Authenticated and Secure Automotive Service Discovery with DNSSEC and DANE

Mehmet Mueller
Dept. Computer Science, *Hamburg University of Applied Sciences*, Germany
mehmet.mueller@haw-hamburg.de

***Abstract*—Automotive softwarization is progressing and future cars are expected to operate a Service-Oriented Architecture on multipurpose compute units, which are interconnected via a high-speed Ethernet backbone. The AUTOSAR architecture foresees a universal middleware called SOME/IP that provides the service primitives, interfaces, and application protocols on top of Ethernet and IP. SOME/IP lacks a robust security architecture, even though security is an essential in future Internet-connected vehicles. In this paper, we augment the SOME/IP service discovery with an authentication and certificate management scheme based on DNSSEC and DANE. We argue that the deployment of well-proven, widely tested standard protocols should serve as an appropriate basis for a robust and reliable security infrastructure in cars. Our solution enables on-demand service authentication in offline scenarios, easy online updates, and remains free of attestation collisions. We evaluate our extension of the common *vsomeip* stack and find performance values that fully comply with car operations.**

***Index Terms*—Automotive security, authentication, attestation, service orientation, SOME/IP, AUTOSAR, standards**

I. Introduction

Future cars will connect to the Internet as well as to other vehicles and infrastructure (Vehicle-to-X (V2X)) for improving road safety, traffic efficiency, and driver comfort. This opens a large attack surface across communication interfaces [1]–[3] and in-car software [4], [5]. Nevertheless, current automotive protocols and Electronic Control Units (ECUs) often lack security mechanisms [6] since they were designed for a closed environment.

Service-Oriented Architecture (SOA) is an emerging paradigm for automotive software, which facilitates service provisioning by various suppliers of the OEM. *Scalable service-Oriented MiddlewarE over IP* (SOME/IP) [7] – standardized by AUTOSAR – is the most widely deployed middleware tailored to the automotive environment and implements service-oriented communication via IP and Automotive Ethernet [8]. Paired with Time-Sensitive Networking (TSN) [9], Automotive Ethernet can meet real-time requirements. In this architecture, services are envisioned to be dynamically updated and orchestrated on the vehicle ECUs [10]. Therefor SOME/IP provides a complementary Service Discovery (SD) [11] that detects service availability and establishes sessions between producers and consumers. SOME/IP, however, does not verify the authenticity of service providers.

The problem of securing SD is not unique to the automotive domain. On the Internet, the endpoints of services are determined with the help of the Domain Name System (DNS). Its Domain Name System Security Extensions (DNSSEC) [12] ensure data integrity and authenticity of the DNS records. In addition, DNS-Based Authentication of Named Entities (DANE) [13] binds public certificates to names to ensure the authenticity of the connection endpoint unambiguously and without attestation collisions. DNSSEC and DANE are well-established and widely deployed Internet standards with almost eight million DNSSEC verified zones and over half a million DANE enabled zones on the Internet[1].

In this paper, we solve the problem of service authenticity and certificate management in vehicles by making use of the established Internet standards DNSSEC and DANE. We use the SOME/IP SD as a showcase, although our approach could be transferred to other in-vehicle protocols. Unlike earlier proposals, which manually pre-provisioned certificates for adding authentication during session establishment [14], our solution manages security credentials dynamically and with fully functional update options. Our solution defines a DNS namespace, which complies with SOME/IP service descriptions. We store parameters of SOME/IP service endpoints in the DNS and bind certificates to their names using DANE. DNSSEC ensures authenticity and integrity of the records, while the DNS organizes distribution, caching, and updates. In addition, we provide a challenge-response mechanism to verify the authenticity of the publisher endpoint. We extend the SOME/IP reference implementation to access a local DNS resolver for service parameters and certificates during the SD. We compare the performance of our scheme to the reference implementation.

The remainder of this paper is structured as follows. Section II provides an overview of the SOME/IP SD and related work on secure discovery of services. Section III presents our concept of DNSSEC-based SD for publisher authenticity. We evaluate our concept in Section IV and discuss performance results. We conclude in Section V with an outlook.

II. In-Car Service Security and Related Work

Modern cars have a wide range of heterogeneous services as analyzed in our previous work [15]. Among them are Advanced Driver Assistance Systems (ADAS), which improve road safety and driving experience, and multimedia applications for infotainment. Traditionally, Electrical/Electronic (E/E) architectures are rigidly integrated at design time and

[1] SecSpider Global DNSSEC deployment tracking [Online]. Available: https://secspider.net/stats.html (Accessed 28.11.2022)

tightly couple software components to their ECUs. As the number of services increases, E/E architectures become more complex. Orchestrating software applications across hardware resources in a dynamic SOA allows for a more flexible software architecture [10]. This enables shorter innovation cycles, frequent updates, and on-demand installation of services.

Current vehicles are vulnerable to networked attacks via various interfaces including V2X communication [2], [3]. In an unprotected network of services, a malicious participant can compromise the communication across the entire network. This could disrupt the function of safety-related services in automotive networks.

Current automotive systems and protocols were often designed for closed environments [6] and lack a robust security layer. The AUTOSAR platform [16] advises two major SOA solutions for the automotive domain, SOME/IP and Data Distribution Service (DDS) [17]. DDS supports basic service authenticity [18], while SOME/IP, the most widely deployed protocol in the automotive domain, is tailored to a closely protected automotive environment. SOME/IP SD lacks security means [14] including data confidentiality, protection against replay attacks, service authorization and authentication. We focus on securing SOME/IP through service authentication using the established Internet standards DNSSEC and DANE combined with common authenticity standards.

A. Common Standards for Authentication

Service authentication mechanisms generate trust by attesting the identity of a service provider. In certificate-based service authentication based on the X.509 Public Key Infrastructure (PKI) standard [19] asymmetric cryptography and a trust anchor are used. The certificate, which contains the public key, proves the identity of an entity e.g., an application or a service. The certificate also contains a signed reference to the trusted entity. A client application can now request this certificate and verify its authenticity using the public key of the trusted instance. Subsequently, the client verifies the endpoint authenticity of an entity via a challenge-response protocol and get proof that the entity possesses the private key.

The public Certification Authority (CA) model uses the X.509 PKI standard to attest certificate authenticity of Internet applications. The Transport Layer Security (TLS) handshake protocol [20] for example can be used to verify the endpoint authenticity of entities. The main problem of the public CA model is that any trusted CA can issue a certificate for any domain name [13]. Multiple signing CAs can generate attestation collisions.

DNSSEC is a well-established infrastructure to secure the DNS against unauthorized modifications of its records. It uses asymmetric cryptography and additional signature records to ensure integrity of the plain text DNS records and to attest authenticity of the data stored in the DNS. Similar to the public CA model in the Web PKI, DNSSEC uses asymmetric cryptography to establish a chain of trust from the root zone to any delegated zone. This chain of trust is built along the name hierarchy, though, and remains resistant against attestation collisions, which the Web PKI generates if multiple CAs sign the same resource name.

A robust and mature ecosystem developed during more than 15 years of DNSSEC deployment. This includes not only software and tooling but also a professional practice and thorough analyses of credential maintenance [21] including the roll-over of the DNSSEC root keys [22]. It is noteworthy that DNSSEC can also be deployed for private namespace management independent of the global Internet naming hierarchy.

DANE enables the binding of certificates to names in the DNS. It uses TLSA records to store certificate data tied to domain names. The certificate presented by a server must then match against the certificate associated with DNS data to determine its integrity and authenticity. The security of DANE is tied to DNSSEC, which assures the integrity and authenticity of the TLSA records. As such DANE benefits from the inherent chain of trust of DNSSEC that ensures data integrity and authenticity.

In this work, we modify the *vsomeip* stack to query DNSSEC-verified service parameters and connection information. For this, we store the service parameters and connection information in the DNS. Further, we deploy certificates using TLSA records to bind them to services. We sign all records to obtain signature records and achieve authenticity and integrity of the service parameters and certificates. The benefit from this approach is a collision-free publisher authenticity that is protected by the well-established DNS infrastructure.

B. DNS Service Records

There are different DNS record types for storing data in the DNS specified by the IETF. We discuss two record types which are the most relevant for our approach.

The SRV record [23] specifies the location of service endpoints. It stores the port number and the domain name of a service in its data fields. One or more address records are required to query the IP address via the domain name. The name of SRV records is intended to be chosen according to the attribute leaf naming pattern [24]. The attribute leaf naming pattern enables semantic scoping for services under a parent domain. It uses underscored names prepended to the parent domain to structure records (e.g., _ldap._tcp.example.com).

The SVCB record [25] is a general purpose service binding record and is still an active IETF Internet-Draft in the converging phase to become a standard. Among the SVCB data are fields for port number, IP address and 255 further fields for private use to store service parameters.

C. SOME/IP Service Discovery

SOME/IP is widely used in automotive networks and is capable of communicating via UDP and TCP transport. Its design goals include scalability and low resource consumption. SOME/IP uses a publish-subscribe model. Publishers can notify subscribers about an update or an event that has occurred. SOME/IP SD announces and discovers services via multicast. It also performs the session establishment between

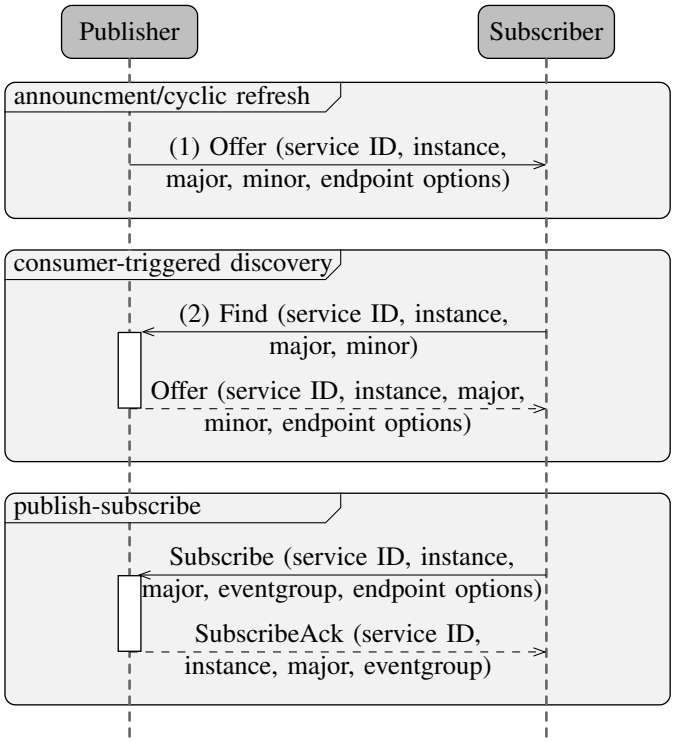

Fig. 1: Service announcement, discovery, and subscription according to the SOME/IP service discovery protocol.

publishers and subscribers after a successful subscription. SOME/IP provides no means for service authenticity.

Figure 1 shows the SOME/IP SD sequence for service announcements, discovery, and subscription. The SOME/IP SD uses multicast Find and Offer messages to request and announce services, which are described by their ID, instance ID, major and minor version. An Offer entry uses so-called endpoint options to describe how to contact a service. There are two concepts of discovering a service. (1) A publisher periodically updates an Offer message, as an Offer has a limited lifetime. (2) A subscriber requests a service via a Find message, whereupon corresponding publisher instances announce it via an Offer message. After the subscriber receives the Offer message, it subscribes to the service via unicast specifying its receiving endpoint description and the desired Eventgroup. If the publisher can provide this service, it acknowledges the subscription, after which the transmission of the requested data begins.

D. In-Vehicle Service Authenticity and Confidentiality

Secure discovery mechanisms are essential to prevent attackers from participating in automotive networks and eavesdropping on in-vehicle communication. Challenge-response schemes can authenticate nodes to control service access. Message encryption keeps unauthorized participants from eavesdropping on network communication. In this work, we focus on publisher authenticity using a challenge-response scheme based on the public credentials obtained from DNSSEC.

Challenge-response mechanisms require cryptographic keys that are commonly pre-deployed on the vehicle ECUs and can be both symmetric secret keys [26] or asymmetric key pairs [27]. Further, a PKI or public CA model uses a trust anchor to enable the authenticity and integrity of certificates with keys that can be revoked when they are no longer secure [28], [29]. In this work, we use asymmetric cryptography for a challenge-response mechanism and DNSSEC with its inherent chain of trust to ensure certificate authenticity.

Iorio et al. propose a message encryption and service authentication mechanisms for securing SOME/IP [14]. Their approach follows both ideas, the public CA model and a challenge-response scheme using asymmetric cryptography. Each vehicle has a different trusted root certificate. Each application has a private key and a public certificate signed with the private key of the trusted root certificate creating a simplified chain of trust. They also bind access control policies to the signed certificates. All required keys and certificates are pre-deployed on every ECU accordingly, which generates a huge challenge of credential management in practice. In contrast to their approach, we use DNSSEC and DANE for ensuring certificate authenticity, and the DNS recursive resolver infrastructure for managing certificate provisioning.

III. DNSSEC in SOME/IP Service Discovery

Our approach transforms the SOME/IP SD to use the established internet technologies DNSSEC and DANE to achieve secure service discovery and authenticity. Therefor we first map the data fields from the SOME/IP SD to DNS names and record data. Second, we bind DANE certificates to the service names to verify the authenticity of the service. For our prototype implementation, we adapt the *vsomeip* reference implementation so our architecture and naming depends on it.

A. Designing a DNS Namespace for SOME/IP Services

The main challenge in designing a suitable DNS namespace is to preserve all SOME/IP SD query properties. Figure 2 shows the data fields used in Find messages. Four fields specify a service: service ID, instance ID, major version, and minor version. A subscriber must specify at least a service ID in a Find message, while any of the other fields can be wildcarded. For example, if a subscriber does not specify an instance ID, they are offered all running instances of a service. So, in total, a service can be requested in 2^3 ways.

Table I shows the DNS entries for one service, using symbolic names for simplicity. We build our names based on the SOME/IP find parameters. As the parent domain we use *service*, which could be adapted, for example, to a specific OEM or tier-X supplier. More specifically, adding *tier-x.oem* as the parent domain would enable a hierarchy that passes down the rights to maintain and certify service records in each subdomain. Next, we prepend the four data fields in the same order as in the Find message service description. An unspecified field in a Find message corresponds to the absence of that field in the query name. The arrangement of the four fields is arbitrary, but must be followed consistently

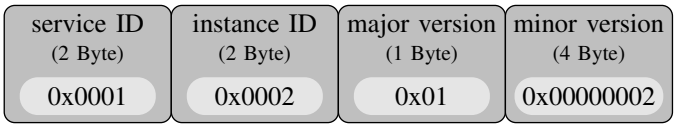

Fig. 2: Service description in Find and Offer messages.

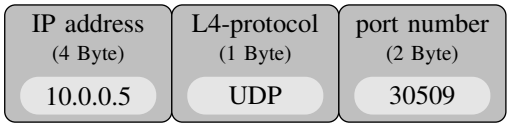

Fig. 3: Endpoint description in Offer and Subscribe messages.

as they determine valid query names. We prepend *_someip* to each branch, following the semantic scope of the attribute leaf name pattern.

We consider the two query names that specify a minor version without a major version (marked in gray) to be invalid because they do not seem practical to us. Even though, the SOME/IP SD specification does not object to such a query. This reduces the number of valid query names per service to six.

To make records uniquely distinguishable, we keep the symbolic name before the value of the data fields. Otherwise, the query names could become ambiguous when different fields are wildcarded. According to our namespace design, a service with ID 1, instance ID 2, major version 1 and minor version 2 has the following concrete query name:

_someip.minor0x00000002.major0x01.instance0x0002.
id0x0001.service.

B. Choosing a Record Type for SOME/IP Endpoints

For the interoperability between the SOME/IP SD and DNSSEC we identify a suitable DNS record that can hold all information originally provided in Offer messages. In addition to the service description (see Figure 2), Offer messages contain an endpoint option as shown in Figure 3. The endpoint option describes how to connect to a service with an IP address, L4-protocol and a port number.

The two main candidates for DNS service records are the SVCB and SRV records. With SRV records, the transport protocol of a service is integrated into the name. This does not comply with the Find message of the SOME/IP SD, since the transport protocol is not specified in Find messages. Offer messages can include endpoints for multiple transport protocols, e.g., TCP and UDP. Next, the SRV record does not provide a data field for the IP address of a service, but holds a domain name. These two restrictions require additional queries as a detour to obtain address records and available transport protocols. We decide for the SVCB record because it provides data fields for the IP address, port number, an application layer protocol, and 255 more fields for private use.

TABLE I: SVCB records for one service with symbolic query names and concrete record data.

QNAME	RDATA (SVCB)
_someip.minor.major.instance.id.service.	port=30509
_someip.major.instance.id.service.	ipv4hint=10.0.0.5
_someip.minor.instance.id.service.	protocol=UDP
_someip.minor.major.id.service.	instance=2
_someip.instance.id.service.	major=1
_someip.major.id.service.	minor=2
_someip.minor.id.service.	
_someip.id.service.	

Table I shows SVCB records for one service. All query names have the same record data. The data fields of the Offer messages, including the endpoint options, are mapped as follows. The port and ipv4hint fields of the SVCB record are used as intended. The instance ID, major version, minor version and layer 4 protocol are each mapped to one of the 255 fields for private use. Thus, the record data in Table I refers to a service which is accessible via UDP at the IP address 10.0.0.5 and port 30509. Further, the record holds the instance ID 2, major version 1, and minor version 2 in case it was wildcarded in the query name.

C. Augmenting the SOME/IP SD for using DNSSEC

DNSSEC ensures that the records are unchanged and correct when the subscriber receives them. This is already an advantage over the SOME/IP SD, where anyone can send conflicting offers. The SOME/IP stack needs to be adapted for using DNSSEC to discover services. We showcase our approach based on the open source reference implementation *vsomeip* [30].

Figure 4 shows the conceptual architecture inherited from *vsomeip*. Both the client and the server use the same stack, which comprises of an application, a routing manager, and the service discovery. The routing manager handles the local transport-specific endpoints for the applications and forwards sent and received messages between them.

The modifications we made to use DNSSEC during SOME/IP SD are also shown in Figure 4. Instead of the original Offer/Find procedure, the client retrieves the endpoint description of the publisher service via a DNSSEC resolver. With that, publisher services no longer announce themselves, and we gain secure service discovery through the implicit trust established by DNSSEC records. After the client obtains a record for the service, it can subscribe to the service which then publishes data to the subscriber.

D. Ensuring Publisher Authenticity with DANE

With DNSSEC, the subscriber knows that the subscription parameters used to access the endpoint are correct. Nevertheless, an attacker can still mimic this endpoint, for example, using IP spoofing. We use DANE to validate the authenticity of the service and to ensure that the subscriber connects to the correct publisher.

Figure 5 shows our secure service discovery and invocation process. After the subscriber has resolved the service endpoint with a DNS query, it uses the information in the SVCB record to subscribe to the service. At the same time, it queries the

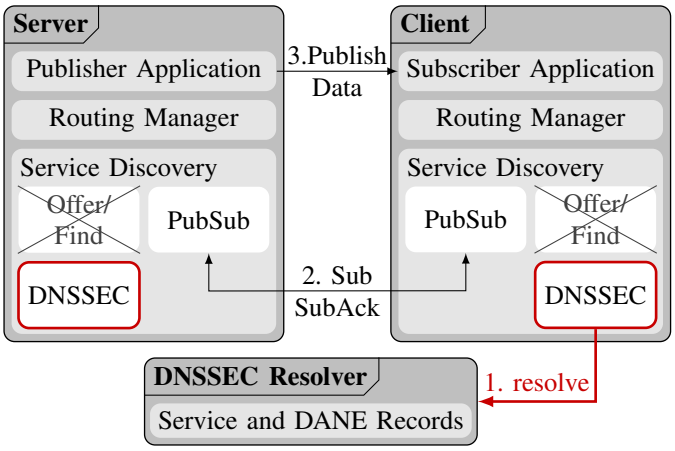

Fig. 4: SOME/IP SD modification for using DNS.

DNS for the DANE TLSA record of the service. The TLSA record contains the public certificate of the service, which is again protected with DNSSEC. With this the subscriber can validate the signature of the publisher.

We use a challenge-response scheme to ensure that the publisher endpoint is authentic and indeed the owner of the corresponding private key. With the Subscribe message, the subscriber sends a random nonce as a challenge to the publisher. For this, we use a SOME/IP configuration option containing a 32-bit nonce. The publisher service signs the challenge with its private key and then sends the subscription acknowledgement with the signed random nonce back to the subscriber, again using a configuration option. The subscriber can then verify the signature with the public certificate of the publisher. If the signature is valid, the subscriber can be sure that the publisher is authentic. With a future extension, the subscriber and publisher could agree on a session key during the challenge-response process to enable confidentiality by encrypting messages.

E. Operating DNS-based Automotive Service Discovery

In operation, we foresee that a car has a local DNSSEC recursive resolver that caches verified records as soon as the car has Internet connectivity. Each time a record is retrieved, the DNSSEC recursive resolver has to ensure the chain of trust before it is cached. This ensures that the service discovery is still operational when the vehicle is disconnected from the Internet. In addition, this has the advantage that no DNSSEC validation needs to be executed during the SD, the presence of the (validated) records suffices. Cached records are refreshed before they expire, and it shall be part of future experimentally driven research to work out appropriate cache lifetimes in real deployments. In this way, our approach exploits the benefits of a well-established standard infrastructure for obtaining data integrity, authenticity, and a robust procedure for certificate management.

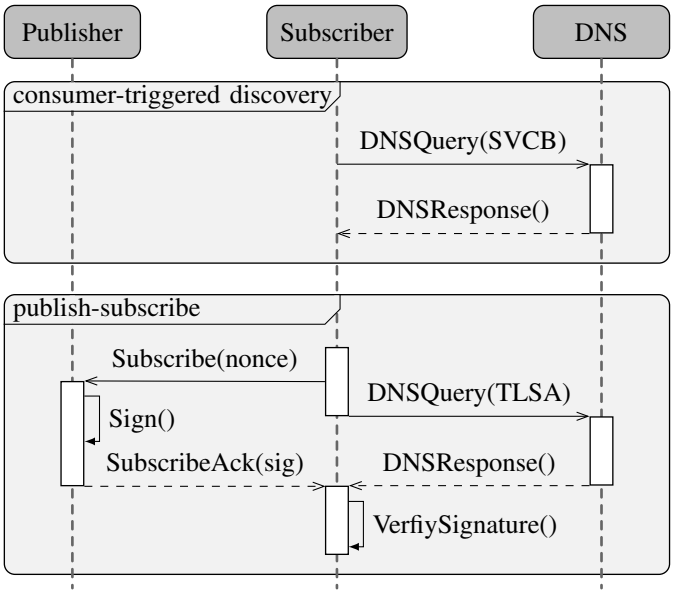

Fig. 5: Augmented SOME/IP SD with DNSSEC and DANE for secure publisher service discovery and authentication.

IV. Discovery Capabilities and Performance

We evaluate the performance of the proposed solution compared to the unchanged SOME/IP SD protocol. Therefore, we first compare the service discovery capabilities showing differences in communication schemes and security mechanisms. Then, we evaluate the performance of our prototype implementation in terms of discovery and subscription latency.

A. SOME/IP SD vs. DNSSEC and DANE

Table II summarizes differences in key features between the proposed approach using DNSSEC and DANE, and the SOME/IP SD protocol. SOME/IP was initially released in 2016 as a module in the AUTOSAR platform and targets local in-vehicle networks. DNSSEC and DANE are defined in RFCs by the IETF. Our approach with DNSSEC and DANE leverages this technology with over 15 years of global deployment and operational experience on the Internet. With this we gain the benefits of a tried, resilient and security hardened infrastructure.

The SOME/IP SD uses group communication, whereas the DNS protocol uses unicasts. For DNS-based discovery, this implies that multiple clients of the same server must all query the DNS resolver separately, while with SOME/IP SD, a publisher can inform subscribers with a single multicast Offer, reducing the network load. An evaluation in a realistic automotive setup with a large number of services would show whether our approach introduces significant performance penalties, but we leave that open for future work.

The discovery of the endpoint is done by the SOME/IP SD with Offer messages initiated by the publisher, while the DNS resolver is directly queried by the consumer. This has several implications. First, with the SOME/IP SD the publisher can provide its endpoint information during runtime, but the

TABLE II: Feature comparison between the SOME/IP SD protocol and the proposed approach based on DNSSEC and DANE.

Feature	SOME/IP SD	DNSSEC and DANE
Standard commission	AUTOSAR [7], [11]	IETF [12], [13], [25]
Introduction and deployment	Basic support in AUTOSAR since Nov. 2016, deployment in production vehicles just starting	DNSSEC first standardized in 1997, over 15 years of global deployment and operational experience
Target environment	Developed for local in-vehicle network	Hardened for global Internet deployment
Service discovery scheme	Multicast Find/Offer messages	Unicast DNS Query/Response
Endpoint information distribution	Provided initiated Offer messages with the service runtime location	Consumer requested DNSSEC-signed SVCB records with pre-defined endpoint information
Authentication scheme	None by default (security extension with challenge-response proposed in [14])	Challenge-response
Server certificate distribution	Pre-deployed on server and client nodes	Consumer requested DNSSEC-signed TLSA records
Server private key distribution	Deployed with application	Deployed with application
Certificate update procedure	Simultaneous certificate update for server and client applications in a car	Update server app, add DNS entries for new version

DNS resolver is not aware of the service runtime location. This requires predefined IP addresses and ports for all service instances in DNS records, which should be the same for all vehicles. However, this is not a problem within a local network where the IP addresses can be freely selected. On the other hand, the DNS records can be verified along the DNSSEC trust chain, which is not possible for the endpoint information provided by the publisher. This also prevents malicious services to Offer false endpoint information, for which there is no protection with SOME/IP SD.

There are no service authentication means in SOME/IP SD by default. Iorio et al. [14] propose a solution to secure the SOME/IP SD using the public CA model to ensure the authenticity of the certificate and the TLS handshake to ensure the authenticity of the endpoint. They pre-deploy public certificates over every node to avoid additional lookups in an external repository. Our approach exploits the DNSSEC and DANE mechanisms to ensure implicit certificate and service information authenticity through the DNSSEC trust chain, and uses a challenge-response mechanism similar to the TLS handshake ensuring endpoint authenticity. With this, we have to perform additional lookups in the DNS to retrieve the certificate and the service information. We evaluate the discovery latency in the following benchmark and show that the certificate lookups do not introduce a significant overhead.

In case of certificate changes that also affect the keys, the private key in question must be updated, both for pre-deployed certificates with SOME/IP and for certificates in TLSA records. If the keys are pre-deployed, they must be updated on each client and server node. With our approach new records can be added to the DNS for the new version of the app and the keys on the publisher node can be updated with a new version of the application. This ensures that older versions can still use the old certificates while the already updated applications can use the new certificates.

B. Evaluation Setup

We measure the service discovery and subscription latency and the cost of the cryptographic operations. In doing so, we compare four different solutions, all of which are implemented based on the *vsomeip* [30] stack:

1) **SOME/IP SD:** An unaltered *vsomeip* implementation that we use as a baseline.
2) **DNSSEC**: A DNSSEC augmented *vsomeip* that replaces the original Offer/Find procedure with our DNSSEC-based, consumer-triggered discovery.
3) **DNSSEC w/ DANE** An authentication approach based on the DNSSEC discovery implementation that uses DANE records to retrieve the publisher certificate, and our challenge-response mechanism during the subscription phase to authenticate the publisher.
4) **SOME/IP SD w/ AUTH**: An authenticated approach returning to the original SOME/IP SD without any DNS operations that uses pre-deployed certificates and our challenge-response mechanism to authenticate the publisher, similar to the proposed solution in [14].

Our evaluation setup consists of three nodes for a client, a server and a DNSSEC resolver, which are arranged as shown in Figure 4. All nodes run on the same host system (CPU: AMD FX-8350 with 8 cores at 4Ghz, RAM: 16GB) in separate

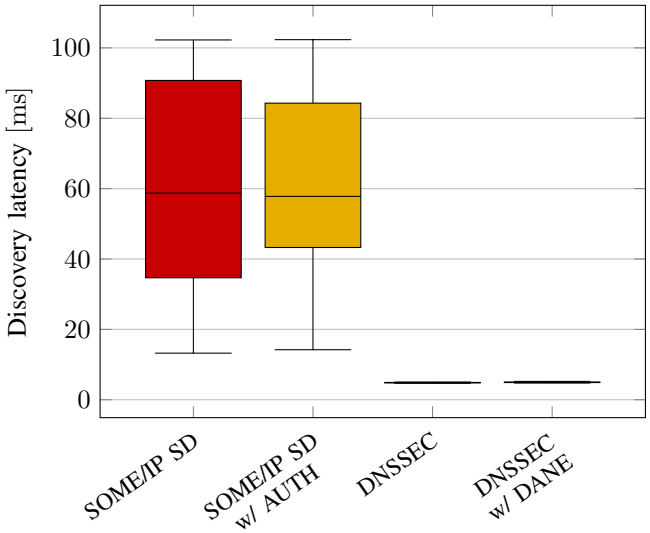

Fig. 6: Whisker plot on the latency of service discovery.

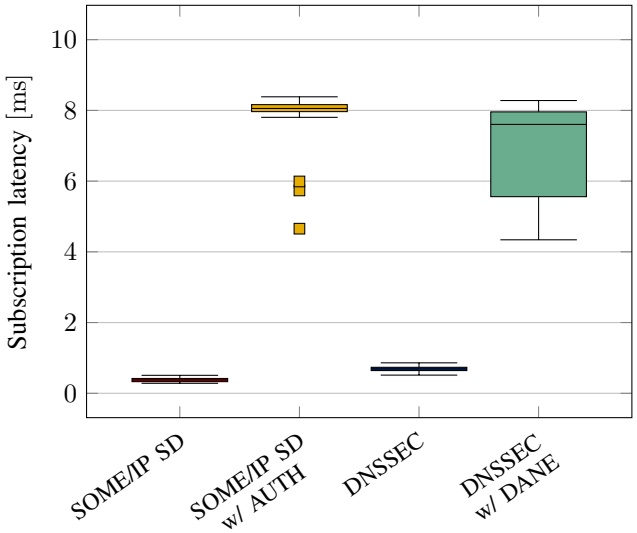

Fig. 7: Whisker plot on the latency of service subscriptions. The small squares are outliers.

containers (*Docker*: 20.10.22) connected via the Docker virtual bridge network.

The server and client containers run on a Linux OS with the SOME/IP stack, and libraries for DNS lookups and cryptography (*Ubuntu*: 18.04.6 LTS, *vsomeip* [30]: 3.1.20.3, *Crypto++* [31]: 8.7.0, *Crypto++ PEM Pack*: 8.2). The DNSSEC resolver runs on a Linux OS (*Ubuntu*: 22.04.1 LTS, *Unbound*: 1.17.0).

The DNS entries for the SVCB and TLSA records of the publisher service are already in the cache of the DNSSEC resolver, as would be the case in an automotive deployment.

As SOME/IP uses group communication for service discovery, it applies common practices for scattering multicast communication to reduce the load on the network and hosts. For example, responses can be delayed collecting multiple requests and answer them in a single response. Since we compare it to standard DNS discovery via unicast queries, which does not include any of such delays, we turn off the request-response delay in *vsomeip* to get comparable results. Moreover, we only look at the connection of one server and client, for which these mechanisms are not needed. The startup phases of SOME/IP SD, however, remain unchanged, and thus a random initial delay between $10\,\mathrm{ms}$ and $100\,\mathrm{ms}$ delays the startup of the discovery phase.

C. Discovery and Subscription Latency Benchmark

Our benchmark evaluates the latencies of the discovery, subscription, and cryptographic operations. For each of the four compared solutions, we collect fifty samples with timestamps indicating the beginning and end of different phases to calculate the latency based on the difference between these timestamps. Figure 1 and Figure 5 show the sequences of the consumer-triggered discovery and publish-subscribe phases for the SOME/IP SD and the DNS discovery, respectively.

Figure 6 shows the consumer-triggered discovery latency of all four different solutions. Here, we measure the time that elapses from the completion of the initialization of the client until the result of the service discovery is available. Since the measured interval for the discovery does not include authentication operations, the latency of the solutions with publisher authentication are expected to be the same as without publisher authentication. The DNS discovery latencies are between $4\,\mathrm{ms}$ and $6\,\mathrm{ms}$. Both SOME/IP SD variants have a latency between $13\,\mathrm{ms}$ and $103\,\mathrm{ms}$ due to the random initial delay between $10\,\mathrm{ms}$ and $100\,\mathrm{ms}$. Without an initial delay, the latency of the SOME/IP SD would be similar to that of the DNS discovery.

Figure 7 shows the subscription latency of the four candidates. We measure the time that elapses between the sending of the first subscription message and the completion of the connection setup, including the verification of the publisher signature in the authenticated approaches. The solutions without publisher authentication have a latency under $1\,\mathrm{ms}$. With publisher authentication the latency is between $4\,\mathrm{ms}$ and $9\,\mathrm{ms}$. In detail, signing the nonce at the publisher takes between $3\,\mathrm{ms}$ and $7\,\mathrm{ms}$, verifying the signature at the client side is below $2\,\mathrm{ms}$. The trade-off in using our challenge response scheme results in a maximum delay of $8\,\mathrm{ms}$.

Considering the overall discovery and subscription latency the publisher authentication does not have a significant impact on the latency, for which the multicast scattering is the most notable delay. DNSSEC and DANE enable publisher authenticity without a large performance penalty even compared to authentication with pre-deployed certificates. We achieve this by querying the TLSA record at the same time as we initiate the subscription. However, the latency of the TLSA response containing the certificate depends on the link to the DNS server. This could impact the results when the DNS query takes longer than the subscription handshake. Here, an evaluation

with an Ethernet-connected DNS server would be interesting to see the impact of the latency. In addition, the performance for a larger number of services should be analyzed to determine scalability with DNS discovery compared to SOME/IP SD in a realistic automotive network.

V. Conclusion and Outlook

In this paper, we designed and analyzed basic security elements for the rapidly evolving service-oriented software architecture in future cars. In provisioning service authentication and managing attestation credentials, we addressed the urgent demand for securing a heterogeneous, distributed, and dynamically updatable software ecosystem that will drive the connected cars of the near future.

Our work was intentionally built on well-established standards. DNSSEC and DANE enable certificate management and service authenticity while being a thoroughly validated, operationally stable Internet standard. SOME/IP is a widely accepted service-oriented middleware standardized by AUTOSAR. We demonstrated how to combine the SOME/IP SD with the Internet name system in design, implementation, and evaluation. Our findings indicated that SOME/IP SD can interact with the DNS without operational overhead, while DNSSEC with DANE contribute not only a robust, reliable security solution but also a stable infrastructure for replication, (off-line) caching, and key management.

This basic solution to automotive service security opens three future research directions. First, the remaining SOME/IP service primitives for onboard session establishment and migration need a detailed security design and assessment. Second, operational guidelines for name-space management and service updates in the automotive ecosystem shall be developed. Third, we aim at configuring a full-featured production-grade vehicle with our security solution and evaluate its properties in macroscopic benchmarks.

References

[1] S. Checkoway *et al.*, "Comprehensive Experimental Analyses of Automotive Attack Surfaces," in *Proceedings of the 20th USENIX Security Symposium*, vol. 4. USENIX Association, Aug. 2011, pp. 77–92.

[2] C. Miller and C. Valasek, "A Survey of Remote Automotive Attack Surfaces," *Black Hat USA*, vol. 2014, 2014.

[3] ——, "Remote Exploitation of an Unaltered Passenger Vehicle," *Black Hat USA*, vol. 2015, p. 91, 2015.

[4] F. Kohnhäuser *et al.*, "Ensuring the Safe and Secure Operation of Electronic Control Units in Road Vehicles," in *2019 IEEE Security and Privacy Workshops (SPW)*. IEEE, 2019, pp. 126–131.

[5] L. Xue *et al.*, "SAID: State-aware Defense Against Injection Attacks on In-vehicle Network," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1921–1938.

[6] A. Martínez-Cruz *et al.*, "Security on in-vehicle communication protocols: Issues, challenges, and future research directions," *Computer Communications*, vol. 180, pp. 1–20, 2021.

[7] AUTOSAR, "SOME/IP Protocol Specification," AUTOSAR, Tech. Rep. 696, Nov. 2021.

[8] K. Matheus and T. Königseder, *Automotive Ethernet*. Cambridge, United Kingdom: Cambridge University Press, Jan. 2015.

[9] IEEE 802.1 Working Group, "IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks," IEEE, Standard Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014), Jul. 2018.

[10] A. Kampmann *et al.*, "A Dynamic Service-Oriented Software Architecture for Highly Automated Vehicles," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. 2019, pp. 2101–2108.

[11] AUTOSAR, "SOME/IP Service Discovery Protocol Specification," AUTOSAR, Tech. Rep. 802, Nov. 2021.

[12] 3rd D. Eastlake, "Domain Name System Security Extensions," IETF, RFC 2535, March 1999.

[13] P. Hoffman and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA," IETF, RFC 6698, August 2012.

[14] M. Iorio *et al.*, "Securing SOME/IP for In-Vehicle Service Protection," *IEEE Transactions on Vehicular Technology*, vol. 69, pp. 13 450–13 466, 2020.

[15] M. Cakir *et al.*, "A QoS Aware Approach to Service-Oriented Communication in Future Automotive Networks," in *2019 IEEE Vehicular Networking Conference (VNC)*. Dec. 2019.

[16] M. Rumez *et al.*, "An Overview of Automotive Service-Oriented Architectures and Implications for Security Countermeasures," *IEEE Access*, vol. 8, pp. 221 852–221 870, 2020.

[17] Object Management Group, "Data Distribution Service," Online, OMG, Standard DDS 1.4, Mar. 2015.

[18] ——, "DDS Security," Online, OMG, Standard DDS-SECURITY 1.1, Jul. 2018.

[19] D. Cooper *et al.*, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," IETF, RFC 5280, May 2008.

[20] E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3," IETF, RFC 8446, August 2018.

[21] E. Osterweil *et al.*, "From the Beginning: Key Transitions in the First 15 Years of DNSSEC," *Transactions on Network and Service Management (TNSM)*, 2022.

[22] M. Müller *et al.*, "Roll, Roll, Roll Your Root: A Comprehensive Analysis of the First Ever DNSSEC Root KSK Rollover," in *Proceedings of the Internet Measurement Conference*, ser. IMC '19. ACM, 2019, p. 1–14.

[23] A. Gulbrandsen *et al.*, "A DNS RR for specifying the location of services (DNS SRV)," IETF, RFC 2782, February 2000.

[24] D. Crocker, "Scoped Interpretation of DNS Resource Records through 'Underscored' Naming of Attribute Leaves," IETF, RFC 8552, March 2019.

[25] B. M. Schwartz *et al.*, "Service binding and parameter specification via the DNS (DNS SVCB and HTTPS RRs)," Internet Engineering Task Force, Internet-Draft draft-ietf-dnsop-svcb-https-11, Oct. 2022, work in Progress.

[26] H. Khemissa and P. Urien, "Centralized architecture for ECU security management in connected and autonomous vehicles," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, Oct 2022.

[27] S. Fassak *et al.*, "A secure protocol for session keys establishment between ECUs in the CAN bus," in *2017 International Conference on Wireless Networks and Mobile Communications (WINCOM)*. IEEE, Nov 2017.

[28] M. A. Al-Shareeda *et al.*, "Survey of Authentication and Privacy Schemes in Vehicular ad hoc Networks," *IEEE Sensors Journal*, vol. 21, pp. 2422–2433, Jan 2021.

[29] M. Asghar *et al.*, "A Scalable and Efficient PKI Based Authentication Protocol for VANETs," in *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. IEEE, Nov 2018.

[30] BMW AG, "vsomeip v3.1.20.3," GitHub repository. [Online]. Available: https://github.com/COVESA/vsomeip

[31] W. Dai, "Crypto++ library 8.7,". Website. [Online]. Available: https://www.cryptopp.com/