EFFICIENT AUTHENTICATION FOR CONSTRAINED DEVICES - SECURING THE LOW POWER INTERNET OF THINGS

Tobias Markmann
Hamburg University of Applied Sciences, Dept. Computer Science,
Berliner Tor 7
20099 Hamburg, Germany
tobias.markmann@haw-hamburg.de

ABSTRACT

In this work we analyze the suitability of identity-based signatures (IBSs) as a means to provide efficient authentication for the Internet of Things (IoT). We analyze how regular key management tasks like key renewal and key revocation in classic public-key infrastructure (PKI) using certificate revocation lists (CRLs) and Online Certificate Status Protocol (OCSP) are treated in identity-based cryptography (IBC) and highlight the consequences of each solution in light of the large scale IoT. Furthermore, we practically study the computational and storage complexity of three IBS by implementing them in C/C++ using the RELIC toolkit followed by a benchmark on an Intel Core i7 and a Raspberry Pi as example for a constrained device.

We implemented and benchmarked the signatures SH–IBS, vBNN–IBS and TSO–IBS and show that signatures based on elliptic curve cryptography (ECC) like vBNN-IBS are most suitable for constrained devices compared to the other two.

1. INTRODUCTION

The Internet of Things (IoT) is a continuously active research field with many practical applications emerging. The applications of the IoT range from simple home control applications, like thermostats or lighting control devices, to more backbone applications, like smart grid or smart city developments [1]. However, in all those applications one core aspect is the interconnectivity of participating devices in one global computer network, the Internet.

While the use of the public Internet provides easy and global interconnection of devices, the installer of IoT devices has little to no control over the actual communication media used in the end, including the security of the media. Internet access can be gained through cable, broadband and wireless connectivity which rarely come with strong security layers.

For low power devices some IoT applications can resemble those of the wireless sensor network (WSN), especially monitoring and controlling applications. However, WSNs are usually locally limited, have their own network and are specialized for their applications. For example, a WSN for detecting fires in wild forests can be considered to have lower security requirements than applications for the global IoT. The nodes in a WSN are also mainly connecting to their collecting sink station and are, on their own, not publicly accessible from the outside network. In contrast to that, it's a key aspect of the IoT that all devices are accessible in a direct way via the Internet.

Identity-based cryptography (IBC) offers a great approach for enabling secure communication in the IoT, because it simplifies key management. To verify signatures, no certificates need to be transmitted or public keys looked up, since already existing identification information in packets can be used to deduce a public key.

On the one side this work provides a detailed overview and comparison of key management tasks like key renewal and key revocation. We compare how key renewal and revocation is handled by the classic public-key infrastructure (PKI) and IBC. These management tasks are important for the IoT because most devices are unmanned and a lot are publicly accessible. Thus tampering of IoT devices is hardly avoidable and the network must deal with the consequences of private key compromises.

On the other side this work aims to present the practical results of an evaluation of different identity-based signatures (IBSs) schemes and show how they compare to each other in the aspects of storage and computational complexity. We cover IBS using three different cryptographic mechanisms: the classic RSA problem, elliptic curve cryptography (ECC) and pairing-based cryptography (PBC).

Organization: The remainder of the paper is organized as follows. Section 2 briefly introduces the basic background on IBC, ECC and PBC, followed by Section 3 showing other projects and how they relate to this work. Section 4 covers the key management situation in classic PKI and in IBC, how they compare to each other and in relation to the IoT. In Section 5 we introduce our practical evaluation of three IBS, SH–IBS, vBNN–IBS, and TSO–IBS. It is performed on an Intel 64-bit desktop platform and a Raspberry Pi (32-bit ARM). We conclude in Section 6 and provide an outlook on our future work in Section 7.

2. BACKGROUND

2.1. Identity-based Cryptography

Identity-based cryptography (IBC) [2] was proposed by Shamir in 1985 as a kind of asymmetric cryptography with easier key management compared to traditional public-key infrastructure (PKI).

Like each user in the PKI has a private/public key pair each user in an ID-based cryptosystem has the same key pair. In the PKI setting the private key can be generated by the user and the trusted third party (certificate authority (CA)) only signs this public key. In the IBC setting the private key must be generated by the third party, in this case the trusted authority (TA).

The generated private key is implicitly bound to the ID string of an authenticated user. In the classic example an ID string is simply an email address (`alice@wonderland.lit`) but one can

use an arbitrary string here, including IP- and Ethernet addresses. A prerequisite is, however, that the ID string is easily predictable by all users of the system because it is used to generate each users' public key.

Having the TA generating the private keys for all users requires an encrypted connection between the user and the TA for the transfer of the private key as with all *key escrow* systems. However, in a classic PKI without *key escrow* only an authenticated connection is needed between the user and the CA. *Key escrow* describes the ou sourcing of private key information to a trusted third party and stands in conflict with plausible deniability.

The ability to generate the public key from an ID string greatly reduces key distribution work. Compared to PKI you do not need to distribute certificates or public keys of the users in advance or on demand over the wire. In case of on demand distribution this also reduces the packet size for authenticated communication. Only the signature needs to be transferred in addition to the already existing identity and the actual message.

2.2. Elliptic Curve Cryptography

The first asymmetric cryptosystems providing signatures were RSA [3] by Rivest, Shamir, and Adleman and the ElGamal [4] cryptosystem by El Gamal. The security of RSA builds on the problem of factoring large composites of prime numbers where the security of the ElGamal cryptosystem is based on the problem of calculating discrete logarithms in groups of large prime order.

However, both problems are not as hard as originally assumed. There are subexponential-time complexity algorithms to factor integers as there are for solving the discrete logarithm problem (DLP) in finite fields [5, Chapter 2]. In response to the advances in solving these problems key sizes have been increased. This results in larger signatures and the more computation is needed for signature generation and verification.

For the elliptic curve discrete logarithm problem (ECDLP), however, there has not yet been discovered a subexpoential-time complexity algorithm to solve it. The best known algorithm today to solve the ECDLP on general elliptic curves is Pollard's rho algorithm for discrete logarithms [6]. The difference in the complexity between the classic problems and the newer ECDLP can also be seen in the recommendation for key sizes from standards and research bodies. Figure 1 shows the recommendation for key sizes for classic cryptosystems and elliptic curve cryptosystems. While elliptic curve cryptography (ECC) key sizes scale linearly with the respective symmetric key size, classic RSA and DLP based cryptosystems scale super linear.

The use of elliptic curves in cryptography was independently suggested by Miller [8] and Koblitz [9] in the mid 1980s. Traditionally, elliptic curves are defined in short Weierstrass form as $E(\mathbb{F}_q) = \left\{(x, y) \in {\mathbb{F}_q}^2 : y^2 = x^3 + ax + b\right\} \bigcup \mathcal{O}$ describing a group of elliptic curve points. Point $\mathcal{O}$ represents the additive identity and $\mathbb{F}_q$ is the field used for the x, y-coordinates. Additionally, to avoid certain attacks the elliptic curves are required to be non-singular. This requires a discriminat of $\Delta = 4a^3 + 27b^2 \neq 0$. The basic group operation of elliptic curve groups is the addition of two points.

The cyclic group of elliptic curve points is defined by the *generator* or *base point* P. Repeated application of the group operation to P generates all elements of the group. For an elliptic curve group of of order n, there exists a point P, which generates the elliptic curve group $E(\mathbb{F}_p) = \{kP : 1 \leqslant k \leqslant n\}$.

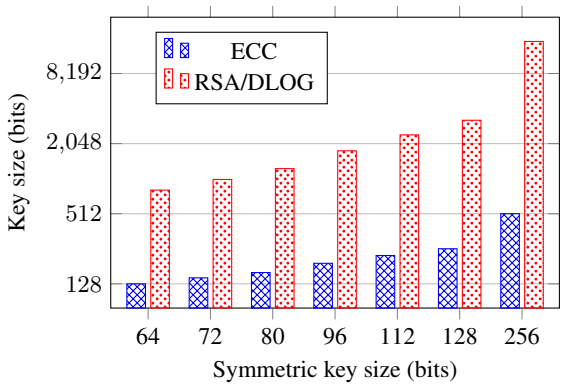

Figure 1: ECRYPT II comparison of key sizes (in bits) at the same security level between symmetric, asymmetric (RSA) and elliptic curve [7].

2.3. Pairing-based Cryptography

A bilinear pairing is a map, e, from elements of two groups, G_1 and G_2, in a target group G_T: $e : G_1 \times G_2 \longrightarrow G_T$. With $a, b \in \mathbb{Z}$, $P \in G_1$ and $Q \in G_2$, it has the following properties:

- Bilinearity: $e(aP, bQ) = e(P, bQ)^a = e(aP, Q)^b = e(P, Q)^{ab}$
- Non-degeneracy: $\forall P \in G_1, Q \in G_2 : e(P, Q) \neq 0$
- Efficiently computable: for bilinear pairings to be useful in cryptographic protocols, efficient implementations computing the pairing must exist.

Furthermore bilinear pairings are categorized as either symmetric or asymmetric. The previous definition specifically describes asymmetric pairings. The symmetric case can be seen as a special case of the asymmetric one, with $G_1 = G_2$ [10]. It is worth noting that symmetric pairings are usually realized using a group of points on a supersingular curve as G_1. Compared to asymmetric pairings symmetric pairings are less efficient.

Pairing-based cryptography (PBC) was first used as crypto-analytic tool in the MOV attack [11]. It reduces the ECDLP on supersingular elliptic curve to the easier DLP in an extension field $\mathbb{F}_{p^k}$.

Joux first suggested to use bilinear pairings for cryptographic purposes in 2000 [12]. He proposed a protocol for computing a shared secret among three parties, also known as tripartite Diffie-Helman, in one round.

3. RELATED WORK

Kiltz and Neven [13] theoretically analyzed and compared various ID-based signature schemes with regard to signature size, computational complexity and security strength. Their comparison provides a good high-level overview on the computational and storage complexity of ten different identity-based signatures (IBSs) including SH–IBS and vBNN–IBS. The complexity is specified in terms of group operations for computation complexity and number of group elements for space complexity.

However, there is no practical evaluation via implementation and benchmark. We implement three IBSs and provide benchmark results for a desktop and an embedded platform.

A direct high-level comparison between identity-based cryptography (IBC) and public-key infrastructure (PKI) is provided by Paterson and Price [14]. They compare not only the architectural issues of either system but also the differences in key management. In addition to their overview on key revocation issues in both systems we provide a detailed analysis of the problem of key renewal and key revocation in IBC and PKI for application in the Internet of Things (IoT).

4. KEY MANAGEMENT FOR ASYMMETRIC CRYPTOSYSTEMS

Employing asymmetric cryptography in real applications comes with essential auxiliary tasks as part of the key management. To provide a good level of security to all members of a communication system, developers have to bow to the inevitable and prepare their systems for incidents like key exposure and key renewal. Key exposure usually requires key renewal, since the now publicly known private key could be used to impersonate a legitimate user.

There are various ways a user's private key could be exposed. This can happen through human error in manual processes and more critically, due to bugs in security relevant protocol implementations exposed to public networks. This has recently occurred with the Heartbleed bug in OpenSSL [15]. After possible exposure of a private key to the public there is only one correct process; the certificate associated with the exposed key needs to be revoked to limit possible damage and a new certificate with new private/public key pair needs to be obtained.

4.1. Traditional Public-key Infrastructure

In a public-key infrastructure (PKI) as used by the world wide web (WWW), users can revoke their certificate at the issuing party, the certificate authority (CA). However, the task of actively checking certificates for their current revocation state is left to the clients. Basically there are two different kind of approaches to this problem:

1. *offline* / *asynchronous*: during the verification step each user checks the certificate against a list of keys that have been revoked, the so called certificate revocation list (CRL). This list is issued by a CA and signed using their private key. CRLs can be downloaded on demand or pushed to the users on a regular basis. The handling of CRLs in the PKI is described in more detail in [16].
2. *online* / *synchronous*: for verification, the user asks a predefined server about the current revocation state of a specific certificate to which the server responds with a signed reply containing the current revocation state. This protocol for the PKI is called Online Certificate Status Protocol (OCSP) [17] and allows a secure on-demand attestation on the state of revocation of a certificate. The OCSP server is provided by the CA that issued the certificate.

The offline approach comes with a scalability issue. The list of revoked certificates only ever increases and has to include all revoked certificates that would otherwise still be valid. In addition, the CRLs need to be updated at all clients at a regular bases to correctly detect revoked certificates.

In contrast, OCSP does not require updating huge lists of certificates. However, it requires that all CAs have an OCSP server running which will reply to the requests of clients checking the revocation state of certificates. In a setup which prioritizes security over usability, a certificate would be considered revoked if an OCSP server is not reachable. Thus, the OCSP server introduces a single point of failure which may be under heavy load considering each validation of a certificate requires a request to an OCSP server.

To reduce load on the OCSP server, an optimization has been proposed. OCSP stapling [18] allows the verifier of a certificate to immediately check the revocation state without further contacting an OCSP server. This is possible because the communication partner already requested an OCSP response from the CA and attached it to the message to be verified.

4.2. ID-based Cryptography

While identity-based cryptography (IBC) eases key distribution compared to traditional PKI, the problems of key management are of at least similar complexity. Especially the inherent implicit binding of public key and identity in IBC makes it hard to revoke keys for users. Simply adding the identity to a revocation list would prevent the user of that identity from ever sending signed messages again. Early systems using IBC avoided the classic approach of revocation altogether and instead went with automatic key renewal [19]. Here Boneh and Franklin proposed to add time related information to the identity before deriving the associated public key from it, e.g. `identity + year`. In this way users are required to get a new private key from the trusted authority (TA) each year and the TA just stops handing out private keys to revoked users. However, adding time related information requires all users to fetch a new private key from the TA in the common time frame.

Revoking public keys in an IBC system equals revoking the associated identity. If however the identity is hard to change (like static Internet protocol (IP) addresses) the identity string needs to be extended to still allow revocation and rekeying. Extending the identity string with additional information is nontrivial. Identities in IBC systems need to be easy to predict to allow easy verification. Adding a rough timestamp as proposed by Boneh and Franklin still allows easy verification considering loosely synchronized clocks between the users. However, adding hard to predict data, e.g. issue numbers, to identities complicates the verification process as information about the current valid issue number needs to obtained out-of-band [14, p. 64].

Boldyreva, Goyal, and Kumar [20] proposed an identity-based encryption (IBE) system with improved revocation handling, providing logarithmic scaling for maintenance work for all users of the system as compared to linear within a revocation time frame. However, it heavily uses pairing-based cryptography (PBC) and thus comes with great computational complexity.

Considering the two major environments where constraint devices are in wide use—the Internet of Things (IoT) and wireless sensor networks (WSNs)—the key management properties are of different relevance in each environment. While required continuous key updates within the revocation timeframe in an IBC setup might well work in small WSNs with up to 1000 nodes, in a sys-

tem at the scale of the IoT this could become a greater problem. The flexible key management in traditional PKI setups on the other hand allow for various different ways of providing revocation information to the users of the system and is easily distributable.

5. PRACTICAL PERFORMANCE EVALUATION

We evaluate three identity-based signatures (IBSs) on two hardware and software platforms to verify the use of IBSs for the Internet of Things (IoT).

5.1. Tested ID-based Signatures

The following three IBSs have been implemented in C/C++ using the RELIC toolkit [21]:

1. SH–IBS [2]: This is Shamir's original signature based on the RSA cryptosystem. It requires arbitrary precision numbers which are supplied by the BN module of RELIC.

   We compare SH–IBS against the other IBS of our evaluation at different asymmetric security levels from 768 bit to 2048 bit.

2. vBNN–IBS [22]: This IBS by Cao, Kou, Dang, *et al.* uses elliptic curve cryptography (ECC) and its security is based on the elliptic curve discrete logarithm problem (ECDLP). vBNN–IBS is compared to the other signature schemes in our evaluation.

   We test the implemented signature schemes at different security levels using standard curves like NIST-P256 [23] but also more recent proposals like Curve383187 [24]. The asymmetric security strength of the curves ranges from 251 bit to 384 bit.

3. TSO–IBS [25]: The bilinear pairing based TSO–IBS was proposed by Tso, Gu, Okamoto, *et al.* and in contrast to the other two test candidates it provides message recovery. This means that the receiver of the signature can recover the original message from it thereby eliminating the need to transfer the actual message.

   TSO–IBS as described by the authors uses symmetric pairings and comes with the associated performance penalty of its implementation described in Subsection 2.3. The RELIC toolkit currently only provides a single supersingular curve, namely SS-P1536. Therefore we are not able to evaluate TSO–IBS at different security levels.

RELIC toolkit is a C library efficiently implementing cryptographic primitives like finite fields, elliptic curves and pairings for use on constrained devices.

5.2. Evaluation Hardware/Software Setup

The hardware/software setup used for the benchmarks is shown in Table 1. All benchmark code including the RELIC toolkit itself is compiled with the clang compiler (version 3.5.0).

We run each benchmark program five times per test combinations and measure the wall-clock time in each run. In addition, the CPU cycle count is measured for signature generation and verification of each test candidate. Within the benchmark program, the RELIC toolkit is initialized and keys are generated. Afterwards we measure the time for generating 100 signatures and the time for verifying 100 signatures. It is worth mentioning that the measurements not only cover the mathematical algorithms, but also the random number generation used during signature generation. The best results of all five runs for each signature scheme are taken in comparison. The best result has the smallest time and cycle measurements. They suggest as little as possible noise activity on the system.

	Desktop	Embedded
Device	Laptop	Raspberry Pi
OS	Mac OS X 10.9.3	Debian 3.10.11-1+rpi7
Vendor	Intel (CISC)	ARM (RISC)
CPU	Core i7	ARM1176
Word size	64 bit	32 bit
Clock speed	2 GHz	800 MHz
L1 Cache	64 kB	32 kB
L2 Cache	256 kB	(256 kB)
L3 Cache	6 MB	—

Note: The L2 Cache is used by the GPU on the Raspberry Pi and therefore is not available to the CPU.

Table 1: Hardware/Software properties of the test environments used for the benchmark.

5.3. Benchmark Results

This subsection presents the results of our conducted benchmark and points out interesting observations in the results.

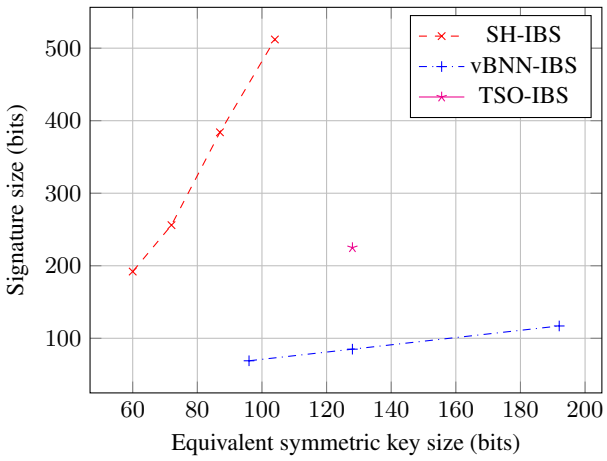

Figure 2: Signature size comparison of the ID-based signatures from the benchmark results.

Signature Size: We begin with the signature sizes of the three IBS as shown in Figure 2. The plot clearly shows the linear scalability of ECC based vBNN–IBS. RSA based SH–IBS has more than twice the signature size even at the lowest security strength level tested. The signatures of TSO–IBS show about twice the size as vBNN–IBS. Since we only have one elliptic curve available for the symmetric pairing setting no statement about scalability of the signature with increasing security level can be made.

Signature Performance: Figure 3 and Figure 4 show the benchmark timing results for signature verification and generation on the embedded 32 bit ARM platform and the desktop 64 bit Intel platform respectively.

Comparing the embedded and desktop platform directly ECC based vBNN–IBS has a clear advantage on the embedded ARM platform. At about 90 bit symmetric key size the ECC based IBS tops the signing and verification performance of SH–IBS. On the desktop platform the same occurs only at about 130 bit key size. This is likely due to extra penalty of big number calculations needed for RSA based SH–IBS which becomes problematic at smaller word sizes. ECC, however, works with smaller numbers in comparison which are easier to handle by 32 bit CPUs. Similar findings have been produced by Gura, Patel, Wander, *et al.* [26] comparing 1024 bit RSA with 160 bit ECC on a low power constrained 8 bit-CPU.

The performance of paring based TSO–IBS is behind SH–IBS and vBNN–IBS at the measured security level regardless of platform. It would be interesting to see how TSO–IBS performs using asymmetric pairings which are known to be more efficient. However, adjustments to the scheme in that direction have yet to be made to conduct an evaluation.

Apart from the architectural advantage of ECC over RSA cryptosystems on embedded platforms, Figure 3 and Figure 4 also indicate the higher performance decrease of RSA with increasing symmetric security strength. While the performance vBNN–IBS decreases rather slowly we stopped benchmarking SH–IBS at 2048 bit asymmetric security strength (105 bit symmetric security) due to long benchmark runtime.

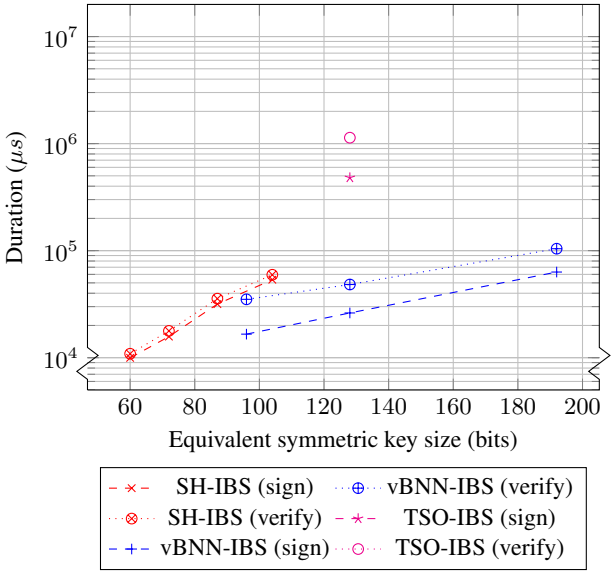

Figure 3: Comparison of signature generation and signature verification timings on the embedded platform.

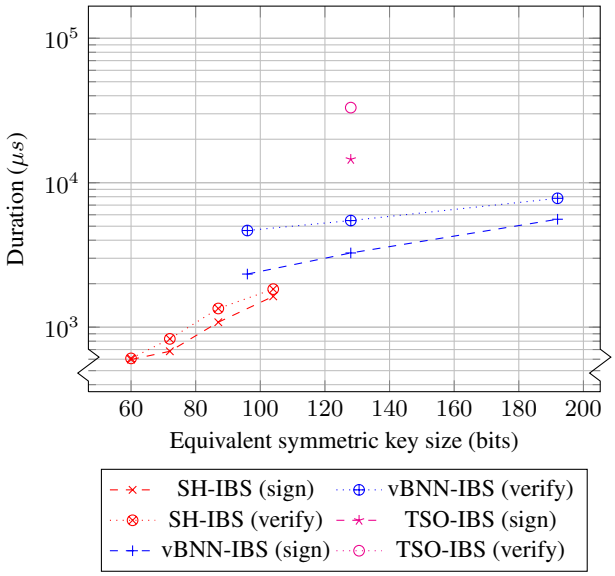

Figure 4: Comparison of signature generation and signature verification timings on the desktop platform.

6. CONCLUSION

Identity-based cryptography (IBC) was initially proposed by Shamir [2] to provide asymmetric cryptography with easier key management as compared to the classic public-key infrastructure (PKI). This statement only holds at a first glance.

Looking at classic key management tasks like key renewal and key revocation IBC shows its own set of problems due to the strong identity/public-key binding as explained in Section 4. Constantly reissuing keys in IBC poses similar difficulties to the system as providing high available Online Certificate Status Protocol (OCSP) servers in the PKI. However, IBC allows automatic revocation where in the PKI you need explicit protocols that check for validity. It turns out that classic asymmetric cryptography with its explicit identity/public-key binding also provides more flexibly revocation schemes. This makes them more suitable to a large scale Internet of Things (IoT) while IBC might suit the smaller and self-contained wireless sensor network (WSN) better.

Over the years the recommendations for key sizes for a specified security level steadily increased. This is due advances in hardware which increase the speed at which cryptosystems can be broken via brute force [27].

With this in mind elliptic curve based vBNN–IBS is a more suitable choice compared to SH–IBS. As the recommended symmetric key size increases over the years cryptosystems based on the RSA problem will have higher storage and computational requirements for signatures as can be handled by constrained embedded devices. elliptic curve cryptography (ECC) allows much smaller signatures and more efficient computation of signatures, especially on embedded devices.

TSO–IBS performance and signature size do not show an advantage over vBNN–IBS. While both are based on ECC TSO–IBS utilizes symmetric bilinear pairings for signature generation and verification which are known for their moderate performance.

7. OUTLOOK

Having gained confidence in the choice of elliptic curves as cryptographic primitives for security mechanisms for constrained devices allows further analysis in this direction.

There are elliptic curves which are more performant than standard Weierstrass curves. A prominent example are Edwards curves which have a complete and more efficient addition formula computing the elliptic curve group operation [28]. Their formula is resilient against side-channel attacks and requires less operations in the underlying field. We plan to implement the addition formula and required curve representation for Edwards curves in the RELIC toolkit to enable further research for optimized Edwards curves for constrained devices.

Modern signature schemes specifically targeted at constrained devices is another important research area. One example is Ed25519 [29], a modern asymmetric signature with comes with significant speed improvements in all areas compared to Elliptic Curve DSA (ECDSA) and a minimal consumption of outside randomness. However, Ed25519 is optimized for 64 bit systems and in the Internet of Things (IoT) 16 bit to 32 bit systems are more widespread.

Finally, we would have to integrate a modern asymmetric signature and elliptic curves with the RELIC toolkit into a development environment for IoT applications. One possible candidate here is the RIOT OS [30]. The RIOT OS is a relatively young open source operating system for the IoT and is developed, amongst others, by FU Berlin and the Hamburg University of Applied Sciences.

8. REFERENCES

[1] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions ", *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] A. Shamir, "Identity-Based Cryptosystems and Signature Schemes", in *Advances in Cryptology — CRYPTO 1984*, ser. Lecture Notes in Computer Science, G. R. Blakley and D. Chaum, Eds., vol. 196, Santa Barbara, California, USA: Springer, Aug. 1985, pp. 47–53.

[3] R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-key Cryptosystems", *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[4] T. El Gamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms", in *Advances in Cryptology — CRYPTO 1984*, ser. Lecture Notes in Computer Science, G. R. Blakley and D. Chaum, Eds., vol. 196, Berlin, Heidelberg, Germany: Springer, 1985, pp. 10–18.

[5] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, Florida, USA: CRC Press, 1996.

[6] J. M. Pollard, "Monte Carlo methods for index computation ($\bmod p$)", *Mathematics of Computation*, vol. 32, pp. 918–924, Jul. 1978.

[7] ECRYPT II, "ECRYPT II Yearly Report on Algorithms and Keysizes", European Network of Excellence in Cryptology II, Tech. Rep., Sep. 2012, `http://www.ecrypt.eu.org/documents/D.SPA.20.pdf`.

[8] V. S. Miller, "Use of Elliptic Curves in Cryptography", in *Advances in Cryptology — CRYPTO 1985*, ser. Lecture Notes in Computer Science, H. C. Williams, Ed., vol. 218, Berlin, Heidelberg, Germany: Springer Berlin Heidelberg, 1986, pp. 417–426.

[9] N. Koblitz, "Elliptic Curve Cryptosystems", *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.

[10] S. D. Galbraith, K. G. Paterson, and N. P. Smart, "Pairings for cryptographers", *Discrete Applied Mathematics*, vol. 156, no. 16, pp. 3113–3121, 2008.

[11] A. Menezes, T. Okamoto, and S. Vanstone, "Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field", *Information Theory, IEEE Transactions on*, vol. 39, no. 5, Sep. 1993.

[12] A. Joux, "A One Round Protocol for Tripartite Diffie—Hellman", in *Algorithmic Number Theory*, ser. Lecture Notes in Computer Science, W. Bosma, Ed., vol. 1838, Berlin, Heidelberg, Germany: Springer, 2000, pp. 385–393.

[13] E. Kiltz and G. Neven, "Identity-Based Signatures", in *Identity-Based Cryptography*, ser. Cryptology and Information Security Series, M. Joye and G. Neven, Eds., vol. 2, Amsterdam, The Netherlands: IOS Press, 2008, pp. 31–44.

[14] K. G. Paterson and G. Price, "A comparison between traditional public key infrastructures and identity-based cryptography", *Information Security Technical Report*, vol. 8, no. 3, pp. 57–72, 2003.

[15] Common Vulnerabilities and Exposures, *CVE-2014-0160*, `http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160`, 2014.

[16] R. Housley, W. Polk, W. Ford, and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", IETF, RFC 3280, Apr. 2002.

[17] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", IETF, RFC 2560, Jun. 1999.

[18] D. Eastlake, "Transport Layer Security (TLS) Extensions: Extension Definitions", IETF, RFC 6066, Jan. 2011.

[19] D. Boneh and M. Franklin, "Identity-Based Encryption from the Weil Pairing", in *Advances in Cryptology — CRYPTO 2001*, ser. Lecture Notes in Computer Science, J. Kilian, Ed., vol. 2139, Berlin, Heidelberg, Germany: Springer, 2001, pp. 213–229.

[20] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based Encryption with Efficient Revocation", in *Proceedings of the 15th ACM Conference on Computer and Communications Security*, ser. Computer and Communications Security 2008, Alexandria, Virginia, USA: ACM, 2008, pp. 417–426.

[21] D. F. Aranha and C. P. L. Gouvêa, *RELIC is an Efficient LIbrary for Cryptography*, `http://code.google.com/p/relic-toolkit/`.

[22] X. Cao, W. Kou, L. Dang, and B. Zhao, "IMBAS: Identity-based multi-user broadcast authentication in wireless sensor networks", *Computer Communications*, vol. 31, no. 4, pp. 659–667, 2008.

[23] Committee on National Security Systems, *National information assurance policy on the use of public standards for the secure sharing of information among national security systems*, `https://www.cnss.gov/Assets/pdf/CNSSP_No%2015_minorUpdate1_Oct12012.pdf`, Oct. 2012.

[24] D. F. Aranha, P. S. L. M. Barreto, G. C. C. F. Pereira, and J. E. Ricardini, "A note on high-security general-purpose elliptic curves", Cryptology ePrint Archive, Tech. Rep. Report 2013/647, 2013, `http://eprint.iacr.org/`.

[25] R. Tso, C. Gu, T. Okamoto, and E. Okamoto, "Efficient ID-Based Digital Signatures with Message Recovery", in *Cryptology and Network Security*, ser. Lecture Notes in Computer Science, F. Bao, S. Ling, T. Okamoto, H. Wang, and C. Xing, Eds., vol. 4856, Berlin, Heidelberg, Germany: Springer, 2007, pp. 47–59.

[26] N. Gura, A. Patel, A. Wander, H. Eberle, and S. C. Shantz, "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs", in *Cryptographic Hardware and Embedded Systems - CHES 2004*, M. Joye and J.-J. Quisquater, Eds., ser. Lecture Notes in Computer Science, vol. 3156, Cambridge, MA, USA: Springer, 2004, pp. 119–132.

[27] A. W. Dent, "Choosing key sizes for cryptography", *Information Security Technical Report*, vol. 15, no. 1, pp. 21–27, 2010.

[28] D. J. Bernstein and T. Lange, "Faster Addition and Doubling on Elliptic Curves", in *Advances in Cryptology — ASIACRYPT 2007*, ser. Lecture Notes in Computer Science, K. Kurosawa, Ed., vol. 4833, Berlin, Heidelberg, Germany: Springer, 2007, pp. 29–50.

[29] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures", *Journal of Cryptographic Engineering*, vol. 2, no. 2, pp. 77–89, 2012.

[30] E. Baccelli, O. Hahm, M. Günes, M. Wählisch, and T. C. Schmidt, "RIOT OS: Towards an OS for the Internet of Things", in *Proc. of the 32nd IEEE INFOCOM. Poster*, Piscataway, NJ, USA: IEEE Press, 2013.