Applications of Pairing-based Cryptography on Automotive-grade Microcontrollers

Tudor Andreica, Bogdan Groza and Pal-Stefan Murvay

Faculty of Automatics and Computers,
Politehnica University of Timisoara, Romania
Email: andreica_tudor@yahoo.com, {bogdan.groza, pal-stefan.murvay}@aut.upt.ro

Abstract. Bilinear pairings have been successfully used both in cryptanalysis and in the design of new cryptographic primitives, e.g., identity based encryptions and signatures. In this work we discuss some computational results for pairing-based libraries on two automotive-grade controllers as well as on an Android smart-phone. This is relevant as the computational resources of automotive-grade controllers have surged in the recent years and Android-based units are quickly entering the car market, e.g., infotainment units. Identity-based primitives open road for representative applications in automotive-based scenarios where the identity of the vehicle or of the OEM may be used for deriving public keys without relying on PKI. We discuss three potential use-cases: the security of in-vehicle buses, vehicle-to-vehicle communications and software updates that could greatly benefit from compact signatures, identity based encryption or signing as well as from group signatures.

1 Introduction and Motivation

With the myriad of attacks recently reported over in-vehicle networks [14], [7], [16] the standardization of cryptographic functions for in-vehicle security was an immediate development. Current standards, e.g., AUTOSAR [1], [2], include specifications for regular cryptographic functions both from the symmetric and asymmetric setting. But more complex mathematical operations such as pairings open door to more advanced cryptographic primitives: identity based signatures, identity based encryptions and group signatures. Further, these may be used in automotive-based scenarios to solve security issues in a more convenient way. The current work is motivated by three practical scenarios that may greatly benefit from the application of pairing based cryptographic techniques: securing in-vehicle buses, vehicle-to-vehicle communication and software updates. We now briefly discuss the advantages for each scenario and in the forthcoming sections we make performance evaluations on two automotive-grade controllers.

In-vehicle buses provide the first applicative scenario. This scenario is suggested in Figure 1 which is also representative for the setting of the work. Several ECUs (Electronic Control Units) are depicted as well as a central display (or possibly an infotainment unit) and a smart-phone, all of them connected to the CAN bus (Controller Area Network). These units are responsible with functionalities related to engine, chassis, telematics or the body (i.e., the Body Control Module - BCM) but nonetheless with user entertainment. Due to inherent connectivity to the outside, the infotainment unit

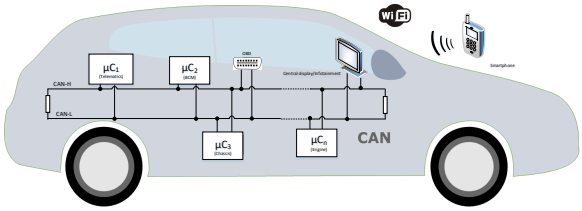

Fig. 1. Application setting, security of in-vehicle networks

represents a realistic attack surface which was previously exploited [7], [16] and since this unit is connected to the CAN bus it opens road to attacks on all other ECUs on the network. Consequently, the security of in-vehicle buses has been a constant preoccupation in the past few years and many solutions have been proposed, a survey can be found in [11]. Pairing based cryptographic primitives do not seem a first choice for this scenario due to their high computation requirements. However, a significant constraint of this scenario is the limited size of the data packets that are sent via the in-vehicle bus. Traditionally, the CAN bus was capable of sending only 64 bits in a single packet which is too small even for symmetric cryptographic functions. This is improved with the recent release of CAN-FD (CAN with Flexible Data-Rate) that extends the packet to 512 bits and newer in-vehicle buses such as FlexRay extend this a bit further. But the packet size is still limited and needs to accommodate both signals from the ECUs as well as security elements (signatures or message authentication codes). This limitation of in-vehicle data packets may favour the use of compact signatures that can be built on pairings such as the Boneh-Lynn-Shacham signature (BLS) [6]. While computational demands for this kind of signatures are somewhat high for automotive grade controllers, compact signatures may still replace conventional signatures when bandwidth is more critical than computational cost.

Vehicle-to-vehicle (V2V) communication is a quickly emerging technology. It is vital both for traffic optimizations and for preventing road accidents. The main security concern in deploying this technology stems from the fact that security mechanisms must be in place in order to validate the source of the broadcast messages. Recent documents from the US Department of Transportation already point out on the necessity of PKI [12] for assuring security in V2V communications. The problem of using PKI comes from the fact that each certificate needs to be signed by a certificate authority (CA) and must be made available to other traffic participants. Nonetheless, once a participant obtains a genuine certificate there is nothing that can stop him to broadcast false messages regarding traffic conditions. We discuss why identity-based cryptography may help.

First, identity-based cryptographic techniques may eliminate the need for a CA that signs the certificates. If a vehicle is directly identifiable by its license plate, then this can serve as the ID from which the public key of the vehicle derives. Subsequently, a secure communication tunnel between two vehicles or an authenticated broadcast channel between one-vehicle and other traffic participants can be easily established. Vehicle-to-vehicle communication can take advantage of ubiquitous communication channels such as Bluetooh, WiFi or its extension for ad-hoc networking WiFi Direct, which allows the creation of ad-hoc networks over a short range. The advantage of these technologies

is that they may be available even where 3G connectivity or road-side units (RSU) are absent. The use of WiFi Direct has been already proposed in [9] for a pedestrian warning system and the use of Multipeer technology (an ad-hoc networking technology based on WiFi and Bluetooth from Apple) was suggested in [10]. Second, identity-based cryptography may partly help in discovering false reports from traffic. While in case of conventional cryptographic techniques each user will have a public-key that is random (and unavailable as visual information to other participants), the use of the license plate number as source for the public-key may allow other traffic participants to visually validate that the car that sends reports from the traffic is present at the location. Of course, in case of conventional PKI one can move the license plate number inside the public-key certificate and obtain a similar result, but identity-based cryptography offers the more intuitive and direct way to achieve this goal by linking a visual/physical clue to the public-key of the principal. Identity based cryptography and pairing in particular have been previously proposed for vehicle-to-vehicle communication [21], [19] but practical evaluations on automotive-grade cores are still needed.

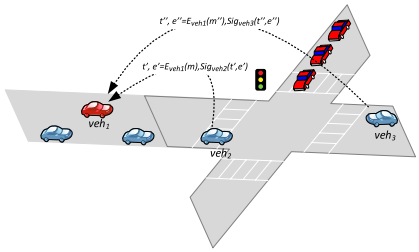

Fig. 2. Vehicle-to-vehicle communication scenario

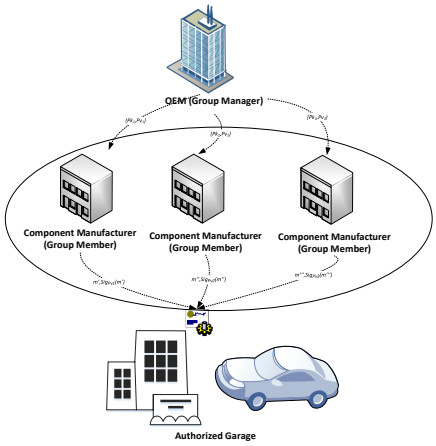

Fig. 3. Software update scenario

In Figure 2 we suggest a collision avoidance scenario. Here, the red vehicle veh_1, from the left side of the figure, is overtaking the other three blue vehicles and is dangerously approaching a potential red light in the middle of the intersection. Consequently the other vehicles veh_2 and veh_3 are signalling him by sending warning messages. The messages are secured by using the identity of the vehicle as public key for encryption and the identity of the other vehicles as private key for signing. Additionally, a timestamp is used to avoid replay attacks. Each participant retrieves the identities of the other cars from the license plates which offer an additional (visual) channel that demonstrates that the vehicles are present in traffic. Cameras are cheap and can be located both in the front and/or read of the vehicle. In fact, rear cameras come as a default for most vehicles nowadays. Thus, an identity-based secure channel can be established between

vehicles whose license plates are in sight of each other. If a vehicle is not in sight, the identity-based channel can still be established by using identities from the broadcast messages without the additional evidence of physical presence from the visual channel which may occur later in time.

Software updates are a constant preoccupation of the industry since long ago. Recent research works are increasingly concerned with performance evaluation of software updates that are performed over the air [17] and the adoption of new technologies such as the Blockchain which has been recently proposed for this purpose [18]. We envision that the role of pairings in software updates may stem from the use of group signatures, a construction which immediately derives from pairings. In this type of signatures, the signature may originate from one or more principals inside a group. The advantage in the automotive scenario is that a piece of software can be signed by one or more parties that have the credentials. This may be of interest since vehicle components and software generally originates from multiple manufacturers. This scenario is suggested in Figure 3 were a vehicle receives a software update from a garage that received the signed software updated from one of several component manufacturers. Prior to this, the component manufacturers received their public-private key pairs from the OEM (Original Equipment Manufacturer) which plays the role of Group manager. The OEM may individually verify from whom the software update originates, but this is transparent for the authorized garaged.

2 Cryptographic Libraries and Platforms

We proceed with a brief background on pairing-based cryptography, then we describe the platforms of our experiments and proceed to the cryptographic libraries that are used in our evaluation.

2.1 Cryptographic Functions

A bilinear pairing is a function $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, where $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ are three cyclic groups the later being called the target group. The function has three properties which in case when $\mathbb{G}_1 \times \mathbb{G}_2$ are multiplicative groups can be described as follows : i) it is bilinear by which $e(g^a, h^b) = e(g, h)^{ab}, \forall g \in \mathbb{G}_1, h \in \mathbb{G}_2, a, b \in \mathbb{Z}$, ii) non-degenerate which means $\exists g \in \mathbb{G}_1, h \in \mathbb{G}_2$ such that $e(g, h) \neq 1$ and iii) $\forall g \in \mathbb{G}_1, h \in \mathbb{G}_2$ the function $e(g, h)$ is efficiently computable.

While bilinear maps where initially used in cryptanalysis [15] they were later successfully exploited to build several cryptographic primitives. Identity based encryption functions were introduced by Boneh et al. in [5] and [3], the later being implemented in [20]. A compact signature scheme was introduced by Boneh, Lynn and Sacham in [6], the signature has a size of only 160 bits while retaining a security level equivalent to the 320-bit DSA. Group signatures based on bilinear pairings are described by Boneh et al. in [6] and Hwang et al. in [13], both schemes are implemented in [20]. This is only a brief enumeration of existing proposals that are relevant to the context of our work, many other schemes exist.

Without entering too much mathematical details, the power of bilinear transforms can be easily illustrated on the compact BLS signature [6]. We now assume that groups are in additive form which is usual when moving to elliptic curves. The holder of the public key $pk = xP$ (here P is a point on $\mathbb{G}_2$) hashes the message M that he wants to sign to a point P_M on $\mathbb{G}_1$ and signs it as $S_M = xP_M$. To verify that a signature is correct, the public-key is used to compute verifier $v = e(pk, P_M)$ which is checked against $u = e(P, S_M)$. Of course, there are intricate technical details behind this operation, e.g., how to map a message to a point of the elliptic curve, how the Weil pairing is computed, etc., but for illustrative purposes the previous description should be sufficient as efficient implementations for pairing-based cryptographic functions already exist in practice.

2.2 Experimental Platforms

For our experimental work we selected several platforms which are summarised in Table 1 and the experimental setup is illustrated in Figure 4.

The SAM V71 Xplained Ultra evaluation kit was chosen because it incorporates a high-end automotive-grade core. Moreover, the SAM V71 family is recommended by the manufacturer for the evaluation of SAM V70, SAM S70, SAM E70, as V71 is a superset of the previous. According to the producer's website [1], the V ARM-based microcontrollers are used in the automotive industry with focus on in-vehicle infotainment connectivity, telematic control units or head units. The evaluation kit contains the ATSAMV71Q21 microcontroller which is a 32-bit ARM Cortex-M7 processor. The maximum operation speed is 300 MHz, the Flash size is up to 2048 Kbytes and the SRAM up to 384 Kbytes. The microcontroller has several cryptographic features, e.g., a true random number generator (TRNG), hardware support for hash functions SHA1, SHA224, SHA256 and AES encryption with 128, 192 and 256-bit keys. To avoid the use of the debugger during run-time measurements, to evaluate the computational performance of this platform we chose to send the results after the test ended through the serial interface USART (Universal Synchronous/Asynchronous Receiver/Transmitter) to a notebook.

Table 1. Platforms evaluated in our work

Platform	Core	Flash size	RAM size	Frequency	Manufacturer
ATSAMV71	Cortex-M7	2MB	384KB	300MHz	Microchip
TC297	TriCore 1.6P	8MB	728KB	300MHz	Infineon
HTC One M7	Krait 300	n/a	n/a	1.7GHz	Qualcomm

The Infineon TC297 platform, a member of the AURIX TriCore family, is aimed at demanding automotive applications such as powertrain, safety or connectivity for high-end products. The three cores included can run at a top frquency of 300MHz and come with 8MB of Flash and 728KB of RAM.

[1] https://www.microchip.com/design-centers/32-bit/sam-32-bit-mcus/sam-v-mcus

Fig. 4. Experimental setup: SAM V71, Infineon TC297 and an Android smartphone

We also chose to include an Android smart phone in our experiments due to two obvious reasons. First, smart phones are gadgets that are commonly brought inside cars and may be present in many future applications that allow a smart phone to communicate with in-vehicle ECUs. Second, Android-based in-vehicle infotainment units are common and these may be similar in computational capabilities to an Android smart phone. The phone that we used was a HTC One M7. It has a 1.7 GHz quad-core Krait 300 processor, based on ARM architecture, and runs the Android 4.1.2 operating system.

2.3 Cryptographic Libraries

The first step in our project was to configure the wolfCrypt library[2] to run on our automotive microcontrollers. WolfCrypt is a lightweight cryptographic library which contains the most popular algorithms, it has a compact size and a good run-time. The wolfCrypt cryptography engine is targeted for embedded devices and for environments where the resources are an important constraint. We used the wolfSSL 3.12.2 package which includes the wolfCrypt engine. The library was necessary to measure the conventional RSA and DSA algorithms both in terms of run-time and storage space which are the baseline of our tests.

Further, we integrated an open-source C library for bilinear pairings made available by the Institute for Applied Information Processing and Communication (IAIK)[3]. The library source code is maintained on a Github repository and the research results are presented in [20]. In the case of SAM V71 we added the source files in our Atmel project then we configured the architecture type to Cortex-M0, because this was the only ARM-based architecture available and eliminated functions that were not compatible with our system. The generic architecture configuration was used for in the case of our Infineon microcontroller.

[2] https://www.wolfssl.com

[3] http://www.iaik.tugraz.at/content/research/opensource

Of high interest for our applications are the short signature schemes in [6] since their size may make them specifically affordable for constrained in-vehicle communication buses. For the BLS primitive we used the C implementation developed by Ben Lynn[4]. This cryptographic library was build on the GNU Multiple Precision Arithmetic Library (GMP), which is developed specificaly for Unix systems. We succeeded to compile the GMP library in Ubuntu with a compiler set for the ARM Cortex-M7 architecture and thus we could build a static library which was added to our Atmel project. Afterwards, we performed small modifications in the source files of the Pairing-Based Cryptography library to successfully compile and run it on our V71 microcontroller. We were not successful in adapting the code to compile on the TC297.

In order to run the BLS signature scheme on our smartphone we downloaded the Java Pairing-Based Cryptography library from [8], which is a port of the PBC library by Ben Lynn, and we performed some modification to make it suitable for our Android device. The integration was smooth without much difficulties.

3 Results

For each function in each library we analyze both the runtime and the storage space needed for the code by summing the data flash and code flash size. We discuss results on the Atmel V71 and Infineon TC microcontrollers then on the Android device.

3.1 Results on Automotive-grade Microcontrollers

We first performed measurements for the RSA and DSA algorithms, which were included as a baseline for comparison with the pairing-based alternatives. The RSA algorithm has signatures that are generally too large for an automotive scenario, e.g., 2048 bits vs. 320 bits for DSA. On the other hand, not surprising, RSA is still the fastest signature in terms of verification speed with up to one order of magnitude faster than DSA. The RSA and DSA implementations that we used are from the aforementioned WolfCrypt library.

Next, we analyzed the C library for bilinear pairings implemented in [20]. We run the following four different examples included in the library: i) demonstration of the bilinear property, ii) the group signature scheme by Boneh et al.,[4] iii) the group signature scheme by Hwang et al., [13] iv) a demonstration of a key encapsulation variant of the identity-based encryption by Boneh et al. [3]. The library operates on two Barreto-Naehrig curves BN158 and BN254 and as expected the first one performs better but also has a lower level of security. For long term use, e.g., software signing, a higher level of security should be desired (resulting in a discrete logarithm key of ~ 256 bits). For real-time applications, e.g., vehicle-to-vehicle communications, a discrete logarithm key of $\sim$160 bits as provided by the first curve, i.e., BN158 which is the faster, should offer sufficient security. Tables 2 and 3 show the results for the Atmel and Infineon cores running the pairing library. The results are also depicted as bar-charts in Figures 5 and 6 for Atmel and Figures 7, 8 for Infineon. The Infineon TC297 clearly outperforms the

[4] https://crypto.stanford.edu/pbc

Atmel core, this was not expected by us since their CPUs have similar capabilities. On the other hand, the TriCore is well known as premium core in the automotive industry.

Finally, we executed on the Atmel device the BLS signature scheme that is part of the Pairing-based cryptography library implemented by Ben Lynn. This library requires the GNU crypto toolset which for the moment we were not able to port on the Infineon core. Thus we only provide results on the Atmel core. In the BLS library[5] the user has the possibility to choose over several types of curves. The predefined types curves for pairings are the following: A, A1, D159, D201, D224, E, F, G149. According to the documentation of the library, type A curves are the fastest while type D, F and G are for compact representations but obviously slower. We have measured computational performance for each type of curve and the results are consistent with the documentation. The only exception was that type G curve gave similar results to type F curves which according to the documentation should have been faster. In Table 4 we present measurement results in terms of run-time and flash size for the Atmel core when running the BLS signature scheme.

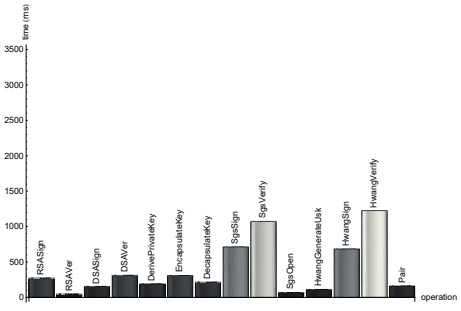

Fig. 5. Results as bar-charts for the pairing library [20] on Atmel ATSAMV71 for BN158

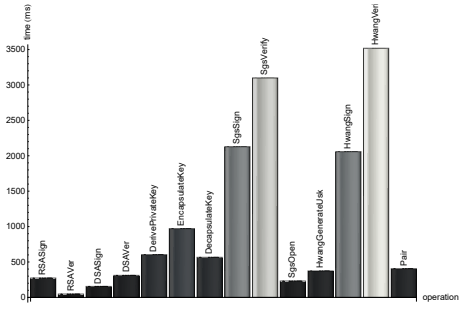

Fig. 6. Results as bar-charts for the pairing library [20] on Atmel ATSAMV71 for BN254

3.2 Results on the Android Device

As expected, Android based smartphones are equipped with highly capable cores. Despite the five years that passed since the release of the HTC phone that we used, it still generally outperforms the automotive grade controllers. On the Android device, we present results only for the most demanding BLS signature scheme for each type of curve. Each result set of the BLS scheme on a specific curve contains only the runtime measurements since the storage space is of no concern on this device. The results collected on Android are shown in Table 5. They are generally better by one order of magnitude compared to the Atmel core, but variations exists, for example on the D159 curve. Giving the use of Android units for more complex tasks, e.g., image processing, it is likely that demanding cryptographic functions such as pairing will raise no computational issues.

[5] https://crypto.stanford.edu/pbc

Table 2. Results for WolfCrypt and the pairing library [20] on Atmel ATSAMV71Q21

Function	Procedure	Flash [bytes]	Runtime [ms]	Signature [bytes]
RSA	MakeRsaKey	34176	7783	128
	RsaSSLSign		273	
	RsaSSLVerify		47	
RSA	MakeRsaKey	34176	54122	256
	RsaSSLSign		1281	
	RsaSSLVerify		155	
DSA	InitDsaKey	24628	38350	40
	DsaSign		155	
	DsaVerify		311	
IBE(BN158)	GenParams	20868	315	N/A
	DerivePrivateKey		190	
	EncapsulateKey		309	
	DecapsulateKey		215	
IBE(BN254)	GenParams	20868	941	N/A
	DerivePrivateKey		603	
	EncapsulateKey		971	
	DecapsulateKey		566	
SGS(BN158)	SgsInit	23416	400	252
	SgsSign		713	
	SgsVerify		1073	
	SgsOpen		68	
SGS(BN254)	SgsInit	23416	1247	396
	SgsSign		2128	
	SgsVerify		3099	
	SgsOpen		231	
HWANG scheme(BN158)	HwangInitParams	27340	474	232
	HwangGenerateUsk		112	
	HwangSign		684	
	HwangVerify		1226	
HWANG scheme(BN254)	HwangInitParams	27340	1571	364
	HwangGenerateUsk		373	
	HwangSign		2058	
	HwangVerify		3518	
Bilinear pairings (BN158)	PbcMapOptAteOptimizedStd	16988	161	N/A
Bilinear pairings (BN254)	PbcMapOptAteOptimizedStd	16988	405	N/A

4 Conclusion

The scenarios envisioned in our work show clear practical advantages for pairing-based primitives in the automotive domain. Computational results obtained on an automotive grade controller with two libraries dedicated to pairing based primitives show that implementation on automotive-grade cores is feasible. Nonetheless, for in-vehicle infotainment units which have even better cores, the performance is improved. This proves that pairing-based primitives are ready for entering the automotive domain. Indeed, traditional public-key primitives such as RSA and DSA still have better performance but they cannot offer the compact size of BLS signatures, nor the flexibility of identity-based signatures/encryptions or the scalability of group signatures. Consequently, the adoption of pairing based primitives in the automotive domain may be a future step. Developing a full-scale functional application for any of the discussed scenario remains as future work for us.

Table 3. Results for WolfCrypt and the pairing library [20] on Infineon TC297

Function	Procedure	Flash [bytes]	Runtime [ms]	Signature [bytes]
DSA	DsaSign	61988	29.4	40
	DsaVerify		57.8	
IBE (BN158)	GenParams	29442	78.9	N/A
	DerivePrivateKey		46.8	
	EncapsulateKey		78	
	DecapsulateKey		54.9	
IBE (BN254)	GenParams	30268	225	N/A
	DerivePrivateKey		146	
	EncapsulateKey		235	
	DecapsulateKey		138.4	
SGS (BN158)	SgsInit	30302	81.2	252
	SgsSign		173.6	
	SgsVerify		264.2	
	SgsOpen		15.7	
SGS (BN254)	SgsInit	31234	264.2	396
	SgsSign		511.6	
	SgsVerify		745.6	
	SgsOpen		53.4	
HWANG scheme (BN158)	HwangInitParams	29704	114	232
	HwangGenerateUsk		24.12	
	HwangSign		166.2	
	HwangVerify		304.4	
HWANG scheme (BN254)	HwangInitParams	30778	368.8	364
	HwangGenerateUsk		80.80	
	HwangSign		488	
	HwangVerify		853	
Bilinear pairings (BN158)	PbcMapOptAteOptimizedStd	26504	40.9	N/A
Bilinear pairings (BN254)	PbcMapOptAteOptimizedStd	27320	102.7	N/A

Table 4. Results for the BLS library [6] on Atmel ATSAMV71Q21

Function	Procedure	Flash [bytes]	Runtime [ms]	Signature size [bytes]
BLS-A	BLSSign	324564	4828	64
	BLSVerify		7286	
BLS-A1	BLSSign	324564	37516	130
	BLSVerify		31122	
BLS-D159	BLSSign	324564	251	20
	BLSVerify		5237	
BLS-D201	BLSSign	324564	488	26
	BLSVerify		10459	
BLS-D224	BLSSign	324564	683	28
	BLSVerify		12581	
BLS-E	BLSSign	324564	41072	128
	BLSVerify		36621	
BLS-F	BLSSign	324564	236	20
	BLSVerify		25621	
BLS-G149	BLSSign	324564	227	19
	BLSVerify		18640	

Acknowledgement. We thank the reviewers for helpful comments that improved our work. This work was supported by a grant of the Romanian Ministry of Research and Innovation, CNCS-UEFISCDI, project number PN-III-P1-1.1.-TE-2016-1317 within PNCDI III (2018-2020).

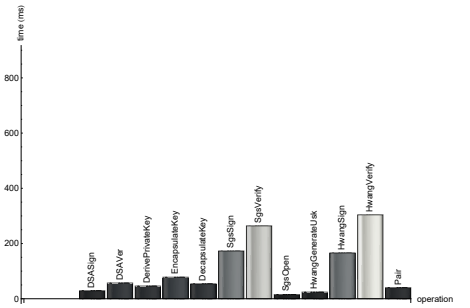

Fig. 7. Results as bar-charts for the pairing library [20] on Infineon TC297 for curve BN158

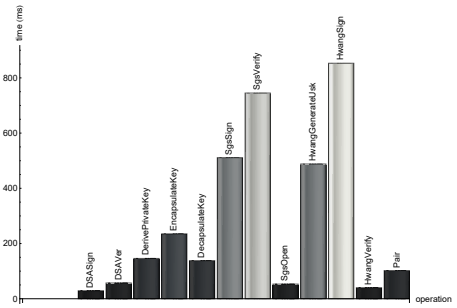

Fig. 8. Results as bar-charts for the pairing library [20] on Infineon TC297 for curve BN254

Table 5. Results for the BLS library [6] on a HTC smartphone running Android

Function	Procedure	Flash [bytes]	Runtime [ms]	Signature size [bytes]
BLS-A	GenKeys	N/A	351	64
	BLSSign		685	
	BLSVerify		1584	
BLS-A1	GenKeys	N/A	3226	130
	BLSSign		2122	
	BLSVerify		10129	
BLS-D159	GenKeys	N/A	3561	20
	BLSSign		156	
	BLSVerify		10796	
BLS-D201	GenKeys	N/A	4122	26
	BLSSign		252	
	BLSVerify		12815	
BLS-D224	GenKeys	N/A	12019	26
	BLSSign		398	
	BLSVerify		15630	
BLS-E	GenKeys	N/A	642	128
	BLSSign		1707	
	BLSVerify		2972	
BLS-F	GenKeys	N/A	5912	20
	BLSSign		184	
	BLSVerify		86174	
BLS-G149	GenKeys	N/A	8823	19
	BLSSign		195	
	BLSVerify		34168	

References

1. AUTOSAR. *Specification of Crypto Abstraction Library*, 4.2.2 edition, 2015.
2. AUTOSAR. *Specification of Crypto Service Manager*, 4.2.2 edition, 2015.
3. D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology Crypto 2004*. Springer Berlin Heidelberg, 2004.
4. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology Crypto 2004*. Springer Berlin Heidelberg, 2004.
5. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Annual international cryptology conference*, pages 213–229. Springer, 2001.
6. D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. volume 17, pages 297–319. Springer, 2004.

7. S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno, et al. Comprehensive experimental analyses of automotive attack surfaces. In *USENIX Security Symposium*. San Francisco, 2011.
8. A. De Caro and V. Iovino. jPBC: Java pairing based cryptography. In *Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011*, pages 850–855, Kerkyra, Corfu, Greece, June 28 - July 1, 2011. IEEE.
9. K. Dhondge, S. Song, B.-Y. Choi, and H. Park. WiFiHonk: smartphone-based beacon stuffed WiFi Car2X-communication system for vulnerable road user safety. In *Vehicular Technology Conference (VTC Spring), 2014 IEEE 79th*, pages 1–5. IEEE, 2014.
10. B. Groza and C. Briceag. A vehicle collision-warning system based on multipeer connectivity and off-the-shelf smart-devices. In *International Conference on Risks and Security of Internet and Systems*, pages 115–123. Springer, 2017.
11. B. Groza and P.-S. Murvay. Security solutions for the controller area network: Bringing authentication to in-vehicle networks. *IEEE Vehicular Technology Magazine*, 13(1):40–47, 2018.
12. J. Harding, G. Powell, R. Yoon, J. Fikentscher, C. Doyle, D. Sade, M. Lukuc, J. Simons, and J. Wang. Vehicle-to-vehicle communications: Readiness of v2v technology for application. Technical report, 2014.
13. J. Y. Hwang, S. Lee, B.-H. Chung, H. S. Cho, and D. Nyang. Group signatures with controllable linkability for dynamic membership. volume 222, pages 761–778. Elsevier, 2013.
14. K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, et al. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 447–462. IEEE, 2010.
15. A. J. Menezes, T. Okamoto, and S. A. Vanstone. Reducing elliptic curve logarithms to logarithms in a finite field. *iEEE Transactions on information Theory*, 39(5):1639–1646, 1993.
16. C. Miller and C. Valasek. A survey of remote automotive attack surfaces. *Black Hat USA*, 2014.
17. M. Steger, C. A. Boano, K. Römer, M. Karner, J. Hillebrand, and W. Rom. Cesar: A testbed infrastructure to evaluate the efficiency of wireless automotive software updates. In *Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems*, pages 311–315. ACM, 2017.
18. M. Steger, A. Dorri, S. S. Kanhere, K. Römer, R. Jurdak, and M. Karner. Secure wireless automotive software updates using blockchains: A proof of concept. In *Advanced Microsystems for Automotive Applications 2017*, pages 137–149. Springer, 2018.
19. J. Sun, C. Zhang, Y. Zhang, and Y. Fang. An identity-based security system for user privacy in vehicular ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 21(9):1227–1239, 2010.
20. T. Unterluggauer and E. Wenger. Efficient pairings and ecc for embedded systems. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 298–315. Springer, 2014.
21. C. Zhang, R. Lu, X. Lin, P.-H. Ho, and X. Shen. An efficient identity-based batch verification scheme for vehicular sensor networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 246–250. IEEE, 2008.