The Feasibility of On-the-Tag Public Key Cryptography

M. Girault[1], L. Juniot[1], and M.J.B. Robshaw[2]

France Telecom Research and Development

[1] 42 rue des Coutures, BP 6243, 14066 Caen, Cedex 4, France
[2] 38-40 rue du Général Leclerc, 92794 Issy les Moulineaux, Cedex 9, France
{marc.girault,loic.juniot,matt.robshaw}@orange-ftgroup.com

Abstract. The cheap and convenient RFID tag is revolutionising supply chain management. More accurate tracking of goods allows for more efficient use of delivery resources and the ability to account for every single manufactured item from the factory to the consumer. Within such restricted environments there is little need, and therefore no drive, towards supporting more than the most basic security features. Beyond the supply chain, however, two factors come into play: (1) the adversarial threat changes and (2) mechanisms for closed environments may become unsuitable. In anticipation of new and widely-deployed applications for RFID tags, we present the results of a prototype FPGA-based implementation of the GPS scheme, an established and ISO-standardised public key technique. The prototype implementation includes all tag and reader functionality as well as a wireless interface. With this prototype, the complete tag authentication process takes 200ms while only 2600 GE are required to support the on-tag cryptography.

1 Introduction

As the widespread deployment of *RFID tags* takes hold, a range of security and privacy issues come to the fore. New applications in new environments can stretch the feasibility of many typical security goals while, at the same time, introduce new and specific challenges. At the same time the range of techniques that permit a security solution can be severely limited; the physical demands of cheaper tags are such that many useful cryptographic tools are impossible to implement since they require either too much space, too much power, or both.

While this is somewhat frustrating, it is important to keep a sense of perspective. Not all security solutions for an application need be cryptographic, and for many RFID deployments one can go a long way with the most basic functionality. Practical limitations on the capabilities of an attacker, an understanding of the value of goods being protected, and alternative security measures can lead to perfectly reasonable RFID-based deployments using current technologies.

However, the ability to implement cryptographic algorithms can be helpful in several ways. First, they may help provide solutions to pressing problems such as product counterfeiting and tag-cloning, for which existing solutions become less satisfying as the value of goods increases. Second, they may help to open up new application areas and, in particular, to facilitate the adoption of RFID-based deployments in open, as opposed to closed, environments.

2 Strong on-tag cryptography

Cryptographic algorithms are often divided into two classes according to how they use key information; algorithms for which the participants use the same key material are termed *symmetric* or *secret key* while algorithms for which participants use different key material are termed *asymmetric* or *public key*. This essential difference leads to very different deployments, different supporting infrastructures, and contrasting suitability as a response to different threats.

It is a common assumption that public key cryptography is computationally more expensive that secret key cryptography. In general terms this is a convenient rule-of-thumb and, particularly recently, there has been much work on optimising both the implementation and the design of symmetric primitives. With regards to block ciphers the benchmark implementation is that of the AES due to Feldhofer *et al* [7] which, when we temporarily concentrate on implementation-size alone, requires around 3500 *gate equivalents* GE. Space requirements of 2300 GE for the proposal DESXL are outlined in [23] while an ultra-lightweight block cipher PRESENT requiring only 1570 GE is described in [3]. This is surprisingly close to the requirements of some optimised stream ciphers such as GRAIN [13] and TRIVIUM [4] that currently feature in the eSTREAM project [6].

When we turn to public key primitives, however, there are substantial obstacles to efficient implementation, particularly for algorithms such as RSA [25]. However, for variants of the GPS identification scheme [11], the *on-tag* memory and power consumption is actually less than that required by most symmetric techniques! At the same time, the computational effort for the reader remains reasonable.

A public key identification scheme [20] allows the possessor of a secret key to prove possession of that secret by means of an interactive protocol. Thus, in the case of an RFID deployment, the tag would "prove" that it contains a tag-specific secret to a reader and the reader is thereby assured that the tag is genuine. Only a device possessing the key could provide the necessary responses during interaction with the reader. While at first sight this might appear to be quite a specialised functionality, for instance we don't have the conventional public key services of encryption or digital signatures[1], interactive identification schemes have in fact been deployed widely. Among the family of identification schemes, however, the GPS scheme seems to allow a particularly compact implementation on the tag and this allows us to consider how an RFID tag with public key capability can open up previously unavailable application areas.

[1] Identification schemes can be converted to signature schemes in a standard way [20] though some computational advantages can be lost.

3 Using public key algorithms

It has already been noted that the supporting infrastructures for secret and public key cryptography are very different. They each have advantages and drawbacks. Generally speaking, however, secret key cryptography is more suited to closed environments where supplying the same key material to different participants in the system can be relatively straightforward. Public key cryptography is perhaps more useful in open systems where the potential users of an RFID tag cannot be identified ahead of time.

If public key technology can be enabled in cheaper RFID tags then this would allow tags to be deployed in open systems. The reader would not need to share a secret key with the tag ahead of time as would be required when using secret key cryptography to authenticate the tag. Instead, the tag's public key can be sent from the tag to the reader, the authenticity of the public key can be verified by the reader due to a signature on the public key, and the public key cryptography—in our case an identification protocol—can follow.

Over recent years there has been some considerable work on the compact implementation of a variety of public key technologies. While most work has centered around using elliptic curves (which we also use in our variant of GPS) this alternative work typically considers implementations of the signature scheme ECDSA [22]. Since the functionality offered by GPS and ECDSA are different, it is not always easy to make a fair comparison. However, while the impressive work of Batina *et al* [2] and Kumar and Paar [17] suggest that of the order of 10^5 GEs are required to support ECDSA on-the-tag, we will see in Section 6.2 that GPS requires of the order of 10^3 GEs and a correspondingly modest power consumption.

There are a variety of applications where such cryptographically-enhanced RFID tags might be of particular interest. The most familiar might be ticketing applications for entertainment or sporting events. They might also play a role in public transport applications. Higher-security situations such as smart labels for airline baggage handling, travel visas, and anti-counterfeiting labels for luxury goods, pharmaceutical products, and even engine components in the automotive or aeronautic industries might also be promising possibilities.

4 The GPS scheme and literature

An exposition of the GPS scheme, its security and that of numerous variants appears in a range of papers [8, 11, 24]. It has been widely referenced in the cryptographic literature, it is standardised within ISO/IEC 9798-5 [14], and it is listed in the final NESSIE portfolio [16]. There are many variants and optimisations of the scheme. One variant uses RSA-like moduli but in Figure 1 we illustrate the essential elements of GPS using elliptic curve operations. This would be the likely choice in a resource-constrained environment since it allows smaller keys.

Of particular practical interest are a series of optimisations designed to ease the computation and storage costs of different solutions. One important optimisation is the use of *coupons*. These are a form of pre-computation and they are described in [9]. Some storage optimisations are described in [12, 14], and a particularly useful proposal termed the *Low Hamming Weight (LHW) challenge* is described in [10]. It is the combination of coupons with the LHW challenge that yields the best performance profile for implementation in resource-constrained environments.

Throughout we will use the coupon approach [9]. As mentioned, this is effectively a form of pre-computation where part of the on-tag computation is performed ahead of time and the partial results stored on-tag. Thus we trade the need for some additional memory with a very effective way to reduce the computation requirements. At the time of manufacture, therefore, t coupons which consist of pairs of numbers (r_i, x_i) for $1 \leq i \leq t$ are computed and stored on the tag along with the tag-specific secret key s. The form of these numbers is illustrated in Figure 1, though a further optimisation that provides some storage advantages will be described in Section 5. One important observation is that even though GPS relies on number theoretic computations like other public key schemes, when we use coupons these computations need not be supported on the tag. Instead the on-tag GPS computation reduces to the simple integer[2] computation $y = r_i - (s \times c)$ where c is a challenge provided by the reader (see Figure 1). This is a regular integer computation consisting solely of a multiplication and a subtraction and we can begin to see why the GPS scheme may be well-suited to constrained environments.

With regards to previous implementations of the scheme, there already exists a preliminary investigation into implementing GPS on smart cards [11]. A detailed analysis of the costs of implementing GPS in hardware—considering computation, power, and storage—is provided in [18]. Follow-on work [19] describes novel architectures that yield a flexible performance trade-off while still retaining the essential implementation qualities required for constrained environments. The work in both [18, 19] is summarised in Section 6.2. To complement these, in the following sections we describe the development of a fully-functioning GPS prototype on an FPGA. The implementation experience for the FPGA-based implementation provides good alignment with the results obtained in [18, 19].

5 A full GPS prototype

In this section we describe the results of an FPGA prototype implementation of the GPS identification scheme.

In this implementation, the GPS secret key s and the shared parameters are chosen so that the system offers a security level commensurate with 1024-bit RSA. This is widely believed to be roughly equivalent to 80-bit symmetric key security and hence to elliptic curve implementations using a 160-bit field.

[2] It is easy to specify a variant whereby the public key has the form $v = -sP$ and the on-tag computation is $y = r_i + (s \times c)$.

Tag		**Reader**
	PARAMETERS	
Curve $\mathcal{C}$, point P		Curve $\mathcal{C}$, point P
	KEYS	
Secret $s \in_R \{0,1\}^\sigma$		
Public $V = sP$		Public $V = sP$
	COUPON PRE-COMPUTATION WITH PRNG	
For $0 \leq i \leq t-1$		
Let $r_i = \text{PRNG}_k(i)$ where $\lvert r_i \rvert = \rho$		
Set $x_i = \texttt{hash}(r_i P)$		
Store coupon x_i		
	PROTOCOL USING ON-TAG PRNG	
At time i fetch x_i	$\xrightarrow{x_i}$	
	$\xleftarrow{c}$	Pick $c \in_R \{0,1\}^\delta$
Generate $r_i = \text{PRNG}_k(i)$		
$y = r_i - (s \times c)$	$\xrightarrow{y}$	$\texttt{hash}(yP + cV) \overset{?}{=} x_i$

Fig. 1. Overview of ECGPS using coupons that are partially re-generated. The parameters ρ, δ, and σ denote three particular bit lengths offering a range of security/performance trade-offs.

Such a security level is likely to be appropriate for most near-term RFID-based applications. The elliptic curve used for the GPS implementation is referred to as *secp160r1*. It is fully specified in *Standards for Efficient Cryptography: SEC2 Recommended Elliptic Curve Parameters* [5].

To authenticate the tag public key, it is signed using ECDSA [22]. In practice the choice of signature scheme for the tag public key is somewhat orthogonal to the tag itself, though ECDSA is likely to be used since this provides compact signatures. (In the case of the prototype the signature is 320 bits.) The storage of all keys and coupons (see below) for the prototype is less than 1700 bits.

The hash function used in both the generation of the coupons and in the verification computation on the reader is SHA-1 [21]. This is a suitable choice for a prototype, but recent cryptanalytic results [26] suggest that an alternative hash function might be considered for any deployment and much will depend on the application and the associated risk analysis.

We have already mentioned that we are using a version of GPS with coupons. Full coupons have the general form (r_i, x_i), however, under the assumption that we are required to use more than a handful of coupons we will follow suggestions in ISO 9798-5 to give a storage saving. To do this we will not store the full coupon (r_i, x_i) but we will store a partial coupon x_i (see Figure 1). We will then use a compact *pseudo-random number generator* PRNG and generate each r_i once at the time of manufacture and a second time on the tag at the time they are needed. Clearly the PRNG needs to be sufficiently compact since it will be implemented on the tag and it will be used when challenged by the reader. The PRNG implemented in the prototype remains proprietary to ASK [1], but a

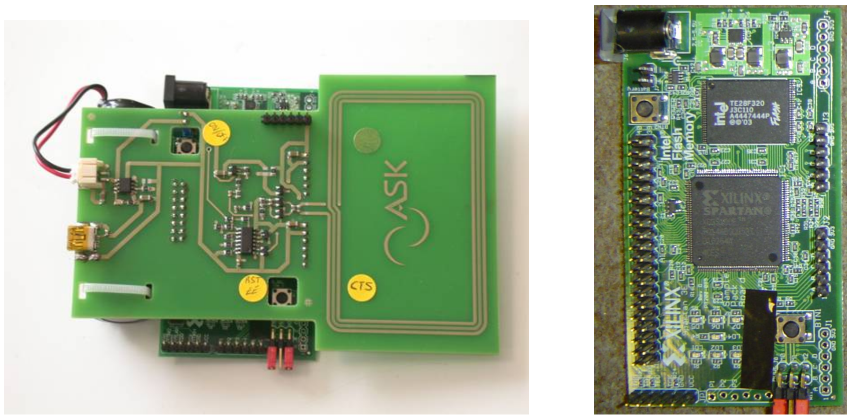

Fig. 2. The ECGPS prototype that simulates a tag.

different PRNG can easily be considered with the corresponding changes to the implementation requirements in Table 1.

The prototype uses the LHW variant [10] of GPS. This variant requires that the reader chooses challenges c of a special form. These challenges are longer than usual (a LHW challenge of 860 bits gives comparable security to a 32-bit regular challenge) but they have a very low Hamming weight. There are few ones in the challenge and they are judiciously spaced. As a result the multiplication $(s \times c)$ in the on-tag computation is effectively turned into a modest number of additions of the secret s. The result is that there is no need to implement multiplication on the tag and this allows us to further simplify an already efficient computation. Consequently, for the prototype we have the following set of bit-lengths:

$$|s| = \sigma = 160$$
$$|c| = \delta = 860$$
$$|r| = \rho = 1100$$

Note that even though the LHW challenge c looks to be rather long, it is very sparse. This means that there are a variety of compact representations for the 860-bit challenge. The compressed form of the challenge used in the prototype is only 25 bits long and the only impact on the tag is to devise a compact implementation to interpret the compression. This is easy to do. In fact, it is interesting to note that the representation of the challenge can have a very significant impact on the performance of the scheme. This is exactly the issue explored in [19] where different encoding schemes for the challenge allow us to devise a range of timing/power/space trade-offs.

At first sight the size of the variable r also appears to be rather large. However, recall that this is regenerated using the PRNG and so there are no storage implications attributed to this variable no matter how many coupons are stored.

Table 1. Space requirements for the different components in the FPGA prototype.

component	$\approx$ *GEs*	$\approx$ *fraction of total*
GPS computation	600	10%
PRNG	1000	17%
Logic + memory	1000	17%
Supporting module	3400	56%
Total	6000	

6 GPS performance in practice

The performance of the GPS scheme has been studied in some detail over the past few years. Here we describe a fully-functioning FPGA prototype, which includes the over-the-air communication protocol, and we summarise some ASIC implementation estimates for the core computation used in GPS.

6.1 The FPGA prototype

The GPS prototype of Section 5 was implemented on a Xilinx Spartan 3E FPGA. This simulates the actions of a tag and uses a 13.56 MHz radio interface for the contactless communication which conforms to ISO 14443-2 Type B and ISO 14443-3 Type B frame format [15].

Specific choices for the implementation of GPS are described in Section 5. It is interesting to consider the break-down of the space requirements for the different parts of the implementation; these are given in Table 1. Note that the item *GPS computation* refers to the computation of the equation $y = r_i - (s \times c)$ and the additional logic and memory are required to process the encoding of the challenge c and to provide working RAM.

The entire implementation required around 6000 gates of which around 34% (43% including logic and memory) were devoted to supporting the GPS identification scheme. If such a gate requirement were replicated in 0.180μm technology then it would correspond to a physical space of around 0.05mm^2. In the prototype 10 coupons were supported as a proof-of-concept. Clearly there is a direct trade-off in the space devoted to memory and the number of coupons that are stored. The optimal balance will depend on both the application and the cost of deployment.

The board was clocked at 847.5 KHz and the entire identification process, including communication and certificate verification, required around 200ms with the tag-based computation requiring 14.2ms.

In moving from an FPGA implementation to a dedicated RFID-tag implementation there are differences to consider. In particular the exact power demands for an RFID-tag implementation remain to be seen, though work in [18, 19] has considered the requirements of the core cryptographic computation within GPS (*i.e.* the computation of the tag response y) when implemented in 0.180μm technology. These are described in the next section.

6.2 ASIC estimates

In a pair of papers [18, 19] some initial analysis of the ASIC implementation requirements for the GPS identification scheme are given. The reader is referred to those papers for details, but the following trade-off points have been identified. These use exactly the same parameter sets, and the same optimisations, as are used for the FPGA-implementation described in Section 6.1.

Some performance trade-offs for GPS in $0.180\mu m$ technology [19].

area (GE)	*current (μA)*	*time (cycles)*	*frequency (kHz)*
317	0.61	1088	100
431	0.67	136	100
900	1.43	68	100
431	2.26	136	500
900	4.91	68	500

It should be noted that these are partial results. First, they are only concerned with the core on-tag operation in GPS. A full on-tag implemenation of GPS would need a PRNG as well as supporting logic, memory, and communications interface. Further, the variant used for this version of GPS is the one referred to in the NESSIE documentation [16] for which the on-tag GPS computation is given by $y = r_i + (s \times c)$ rather than $y = r_i - (s \times c)$ (with appropriate change to the form of the public key).

Nevertheless, the overall message is clear and the on-tag requirements of the GPS identification scheme are particularly modest. We anticipate that other researchers may well identify other performance profiles of interest to implementers in the future.

7 Conclusions

Everything comes at a price. But in the most extremely constrained devices additional security features command a substantial premium. In the field of symmetric cryptography there has been much recent work, and considerable success, in the implementation and design of new, and compact, primitives. By contrast, however, asymmetric cryptography has been viewed as being almost fundamentally unsuitable.

In this paper we consider the implementation of the GPS public key identification scheme. We have described the algorithm and the optimisations of most value in a fielded application, and we have given details of a prototype implementation of GPS on an FPGA. This fully functioning prototype, for which the reader software and communication interface have also been implemented, requires only 2600 GE for the cryptographic components. This work complements other implementation analysis of GPS, and suggests that—for the right application—public key technology may well be both desirable and feasible on cheaper RFID tags.

8 Acknowledgements

The authors gratefully acknowledge the work of Nicolas Pangaud from ASK and Yoann Thomas at Teamlog for their work on implementing the prototype.

References

1. ASK. Information available via `http://www.ask.fr`.
2. L. Batina, J. Guajardo, T. Kerins, N. Mentens, P. Tuyls and I. Verbauwhede. An Elliptic Curve Processor Suitable for RFID-Tags. IACR eprint. Available at http://eprint.iacr.org/2006/227, July 2006.
3. A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar, A. Poschmann, M.J.B. Robshaw, Y. Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In P. Pallier, editor, Proceedings of CHES 2007, Springer, to appear.
4. C. de Cannière and B. Preneel. Trivium. Available via `www.ecrypt.eu.org`.
5. Certicom. Standards for Efficient Cryptography: SEC2 Recommended Elliptic Curve Domain Parameters. Version 1. September 20, 2000. Available via `http://www.secg.org/`.
6. ECRYPT Network of Excellence. The Stream Cipher Project: eSTREAM. Available via `www.ecrypt.eu.org/stream`.
7. M. Feldhofer, S. Dominikus, and J. Wolkerstorfer. Strong Authentication for RFID Systems Using the AES Algorithm. In M. Joye and J.-J. Quisquater, editors, Proceedings of CHES 2004, LNCS 3156, pages 357–370, Springer Verlag, 2004.
8. M. Girault. Self-certified Public Keys. In D. Davies, editor, *Proceedings of Eurocrypt '91*, volume 547 of *Lecture Notes in Computer Science*, pages 490–497, Springer-Verlag, 1992.
9. M. Girault. Low-size Coupons for Low-cost IC Cards. In J. Domingo-Ferrer, D. Chan, and A. Watson, editors, *Proceedings of Cardis 2000*, IFIP Conference Proceedings 180, pages 39–50, Kluwer Academic Publishers, 2000.
10. M. Girault and D. Lefranc. Public Key Authentication With One (On-line) Single Addition. In M. Joye and J.J. Quisquater, editors, *Proceedings of CHES '04*, volume 3156 of *Lecture Notes in Computer Science*, pages 413–427, Springer-Verlag, 2004.
11. M. Girault, G. Poupard and J. Stern. On the Fly Authentication and Signature Schemes Based on Groups of Unknown Order. Journal of Cryptology, pages 463–488, volume 19, number 4, 2006.
12. M. Girault and J. Stern. On the Length of Cryptographic Hash-values Used in Identification Schemes. In Y. Desmedt, editor, *Proceedings of Crypto '94*, Lecture Notes in Computer Science 839, pages 202–215, Springer-Verlag, 1994.
13. M. Hell, T. Johansson and W. Meier. Grain - A Stream Cipher for Constrained Environments. Available via `www.ecrypt.eu.org`.
14. ISO/IEC. International Standard ISO/IEC 9798 Part 5: Mechanisms Using Zero-knowledge Techniques. December, 2004
15. ISO/IEC. International Standard ISO/IEC 14443 Parts 1-4: Contactless integrated circuit(s) cards. Proximity cards. October, 2003.
16. IST-1999-12324. Final Report of European Project IST-1999-12324: New European Schemes for Signatures, Integrity, and Encryption (NESSIE). Available via `https://www.cosic.esat.kuleuven.be/nessie/`.
17. S. Kumar and C. Paar. Are Standards Compliant Elliptic Curve Cryptosystems Feasible on RFID? Presentation at Workshop on RFID Security RFIDSec 06, July 2006.

18. M. McLoone and M.J.B. Robshaw. Public Key Cryptography and RFID. In M. Abe, editor, *CT-RSA2007*, volume 4377 of *Lecture Notes in Computer Science*, pages 372–384, Springer, 2007.
19. M. McLoone and M.J.B. Robshaw. New Architectures for Low-Cost Public Key Cryptography on RFID Tags. To appear in the *Proceedings of ISCAS 2007*.
20. A. Menezes, P.C. van Oorschot, and S. Vanstone. The Handbook of Applied Cryptography. CRC Press, 1996.
21. National Institute of Standards and Technology. FIPS 180-2: Secure Hash Standard. Available via `http://csrc.nist.gov/publications/fips/`.
22. National Institute of Standards and Technology. FIPS 186-2: Digital Signature Standard. Available via `http://csrc.nist.gov/publications/fips/`.
23. A. Poschmann, G. Leander, K. Schramm, and C. Paar, A Family of Light-Weight Block Ciphers Based on DES Suited for RFID Applications. In A. Biryukov, editor, *Proceedings of FSE 2007*, LNCS, Springer-Verlag, to appear.
24. G. Poupard and J. Stern. Security Analysis of a Practical "On the Fly" Authentication and Signature Generation. In K. Nyberg, editor, *Proceedings of Eurocrypt '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 422–436, Springer-Verlag, 1998.
25. R.L. Rivest, A. Shamir, and L.M. Adleman. A Method for Obtaining Digital Signatures and Public-key Cryptosystems, Communications of the ACM, pages 120–126, volume 21, number 2, 1978.
26. X. Wang, Y.L. Yin, and H. Yu. Finding Collisions in the Full SHA-1. In V. Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 17–36. Springer-Verlag, 2005.