Implementing Direct Anonymous Attestation for the TPM Emulator Project

Heiko Stamer

University of Kassel
Department of Mathematics/Computer Science
Heinrich-Plett-Straße 40, D-34132 Kassel

stamer@theory.informatik.uni-kassel.de

76F7 3011 329D 27DB 8D7C 3F97 4F58 4EB8 FB2B E14F

4. Krypto-Tag, May 2006, Ruhr-Uni Bochum

UNIKASSEL VERSITÄT

"THE TRUSTED COMPUTING GROUP (TCG) IS AN INDUSTRY ORGANIZATION THAT AIMS TO DEVELOP AND PROMOTE OPEN, VENDOR-NEUTRAL STANDARD SPECIFICATIONS FOR TRUSTED COMPUTING BUILDING BLOCKS AND SOFTWARE INTERFACES ACROSS MULTIPLE PLATFORMS." [TCG]

- Hardware: Specification of the Trusted Platform Module (TPM)
  - Version 1.1b (February 2002), Version 1.2 (Revision 62: October 2003, Revision 85: February 2005, Revision 94: March 2006)
- Mainboard/Firmware/BIOS: Platform Specific Specifications
  - PC Client TPM Interface Specification (TIS), Version 1.2
  - Implementation Specification for Conventional BIOS, Version 1.2
- Software: Specification of the TCG Software Stack (TSS)
  - `TSS ::= (TCG | TPM) Software (Stack | Specification)`
  - Version 1.1 (September 2003), Version 1.2 (January 2006)
  - News: Auditing, delegation, monotonic counters, DAA, ...
- Network: Specification of the Trusted Network Connect (TNC)

Main Functionality of the TPM:

- "Cryptographic co-processor" (RNG, SHA-1, HMAC, RSA)
- Hardware protected storage for cryptographic keys
- Measurement of the platform configuration (PCR)
  - Application: Secure Boot and Remote Attestation
- Sealed Storage: Decryption keys are tied to a PCR value

Remote Attestation: Certify a platform characteristic (e.g. the current software configuration by means of PCR values) to a remote party.

Application:

- System integrity check (to detect corrupted software, e.g. root kits)
- Enforcement of a corporate-wide software stack
- Digital rights management (DRM), pay-per-use services
- Product activation, tethering, and customization
- Vendor lock-in, forced upgrades and downgrades, ...

Previous Solution (Trusted Third Party): introduced by TPM Specification, Version 1.1b

1. Owner initiates the creation of an attestation identity key (AIK)
   - AIKs are special-purpose signature keys (non-migratable, RSA 2048 bit, $e = 2^{16} + 1$) generated and protected by a TPM
2. Privacy-CA certifies this AIK, if the platform is able to show its conformance with a policy (e.g. valid EK/platform credentials)
3. AIK and the obtained AIK-certificate are used by the platform to perform a desired remote attestation (i.e. sign PCR values)

Disadvantages:

- Different attestations are linkable, if the same AIK is used multiple times. Thus owners should always create fresh AIKs.
- The Privacy-CA is a very sensitive entity. Therefore it must be carefully protected and maintained to guarantee security.
- The Privacy-CA must be highly available, because it is involved in every attestation. (but this contradicts the point above)

Brickell, Camenisch, and Chen [BCC04]: Direct Anonymous Attestation

- Combine ideas from group signature schemes with the efficient Camenisch-Lysyanskaya anonymous credential system [CL01, CL02]
- "GROUP SIGNATURE SCHEME WITH~~OUT~~ ANONYMITY REVOCATION"
- DAA entities:

  Host/TPM: TC-platform consists of a host and a trusted observer (TPM)
  Issuer issues a DAA-certificate, if TC-platform possess a valid EK (policy)
  Verifier provides a service, if DAA-signature is valid w.r.t. desired message

- DAA sub-protocols:

  DAA-Join: TPM chooses a secret f and performs a two-party protocol with an issuer to obtain a secret DAA-certificate $A_I(f)$ (i.e. a CL-signature on f).
  DAA-Sign: TPM signs an attestation message m (e.g. hash value of an AIK). The appended non-interactive zero-knowledge proof of knowledge shows that the corresponding DAA-signature $\sigma_I(f, m)$ is valid w.r.t. f, m, and I (issuer key).

- Issuer and verifier are able to detect "broken TPMs" (Rogue Tagging)
- Unforgeability of DAA-certificates relies on Strong RSA Assumption
- Unlinkability of DAA-certificates/signatures relies on DDH Assumption

Advantages:

- DAA-certificates need to be issued only once (no bottleneck)
- Issuer and verifier cannot link DAA-certificates and DAA-signatures, even if they are the same entity (“repairs the broken business model”)
- Anonymity degradation is possible (named base vs. random base)
  - Detect and exclude malicious TPMs (e.g. black list)
  - Perform frequency analysis (DoS attack on issuer/verifier)

1st Extension: (does not break the current TPM Specification)

- Better privacy by combining DAA with a Privacy-CA which issues “one-time” DAA-certificates (Camenisch, ESORICS 2004)

2nd Extension: (breaks the current TPM Specification)

- Ensure anonymity on untrusted TPMs (Camenisch, unpublished)

DAA in TCG Specifications: introduced by TPM and TSS Specification, Version 1.2

TPM 1.2 TPM part of DAA-Join and DAA-Sign (structures and commands)

TSS 1.2 Host part of DAA-Join and DAA-Sign, NIZK verification (issuer key), supplemental functions for issuer and verifier, ...

"Additional Features" Introduced by TCG Specifications:

TSS 1.2 Arbitrary attributes (e.g. expiration date) for DAA-certificates

TSS 1.2 Optional anonymity revocation based on verifiable encryption (cf. Camenisch and Shoup, CRYPTO 2003)

Technical Problems:

- DAA-Join resp. DAA-Sign are highly resource intensive protocols (TPM: 11 resp. 7 modular exponentiations with large exponents)

  → `TPM_DAA_Join` and `TPM_DAA_Sign` are executed in atomic stages; in-between they may interruptible by other commands (save context)
- DAA-Join must not run arbitrarily interleaved (restriction handled by TSS)
- Protection against timing attacks? (e.g. exp. in stage 4 of `TPM_DAA_Join`)

Mario Strasser: Software-based TPM Emulator for Linux [St04]

`https://developer.berlios.de/projects/tpm-emulator/`

Goal: Create a fully working Trusted Platform Module emulator according to TCG Specification, Version 1.2 (Revision 62 → 94)

Application: Explore TPMs for educational/experimental purposes

Current State: Release 0.3 (January, 2006), GNU GPL v2

- Kernel module `tpm_emulator.ko` (provides char. device `/dev/tpm`)
- Currently, 80 out of 120 TPM commands are implemented
  (admin startup, admin testing, admin opt-in, admin ownership, auditing, storage functions, cryptographic functions, endorsement key handling, identity creation, integrity collection and reporting, authorization sessions, session management, eviction, timing ticks, transport sessions, monotonic counter, DAA, deprecated commands)
- Not yet/only partially implemented: capability, migration, maintenance, identity activation, delegation, NV storage
- Packetized for Gentoo Linux (`$ emerge tpm-emulator`)

Prerequisites:

- Linux Kernel 2.6.x, GNU Compiler Collection, ...
- GNU Multiple Precision Arithmetic Library (libGMP)

Roadmap/TODO:

1. Conformance with Revision 94 of the TPM Specification 1.2
2. Obtain better portability (kernel space vs. user space)

   1st Problem: Kernel stack size is very limited
   (architecture dependent, e.g. 4K resp. 8K on x86)

   2nd Problem: Persistent storage is needed to save the state

   Possible Solution:

   - Dummy “hardware interface” in the common TPM device driver
   - TPM emulator serves only as user space daemon
3. Implementation of all mandatory commands (v1.2 rev 94)
4. Adding optional commands and algorithms (e.g. AES)

- Approximately 3850 lines of code (including 600 lines of comments)
- Implementation time: $\approx$ 9 days (3–4 hours per day)
- Testing time: $\approx$ 6 weeks (IBM DAA Test Suite [Zi05])

Requirements:

- Kernel stack size of at least 8K (libGMP calls, large structures)

Implementation Problems:

- Missing kernel debugger (e.g. to detect call paths with stack overflows)
- Alignment of large integers in combination with hashing
- TPM specification contains many typographical errors:
  - "`TPM computes a TPM-specific secret f0 (104-bit) = f mod 2104`" (Part 1, rev 94)
  - "`#define DAA_SIZE_r3 158`" (Part 2, rev 62) "`#define DAA_SIZE_r3 168`" (Part 2, rev 85)
  - "`obtain DAA_SIZE_NT bits from RNG`" (20 bits vs. 20 bytes) (Part 3, rev 94)

- The implementation works well and was carefully tested with the IBM DAA Test Suite [Zi05] (thanks to Roger Zimmermann for his support).
- TPM and TSS 1.2 specifications (even the current revision 94) contain many typographical errors, often change data structures, and thus are difficult to implement resp. hard to keep up to date.
- Open-source TSS: Kent Yoder (IBM, TrouSerS Project [Yo06])
  - "RIGHT NOW WE HAVE TWO PEOPLE IMPLEMENTING DAA."
  - "WE'RE PLANNING ON INTEGRATING THE IMPLEMENTATION SOME TIME IN THE NEXT COUPLE MONTHS."
- Contributions to TPM-Emulator Project [St06] are very welcome!

Thank you!

[TCG] Trusted Computing Group.
Web pages. https://www.trustedcomputinggroup.org/

[BCC04] Ernie Brickell, Jan Camenisch, and Liqun Chen.
Direct Anonymous Attestation.
Proceedings of 11th ACM Conference on Computer and Communications Security, ACM Press, 2004.

[CL01] Jan Camenisch and Anna Lysyanskaya.
Efficient Non-transferable Anonymous Multi-show Credential System with Optional Anonymity Revocation.
Proceedings EUROCRYPT 2001, LNCS 2045, 2001.

[CL02] Jan Camenisch and Anna Lysyanskaya.
A Signature Scheme with Efficient Protocols.
Proceedings of 3rd Conference on Security in Communication Networks, LNCS 2576, 2002.

[St04] Mario Strasser.
Software-based TPM Emulator for Linux.
Semester Thesis, ETH Zurich, 2004.

[Zi05] Roger Zimmermann, et al.
IBM Direct Anonymous Attestation Tools – TPM Test Suite.
Release 1.2.20, 2005. http://www.alphaworks.ibm.com/tech/daa/

[St06] Mario Strasser, et al.
TPM-Emulator Project.
Release 0.3, 2006. https://developer.berlios.de/projects/tpm-emulator/

[Yo06] Kent Yoder, et al.
TrouSerS Project.
Release 0.2.6, 2006. http://trousers.sourceforge.net/