Summary–Cryptography Group PLA Contributions

Billy Brumley

1 PLA Overview

The goal of PLA is to attach digital signatures to every packet sent over the network, allowing every hop along the route to verify the packet. This is different from, for example, IPSec, where digital signatures can only be verified once they reach the destination; PLA can prevent distributed denial of service attacks, while IPSec cannot. A PLA header is attached to IPv6 packets with fields which allow the authenticity of packets to be verified.

The goal of the Cryptography Group in the PLA project is to minimize the space requirements of the certificate and signature fields in the PLA header and at the same time maximize the speed of packet verifications. Respectively, this decreases the packet overhead and increases the throughput for PLA. There are a number of standardizes signature schemes to choose from (RSA, DSA, ECDSA, ...) with many different cryptographic settings (generic multi-precision integer arithmetic, prime field arithmetic, binary field arithmetic, ...)—thus an interesting research question for PLA. Different alternatives for public key infrastructures also influence the aforementioned choices (such as identity-based cryptography). A description of the cryptosystems used in PLA follows.

2 Description of Cryptography for PLA

The cryptographic settings and primitives for PLA were chosen in such a way to achieve minimal overhead in the packet as well as high speed verification of packet signatures. Elliptic curve cryptosystems are a natural choice for minimizing the overhead, as signatures and keys are extremely small. The level of security is based on the size of the group. In a multiplicative group of integers modulo a prime, there are sub-exponential attacks to solve the discrete log problem (namely Index Calculus methods). In elliptic curve groups, the best known attacks on the elliptic curve discrete log problem are general attacks for solving the discrete log (namely Pollard's ρ method) which has exponential running times, and thus significantly smaller keys can be used. A comparison of key sizes is given in the following table.

2.1 Elliptic Curve Cryptography for PLA

Specifically, the PLA implementation is done using the standardized Koblitz curve K-163 (main prime subgroup order $r \approx 2^{163}$) defined over a binary field, which provides $\approx 2^{80}$ bits of security. Binary field operations are fast in both

Symmetric	ECC	DSA/RSA
80	163	1024
112	233	2048
128	283	3072
192	409	7680
256	571	15360

Table 1: Comparable key sizes (in bits).

hardware and software. Signatures are 326 bits, compressed public keys 164 bits, and private keys 163 bits. The Nyberg-Rueppel signature scheme is used to sign packets. A description follows.

2.1.1 Digital Signatures

Elliptic curve E is chosen with base point generator G of prime order r where $r \mid \#E$. Alice generates a private key s and public key W by computing $W = sG$ where $s \in_R \mathbb{Z}_r^*$. This requires one scalar multiplication involving a fixed point G. To generate a signature (c, d) on a message m, Alice calculates

$$c = [uG]_x + \mathrm{H}(m) \pmod r \text{ where } u \in_R \mathbb{Z}_r^*$$
$$d = u - sc \pmod r$$

where $[P]_x$ denotes the x-coordinate of the point P converted to an integer and H is a collision-resistant hash function. To verify the signature (c, d) on the message m, Bob checks that $\mathrm{H}(m) = c - [dG + cW]_x \pmod r$.

Instead of a traditional PKI approach, identity-based self-certified keys (implicit certificates) are used. This eliminates the need for an explicit trusted third party signature on users' public keys, thus reducing the overhead in the packet. A comparison of the two approaches is given in the following table, where r is the group order, q the field size, and ESM the elliptic scalar multiplication operation (a good measure of the computational time). The certificate therein is a minimal one, containing only a TTP signature on the client's public key—in practice more information is needed (validity period, etc.), but only the cryptographic storage requirements are measured here.

Certificate-Based PKI	PLA	Self-Certified	PLA
signature ($2r$)	326	signature ($2r$)	326
public key ($q+1$)	164	self-certified public key ($q+1$)	164
TTP signature on public key ($2r$)	326	-	0
verify public key	2 ESMs	extract public key	1 ESM
verify signature	2 ESMs	verify signature	2 ESMs
Packet Total (bits)	816		490
Computation (ESMs, Sign/Verify)	1/4		1/3

Table 2: Storage and computation requirements.

One could argue that to save computation time, in traditional certificate-based PKI the certificate need only be verified once, then a hash stored (the same can be said of implicit certificates and extracting a client's public key). However, this requires extra storage and time and as the number of trusted

third parties grows, this is not convenient. Additionally, the 4 ESMs required can be reduced to 2 simultaneous ESMs and the 3 ESMs to 1 simultaneous ESM (summary given later)—hence one could argue the exact opposite when using implicit certificates, that such storage and hashing will in fact not reduce the computation time, but simply increase the complexity.

2.1.2 Implicit Certificates

PLA uses a Nyberg-Rueppel based scheme and requires that users have a unique public identity assigned to them. A description follows.

Elliptic curve E is chosen with base point generator G of prime order r where $r \mid \#E$. The Trusted Third Party (TTP) generates a domain private key $s_D \in_R Z_r^*$ and domain public key $W_D = s_D G$. Generating client implicit certificates is done using the following protocol in such a way that only the client knows the private key—TTP does not learn it. To generate a private key on user Alice's identity ID_A, Alice calculates $k_A G$ where $k_A \in_R \mathbb{Z}_r^*$ and sends $k_A G$ to TTP. TTP calculates[1]

$$\begin{aligned}(\bar{r}_A, b_A) &= \text{COMPRESS}(k_A G + k_T G) \text{ , where } k_T \in_R \mathbb{Z}_r^* \\ r_A &= \bar{r}_A + \text{H}(ID_A) \\ \bar{s}_A &= k_T - r_A s_D \pmod r\end{aligned}$$

where COMPRESS is the point compression function, yielding the x-coordinate of uG and the compression bit b_A; TTP sends $(r_A, b_A), \bar{s}_A$ to Alice. Alice calculates the private key $s_A = k_A + \bar{s}_A \pmod r$. Alice's public key is $W_A = s_A G$, extracted given the implicit certificate (r_A, b_A) and Alice's identity ID_A by computing $W_A = \text{DECOMPRESS}(r_A - \text{H}(ID_A), b_A) - r_A W_D$ where DECOMPRESS is the point decompression function given an x-coordinate and compression bit b. This requires one scalar multiplication and one point addition.

3 Summary of Contributions

The combination of self-certified keys and Koblitz curves has led to many interesting contributions for the cryptography group. These are summarized below; for details, see the corresponding referenced papers.

Fast signature verification. In [Bru06b] we show how to accomplish self-certified public key extraction and signature verification simultaneously using only one three-term scalar multiplication. This means that verifying signatures is very fast and there is no need to store the previously extracted public keys and hashes. In [Bru06c] we show how to reduce the joint weight of an arbitrary number of scalars when using Koblitz curves. For PLA, this reduces the total number of elliptic curve operations needed, thus the signature verification is faster. An FPGA implementation was done independently by the hardware group [JFS07] using these ideas.

[1] Note that the values $(r_A, \bar{s}_A)$ are simply a Nyberg Rueppel signature on the message $m = ID_A$.

Software implementation. A software implementation was also done using these ideas as part of a Master's thesis [Bru06a]. The cryptographic implementation done in C was later integrated into the PLA software package which handles the verification of packets. The software was written to be compatible with a future hardware implementation; this particular cryptographic setting is not common in software, so the implementation has merit.

Preventing impersonation. In [BN07] we prove some important differential properties of elliptic curve mappings. For PLA, this means that when running the self-certified key generation protocol between a user and the trusted third party, a proof-of-knowledge need not be performed to prevent impersonation and one roundtrip communication can be eliminated—a significant savings.

Fast signature generation. In [BJ07] we provide a new method for computing the integer equivalent of random Frobenius expansions. This can significantly speed up signature generation, as no costly modular reduction needs to be performed and the hardware implementation takes much less area. Joint work with the hardware group.

References

[BJ07] Billy Bob Brumley and Kimmo Järvinen. Koblitz curves and integer equivalents of Frobenius expansions. In *Selected Areas in Cryptography, 14th International Workshop—SAC '07*, volume 4876 of *Lecture Notes in Computer Science*, pages 126–137. Springer-Verlag, 2007.

[BN07] Billy Bob Brumley and Kaisa Nyberg. Differential properties of elliptic curves and blind signatures. In *Information Security, 10th International Conference—ISC '07*, volume 4779 of *Lecture Notes in Computer Science*, pages 376–389. Springer-Verlag, 2007.

[Bru06a] Billy Bob Brumley. Efficient elliptic curve algorithms for compact digital signatures. Master's thesis, Helsinki University of Technology, Department of Computer Science and Engineering, Laboratory for Theoretical Computer Science, November 2006.

[Bru06b] Billy Bob Brumley. Efficient three-term simultaneous elliptic scalar multiplication with applications. In Viiveke Fåk, editor, *Proceedings of the 11th Nordic Workshop on Secure IT Systems—NordSec '06*, pages 105–116, Linköping, Sweden, October 2006.

[Bru06c] Billy Bob Brumley. Left-to-right signed-bit τ-adic representations of n integers (short paper). In *Information and Communications Security, 8th International Conference—ICICS '06*, volume 4307 of *Lecture Notes in Computer Science*, pages 469–478. Springer-Verlag, 2006.

[JFS07] Kimmo Järvinen, Juha Forsten, and Jorma Skyttä. FPGA design of self-certified signature verification on Koblitz curves. In *Cryptographic Hardware and Embedded Systems—CHES '07*, volume 4727 of *Lecture Notes in Computer Science*, pages 256–271. Springer-Verlag, 2007.