Implementing the PKCS#1 v1.5 Signature Scheme with provably secure parameters.

Project Plan

Jude Asare

IY4500 - Information Security MSci Project

Supervised By: Saqib Kakvi

Information Security Group

/

Department of Computer Science

Royal Holloway, University of London

1 Abstract

The PKCS#1 v1.5 digital signature scheme has been widely utilised in protocols such as SSH, DNSSEC, IKE, and most prominently, in TLS up to version 1.2. Since its inception in 1998 [1] it has played a pivotal role in the landscape of digital security comprising the most widely used digital signature scheme in practice. The scheme, renowned for its straightforwardness and expedited verification capabilities, has seen persistent integration across diverse cryptographic systems.

Nevertheless, amidst its widespread acceptance, it's been marred by several challenges. Among these are the targeting of latent vulnerabilities [2]–[12] inherent to its associated encryption paradigm [1] which aided in revealing potential weaknesses in the signature scheme itself ([13], [14], [15]–[20]) and a glaring absence of a rigorous security proof that validates its resilience. Even though alternatives like RSA-PSS offer provable security ([21], [22]), they come with inherent problem, such as the introduction of randomness and a hike in computational complexity. These issues have created reservations for its wholesale adoption in place of PKCS#1 v1.5 e.g., RSA-PSS and was only upgraded to a requirement for new applications in PKCS#1 v2.2 [23] long after initially being suggested as the replacement for PKCS#1 v1.5 in PKCS#1 v2.1 [24]. Imperative requirements of backward compatibility and interoperability have been the main detractors of the aversion to replacing PKCS#1 v1.5 and while RSA-PSS is now required in new applications, they entail retaining PKCS#1 v1.5 in some form at the very least, as a preferable choice.

A significant breakthrough came in 2018 when Jager, Kakvi, and May [25] provided a security proof for the PKCS#1 v1.5 Signature scheme building on the work of Coron [26]. Although still requiring the adoption of larger cryptographic parameters deviating slightly from standard use, their methods were flexible enough to show instantiations in practice such that the improved proofs apply. Benefits not limited to PKCS#1 v1.5, their work also offered insights enabling the proof to apply to other deterministic RSA signature schemes, with similar construction patterns including ISO/IEC 9796-2 and ANSI X9.31 schemes.

Guided by this revelation, this project seeks to concretely implement these signature schemes, with a primary emphasis on the PKCS#1 v1.5 signature scheme, using the aforementioned provably secure parameters. This project aims not only to showcase its practicality, but also as a primary objective dissect the computational burdens these parameters introduce into deterministic RSA schemes. This objective is supplemented by supporting objectives to produce algorithms that facilitate its implementation with both provably secure parameters and separately standard parameters. From there the aim is to produce an all-incorporative user-interfaced benchmarking program to explore aforementioned overhead across standards.

2 Summarised References

*See references section at the end of document for complete reference list.

Reference	Description of relevance
1	RFC 2313 - PKCS#1 v1.5 Original introduction and documentation of the PKCS#1 v1.5 encryption and digital signature scheme.
2	Bleichenbacher's padding oracle attacks on PKCS#1 v1.5 encryption scheme
3-12	Follow ups and variants of Bleichenbacher's padding oracle on PKCS#1 v1.5 encryption scheme

13	Original introduction and documentation of Bleichenbacher's attack on vulnerable implementations of signature verification for PKCS#1 v1.5 signature schemes.
14	A follow-up analysis of the attack in [13] under shorter moduli.
15-20	A collection of reports and articles detailing real-world specific implementations of the PKCS#1 v1.5 signature scheme being targeted by Bleichenbacher's attack.
21	Security proof for the original RSA-PSS.
22	Security proof for the standardised variant of RSA-PSS adapted from the Security proof for the original RSA-PSS .
23	RFC 8017 - PKCS#1 v2.2 Formal documentation detailing the requirement of RSA-PSS as a replacement for PKCS#1 v1.5 Signature scheme in new applications.
24	RFC 3447 - PKCS#1 v2.1 Formal documentation detailing the suggestion of RSA-PSS as a replacement for PKCS#1 v1.5 Signature scheme in new applications.
25	Main related paper of the whole project: security proof for the implementation PKCS#1 v1.5 Signature scheme to be implemented as part of the project for examination of overhead introduced by using provably secure parameters.
26	Security proof applicable to the Rabin-Williams variant of RSASSAPKCS1-v1.5

3 Timeline

During the first term my plan is to split work dually between research/report writing and implementation of appropriate algorithms and/or programs that follows a full implementation life cycle based on standard software engineering principles (Test-driven Development etc). By strategically dividing my efforts between these two aspects of the project, I intend to maintain a balanced and efficient workflow that maximises productivity and ensures that both the theoretical and practical dimensions of the work progress harmoniously.

The split will be between background theory in relation to provable security and a basic implementation of the PKCS#1 v1.5 signature scheme. This POC program may also include the other deterministic signature schemes but PKCS#1 v1.5 signature scheme will be prioritised.
This will thereby provide solid foundation and smooth transition into subsequent project phase in term two which will be mostly on software engineering. Although later weeks will provide ample time to revisit and improve theory with new insights based on a greater understanding.

During Term two my plan will be to extend implementation of the signature schemes to work for provably secure parameters and then subsequently the main deliverable of benchmarking program that incorporates all prior algorithms to compare the scheme with provably secure parameters and standard parameters. The remaining time will be used for analysis and assessment of results derived from the benchmarking program.

Each of the constituent software development phases are fluid and adaptable, I do not plan to stick to them rigidly and may revisit earlier stages at various points, where I may have mis-estimated certain tasks, where feedback informs or at review reflection points.

3.1 Term 1

Week	Aims
1	summarised literature review on related proof papers Theory: Background on digital signature Schemes (generalised, RSA hash-and-sign signatures, Message Recovery, appendix).
2	Theory: finish literature review, and start conceptualisation of Schemes derived from the three main standards (e.g., PKCS#1 V1.5, ISO/IEC 9796-2 Scheme (2010), ANSI X9.31 rDSA). Theory: Background/Research on definition Provable Security.
3	Theory: Background/Research on motivation for use of Provable Security (in context). Theory: Background/Research on concepts required for the proof of signature scheme
4	Theory: Delve into the precise technical details of RSA assumption proofs relevant for the implementation of the signature scheme. SE: Enumerate requirements and analysis for PKCS#1 V1.5 Signature Scheme POC program
5	Theory: Delve into the precise technical details of Phi hiding proofs relevant for the implementation of the signature scheme. SE: Conceptualisation and Implementation of RSA key generation
6	SE: Design phase for PKCS#1 V1.5 Signature Scheme POC program. Theory: Revisit and Improve Background sections on concepts required for the proof of signature scheme

7	□ SE: Begin implementation phase for PKCS#1 V1.5 Signature Scheme POC program. □ SE: Design phase for secondary schemes: ANSI X9.31 rDSA scheme and ISO/IEC 9796-2 Scheme
8	□ SE: Continue implementation phase for PKCS#1 V1.5 Signature Scheme POC program. □ SE: Implementation phase for ANSI X9.31 rDSA scheme and ISO/IEC 9796-Scheme
9	□ SE: Integrate ANSI X9.31 rDSA scheme and ISO/IEC 9796-Scheme with PKCS#1 V1.5 Signature Scheme POC program. □ SE: Begin testing phase for integrated POC program
10	□ Continue testing and finalising the integrated POC program. □ Prepare for interim report and presentation
11	Prepare for interim report and presentation

3.2 Term 2

Week	Aims
1	□ SE: Enumerate requirements and analysis: Extension of POC program to work with provably secure parameters. □ SE: Design phase for Signature Schemes with provably secure parameters
2	□ SE Implementation of changes to enable POC to work with provably secure parameters
3	□ SE: Enumerate requirements and analysis for benchmarking program □ SE: Start design Phase for benchmarking program
4	□ SE: Complete design Phase for benchmarking program □ SE: Implementation Phase for benchmarking program
5	SE: Implementation Phase for benchmarking program
6	□ SE: Implementation (and testing) Phase for benchmarking program □ Begin work on generating results data set.
7	□ SE: Complete Implementation (and testing) Phase for benchmarking program □ Finish work on generating results data set. □ Analysis and Evaluation of results to examine overhead introduced by using provably secure parameters
8	Analysis and Evaluation of results to examine overhead introduced by using provably secure parameters.
9	□ Analysis and Evaluation of results to examine overhead introduced by using provably secure parameters. □ Appropriate conclusion
10	Prepare for final report and viva.
11	Prepare for final report and viva.

4 Risks and Mitigations

In general terms risk is the uncertainty on objectives. Not limited to my project , eliminating risk fully is not possible, so I will need manage the risk that comes with the project. This is a continuous process which I will have to engage with throughout but starts with this initial analysis that I will undertake. This section entails a discussion of notable risks with accompanying descriptions and corresponding measures I will attempt to mitigate the risks.

4.1 Complexity of Proofs and Implementation

The project includes a substantial theoretical component that underlies the implementation of the signature programs I will be developing. It is therefore possible that misunderstanding or improperly implementing a proof algorithm could happen resulting in a vulnerability ridden program. To mitigate the risk of this occurring I have planned a significant amount of early project time into understanding and researching related theory i.e., the first five weeks of term. The development of code following appropriate software engineering practices does not start until halfway through term. This is to give as much time as reasonably possible into obtaining a sound theoretical understanding.
As general approach I plan to regularly review literature throughout term and consult with experts (i.e., supervisor) which I have already started.

4.2 Time Management

It is possible that I may have underestimated the time required to complete some tasks or particular phases. The scope of the project is vast enough that overcommitting to specific theoretical pursuits could lead to last-minute rushes or incompletion both in some intermediary tasks or alternatively if this effect cumulatively builds the later tasks which directly relate to implementation of a main deliverable.
To mitigate this with my project plain I have attempted to constrain tasks sufficiently to specific deliverables all forming subtasks that contribute to a higher-level task. Additionally, this informed my thinking for categorising the weekly deliverables in distinct categories of either theory or software engineering. The aim is to take a balanced approach. Part of this is to have review reflection points to regularly review progress such as mid-November, a deadline I have agreed for a draft report and d by following the agile software engineering methodology.

4.3 Scope Creep /Tool dependencies

It is possible that the project could dependend on specific software or tools that might get outdated, have bugs, or become incompatible. This may also expand the project scope beyond initial expectations, leading to potential delays or incomplete objectives. At one point I had considered building a big number library in term one to accompany the proof-of-concept signature program. However, Efficient implementations often require algorithmic insights that are non-trivial such as the Karatsuba for multiplication which is more efficient but also more difficult to implement correctly. Given the complexity and the number of operations, there is a high likelihood of subtle bugs, which can have severe consequences especially if the library is used in security-sensitive applications. This only scratches the service of direct potential problems, let alone balancing the development of this and understanding the theory for the main project deliverables. To mitigate this, I decided to avoid this risk and stick with Java's tried and tested big number library which is widely adopted and well-supported. This adheres to the strict attitude about not extending the scope unless absolutely necessary. It also helps that my plan has clearly defined project objectives from the outset.

4.4 Integration challenges

It is possible that I encounter difficulty in integrating the other RSA signature schemes (ANSI and ISO/IEC 9796-2) into the single Proof-of-Concept (POC) program which prioritises the PKCS#1 v1.5 signature scheme. To mitigate this, I will try to ensure modularity in the codebase to simplify integration and only attempt the integration of the code when I have first considered the multiple different aspects of the software development process.

4.5 Hardware fault

I will be conducting the project almost exclusively, but particularly in relation to the code on a sole device. Although unlikely it is entirely possible that my device could fail or due to its portable nature, simply be misplaced for reasons out of my control.
To mitigate this, I plan to make my code as maintainable as possible by using a build tool compatible across systems in Maven to manage the code and its structure as well as any dependencies it may have. Additionally, I will keep the code and software engineering artefacts under version control with the personal Gitlab repository that the college provides. Any reports I write I will periodically commit to this repository and in between these periods at point of immediacy the reports will be written using OneDrive which automatically saves on every modification. Overall, I will be making use of the cloud so that my project is not tied to a single point of failure.

4.6 Disproportionate report-to-code ratio

The code and reports are interdependent. So skewed efforts to either could negatively impact the project. Focussing too much on code may mean I will not build up sufficient understanding of theory to actually understand what I want to implement in relation to proof ideas required for the signature scheme. Whereas focussing overly on the reports/theory may leave insufficient time for the incorporation of appropriate software development process to the detriment of the quality and maintainability of code.
To mitigate as discussed during the timeline section I have planned to split work dually between research/report writing and implementation of appropriate algorithms and/or programs that follows a full implementation life cycle based on standard software engineering principles. This consists of focussing first on theory to build a necessary understanding before gradually moving onto the development of code in the second half of term.

4.7 Benchmarking variability

It is possible that results from benchmarking may vary due to uncontrollable external factors, leading to inconsistent or non-reproducible outcomes. To mitigate this, I will ensure a controlled testing environment through being consistent with the same device and state of system for all benchmarking runs.

5 Risks Table

- Probability can take the values **Very Low** (The event is extremely unlikely to occur during the project's timeline), **Low, Moderate, High, Very High** (The event is almost certain to occur).
- Impact can take the values **Very Low** (The repercussions of the event on the project are negligible or minimal.), **Low, Moderate, High, Very High** (The event would critically compromise the project's objectives)
- Priority can take the values **Very Low** (minimal repercussions), **Low, Moderate, High, Critical** (severe repercussions).

Risk	Probability	Impact	Priority
Complexity of Proofs and Implementation	Moderate	Very high	**Critical**: a misunderstanding or improper implementation could fundamentally undermine the project's objectives:
Time Management	High	High	**High**: If not properly managed, it could jeopardise the success of the project.
Scope Creep /Tool dependencies	Moderate	Moderate	**Moderate**: Project extension to other deterministic schemes may lead to unforeseen complications, but there's already some control in place to manage this risk.
Integration challenges	High	Low	**Moderate**: Integrating multiple RSA signature schemes into a single PoC program, it's reasonable to anticipate potential integration challenges. Can lead to delays but might not necessarily compromise the foundational aspects of the project which emphasises PKCS.
Hardware fault	Very Low	Low	**Very low**: Modern hardware is generally reliable. Even if it occurs, mitigation strategies are in place (version control).
Disproportionate report-to-code ratio	Moderate	High	**High** Balancing between theoretical work (reports) and practical implementation (code). Given the dual nature of your project, the risk

			of skewing too much towards one end is real
Benchmarking variability	Low	Very High	**Moderate**: External factors are possible, but with the mitigation of a consistent testing environment, it's less likely.

Acronyms

SE Software Engineering.

TDD Test-driven Development.

RSA Rivest–Shamir–Adleman.

PSS Probabilistic signature scheme.

PKCS Public Key Cryptography Standards

ISO/IEC International Organization for Standardisation/International Electrotechnical Commission

ANSI American National Standards Institute

IKE Internet Key Exchange.

DNSSEC Domain Name System Security Extensions.

SSH Secure Shell Protocol.

References

[1] B. Kaliski, PKCS #1: RSA Encryption Version 1.5, RFC 2313, Mar. 1998. doi: 10.17487/RFC2313. [Online]. Available: https://www.rfc-editor.org/info/rfc2313.

[2] D. Bleichenbacher, "Chosen ciphertext attacks against protocols based on the rsa encryption standard pkcs# 1," in Advances in Cryptology—CRYPTO'98: 18th Annual International Cryptology Conference Santa Barbara, California, USA August 23–27, 1998 Proceedings 18, Springer, 1998, pp. 1-12.

[3] D. Coppersmith, M. Franklin, J. Patarin, and M. Reiter, "Low-exponent rsa with related messages," in International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 1996, pp. 1–9.

[4] J.-S. Coron, M. Joye, D. Naccache, and P. Paillier, "New attacks on pkcs# 1 v1. 5 encryption," in International conference on the theory and applications of cryptographic techniques, Springer, 2000, pp. 369–381.

[5] V. Kl´ıma, O. Pokorn´y, and T. Rosa, "Attacking rsa-based sessions in ssl/tls," in Cryptographic Hardware and Embedded Systems - CHES 2003, C. D. Walter, C¸. K. Ko¸c, and C. Paar, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 426–440.

[6] J. P. Degabriele, A. Lehmann, K. G. Paterson, N. P. Smart, and M. Strefler, "On the joint security of encryption and signature in emv," in Topics in Cryptology–CT-RSA 2012: The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27–March 2, 2012. Proceedings, Springer, 2012, pp. 116–135.

[7] R. Bardou, R. Focardi, Y. Kawamoto, L. Simionato, G. Steel, and J.-K. Tsay, "Efficient padding oracle attacks on cryptographic hardware," in Annual Cryptology Conference, Springer, 2012, pp. 608–625.

[8] C. Meyer, J. Somorovsky, E. Weiss, J. Schwenk, S. Schinzel, and E. Tews, "Revisiting { ssl/tls } implementations: New bleichenbacher side channels and attacks," in 23rd USENIX Security Symposium (USENIX Security 14), 2014, pp. 733–748.

[9] Y. Zhang, A. Juels, M. K. Reiter, and T. Ristenpart, "Cross-tenant side-channel attacks in paas clouds," in Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, 2014, pp. 990–1003.

[10] T. Jager, J. Schwenk, and J. Somorovsky, "On the security of tls 1.3 and quic against weaknesses in pkcs# 1 v1. 5 encryption," in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015, pp. 1185–1196.

[11] T. Jager, J. Schwenk, and J. Somorovsky, "Practical invalid curve attacks on tlsecdh," in Computer Security–ESORICS 2015: 20th European Symposium on Research in Computer Security, Vienna, Austria, September 21-25, 2015, Proceedings, Part I 20, Springer, 2015, pp. 407–425.

[12] H. Boock, J. Somorovsky, and C. Young, "Return of { bleichenbacher's } oracle threat," in 27th USENIX Security Symposium (USENIX Security 18), 2018, pp. 817–849.

[13] H. Finney, “Bleichenbacher’s rsa signature forgery based on implementation error,” http://www.imc. org/ietf-openpgp/mail-archive/msg14307. html, 2006.

[14] U. K¨uhn, A. Pyshkin, E. Tews, and R.-P. Weinmann, “Variants of bleichenbacher’s lowexponent attack on pkcs# 1 rsa signatures,” SICHERHEIT 2008–Sicherheit, Schutz und Zuverl¨assigkeit. Beitr¨age der 4. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft f¨ur Informatik eV (GI), 2008.

[15] MITRE Corporation. “Cve-2006-4790.” (2006), [Online]. Available: https :/ /cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-4790.

16] MITRE Corporation. “Cve-2006-4340.” (2006), [Online]. Available: https : / / cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-4340.

[17] Bugzilla. “Rsa pkcs#1 signature verification forgery is possible due to too-permissive signaturealgorithm parameter parsing.” (2014), [Online]. Available: https://bugzilla.mozilla.org/show_bug.cgi?id=1064636.

[18] Intel Security: Advanced Threat Research. “Berserk vulnerability – part 2: Certificate forgery in mozilla nss.” (2014), [Online]. Available: https://bugzilla.mozilla.org/attachment.cgi?id=8499825.

[19] S. Josefsson. “[gnutls-dev] original analysis of signature forgery problem.” (2006), [Online]. Available: https://lists.gnupg.org/pipermail/gnutls-dev/2006-September/ 001240.html.

[20] F. Valsorda. “Bleichenbacher’06 signature forgery in python-rsa.” (2016), [Online]. Available: https://blog.filippo.io/bleichenbacher-06-signature-forgeryin-python-rsa/.

[21] M. Bellare and P. Rogaway, “The exact security of digital signatures-how to sign with rsa and rabin,” in International conference on the theory and applications of cryptographic techniques, Springer, 1996, pp. 399–416.

[22] J. Jonsson, “Security proofs for the rsa-pss signature scheme and its variants,” Cryptology ePrint Archive, 2001.

[23] K. Moriarty, B. Kaliski, J. Jonsson, and A. Rusch, PKCS #1: RSA Cryptography Specifications Version 2.2, RFC 8017, Nov. 2016. doi: 10.17487/RFC8017. [Online]. Available: https://www.rfc-editor.org/info/rfc8017.

[24] J. Jonsson and B. Kaliski, Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1, RFC 3447, Feb. 2003. doi: 10.17487/RFC3447. [Online]. Available: https://www.rfc-editor.org/info/rfc3447.

[25] T. Jager, S. A. Kakvi, and A. May, “On the security of the pkcs# 1 v1. 5 signature scheme,” in Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 1195–1208.

[26] J.-S. Coron, “Security proof for partial-domain hash signature schemes,” in Advances in Cryptology—CRYPTO 2002: 22nd Annual International Cryptology Conference Santa Barbara, California, USA, August 18–22, 2002 Proceedings 22, Springer, 2002, pp. 613–626.