, Authorization, Special Uses, Encoding, Characters, Matching, and Root Structure: Time for Another Look?", RFC 8324, DOI 10.17487/RFC8324, February 2018, <https://www.rfc-editor.org/info/rfc8324>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[RFC8558] Hardie, T., Ed., "Transport Protocol Path Signals", RFC 8558, DOI 10.17487/RFC8558, April 2019, <https://www.rfc-editor.org/info/rfc8558>.

[RFC8932] Dickinson, S., Overeinder, B., van Rijswijk-Deij, R., and A. Mankin, "Recommendations for DNS Privacy Service Operators", BCP 232, RFC 8932, DOI 10.17487/RFC8932, October 2020, <https://www.rfc-editor.org/info/rfc8932>.

[SGX] Hoekstra, M.E., "Intel(R) SGX for Dummies (Intel(R) SGX Design Objectives)", Intel, https://software.intel.com/content/www/us/en/develop/blogs/protecting-application-secrets-with-intel-sgx.html , September 2013.

[SmartTV] Malkin, N., Bernd, J., Johnson, M., and S. Egelman, "What Can't Data Be Used For? Privacy Expectations about Smart TVs in the U.S.", European Workshop on Usable Security (Euro USEC), https://www.ndss-symposium.org/wp-content/uploads/2018/06/ eurousec2018_16_Malkin_paper.pdf" , 2018.

[Stallman] Stallman, R., "Can You Trust Your Computer?", GNU.org, https://www.gnu.org/philosophy/can-you-trust.html , n.d..

[Toys] Chu, G., Apthorpe, N., and N. Feamster, "Security and Privacy Analyses of Internet of Things Childrens' Toys", IEEE Internet of Things Journal 6.1 (2019): 978-985, https://arxiv.org/pdf/1805.02751.pdf , 2019.

[Unread] Obar, J. and A. Oeldorf, "The biggest lie on the internet{:} Ignoring the privacy policies and terms of service policies of social networking services", Information, Communication and Society (2018): 1-20 , 2018.

[Vastaamo] Redcross Finland, ., "Read this if your personal data was leaked in the Vastaamo data system break-in", https://www.redcross.fi/news/20201029/read-if-your-personal-data-was-leaked-vastaamo-data-system-break , October 2020.

Authors' Addresses

Jari Arkko
Ericsson

Email: jari.arkko@ericsson.com

Jiri Novotny
Ericsson

Email: jiri.novotny@ericsson.com