Information & Security: An International Journal

Baicheva & Dimitrov, vol.37, 2017, 29-34 https://doi.org/10.11610/isij.3702

A SOFTWARE TOOL 'ASR' FOR A POSTERIORI CRYPTANALYSIS ON PUBLIC KEYS GENERATED WITH 'RSA'

Tsonka BAICHEVA and Miroslav DIMITROV

Abstract: The asymmetric cryptosystem RSA is one of the first practical public-key cryptosystems, widely used for secure data transmission. The key generation of the RSA algorithm relies on specifically chosen numbers. Through the years, as the variety of attacks on RSA increased, plenty of recommendations and strategies for keys generation were published, for example NIST.FIPS.186-4.[1] Following the good practices, the regular users, companies or governments, can bootstrap their implementation of key generation software (a priori analysis). From the perspective of the other side of the communication channel, the key generation process and the keys themselves are a form of a "black box." Furthermore, most of the recommendations and good practices published online don't reference the reasons/attacks, which can exploit the specific wrongly chosen parameter. An automatic a posteriori cryptanalysis tool will link each recommendation with an existing attack, giving the other side of the communication channel a tool to detect unsafely (including weakened by purpose) generated keys.

Keywords: asymmetric cryptosystems, RSA, cryptanalysis, public keys, software tool

Introduction

In 1977, Ron Rivest, Adi Shamir and Leonard Adleman[2] created one of the first asymmetric cryptosystems "RSA" (abbreviation of their family names). It is currently one of the most widely used cryptosystems for providing security over insecure channels.

The increasing demands of modern digital society, as well as the rising number of computer devices, network devices, IoT devices, etc, are rapidly increasing the sources of communication over the Internet. Each communication flow, which holds sensitive information, should be independently protected by providing confidentiality and integrity.[6] Unfortunately, an error in the implementation of a cryptographic algorithm can lead to critical incidents or sensitive information leaks.[5,7] In cases when providing security over insecure channels is a must, such as the Internet, the symmetric cryptographic algorithm's secrets (configurations, keys, etc.) are heavily depended on the asymmetric algorithm.[8]

The tool for a posteriori automatic cryptanalysis "ASR" on public keys (and messages) generated with RSA will provide a compact library for further testing the built-in parameters, testing various techniques to extract the plaintext from an intercepted message encrypted with RSA and a rich field for testing new and combined attacks such as those discussed by Dan Boneh.[3]

The "ASR" tool can be logically divided into two layers – inner and outer. The diagram of the outer layer is displayed in "Figure 1", while the diagram of the inner layer is displayed in "Figure 2." The following sections will briefly describe the logic flow of the software tool.

ASR – Outer Layer

Figure 1 represents the other layer of the "ASR" tool. Its components and main functionalities are further discussed below.

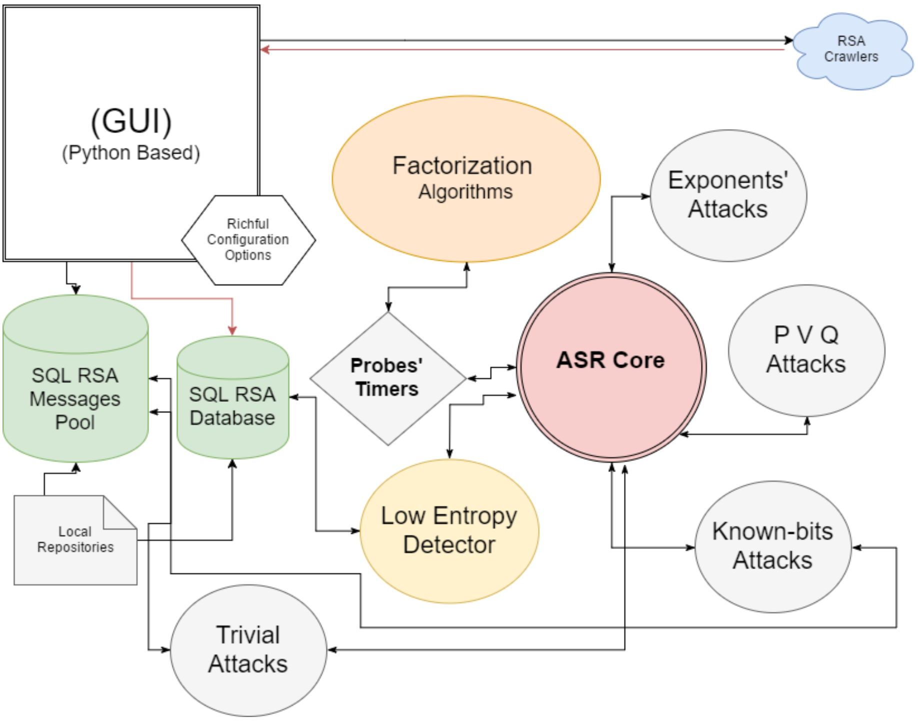

Figure 1: ASR Outer Layer.

The other layer components are, as follows:

- GUI: the graphical user-interface is going to be built by Python. It will be highly configurable, providing the user with many configuration options.
- RSA Crawlers: this set of modules is populating the SQL RSA Database. It will have an option to be constantly running in the background, collecting public keys from different sources - websites, digital signatures, network devices, IoT devices, etc. It will be further divided into sub-modules and parsers. Again, it will be highly configurable.
- SQL RSA Messages Pool: This database will be further logically divided by two different scenarios - set of different messages, encrypted with different public keys, or set of different public keys, which encrypted the same message.
- Probes' Timers: If the user launches some of the factorizations algorithms [4,11] on a specific modulus and the public key is generated following the recommendations, it is highly unlikely the algorithm to produce an answer (factors) within a reasonable timeline. This demands a module, which automatically bounds the running time of the specific factorization algorithm.
- ASR Core: this module encapsulates all the libraries and attacks implemented in the ASR tool. The literature provides examples of various attack scenarios.[9, 10, 12]
- Low Entropy Detector: Having a large populated SQL RSA Database, the module can initiate a common modulus attack. It will support an option to use simple GCD techniques, as well as an option of efficiently computing all-pairs GCD.[7]
- Local Repositories: The user can supply the encrypted messages manually.

ASR – Inner Layer

The variety of the most popular attacks on RSA, which the ASR tool will implement, depends on various constructions, abstract mathematical structures and complex algorithms. In “Figure 2,” the inner relationships of the ASR code are displayed, linking each attack with the desired features.

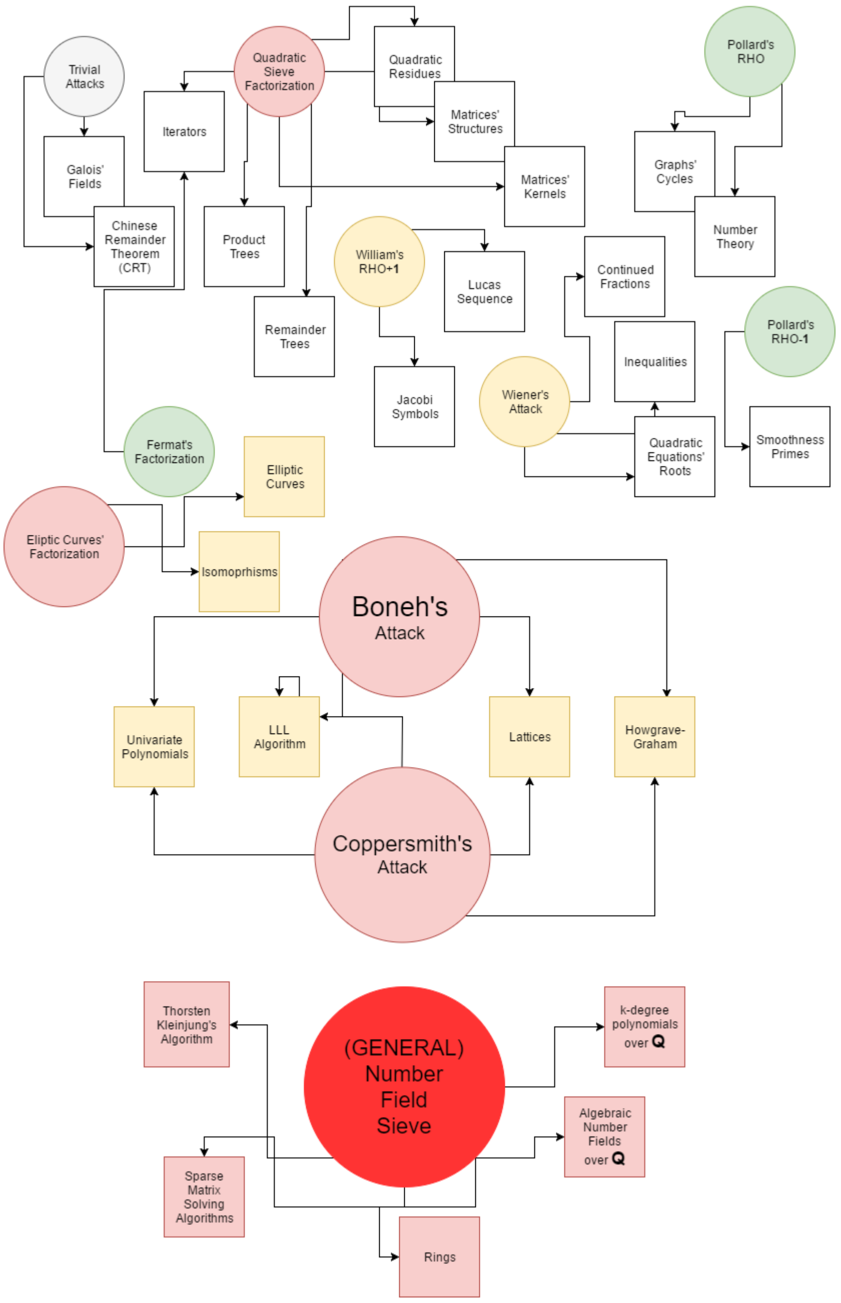

Figure 2: ASR Inner Layer.

Conclusion

The ASR software tool will be dynamically extended – new attack vectors can be easily attached to the project and the SQL RSA certificates' database will be automatically updated. The latter can be defined as a real-case snapshot of the current RSA certificates in use.

The documentation of the project will be exhaustive, having the ambitious goal to link every good practice or recommendation to an implemented attack within the ASR software tool.

It can be used by the developers of cryptographic applications from the private companies and the government sector as well as from any end-user willing to test to which extent its private data are adequately protected.

Endnotes

1 N.I.o.S.a.T. Information Technology Laboratory, "FIPS PUB 186-4 Digital Signature Standard (DSS)," Federal Information Processing Standards, 2013.

2 R.L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM* 21, no. 2 (Feb. 1978): 120–126.

3 Dan Boneh, "Twenty Years of Attacks on the RSA Cryptosystem," *Notices of the AMS* 46 (2002).

4 Divesh Aggarwal and Ueli Maurer, "Breaking RSA Generically Is Equivalent to Factoring," in: A. Joux A, ed., *Advances in Cryptology - EUROCRYPT 2009. Lecture Notes in Computer Science*, vol. 5479 (Springer, Berlin, Heidelberg, 2009), https://doi.org/10.1007/978-3-642-01001-9_2.

5 Daniel J. Bernstein, Yun-An Chang, Chen-Mou Cheng, Li-Ping Chou, Nadia Heninger, Tanja Lange, and Nicko van Someren, "Factoring RSA keys from certified smart cards: Coppersmith in the wild," *Cryptology ePrint Archive*, Paper 2013/599 (2013).

6 Jonathan Katz and Yehuda Lindell, *Introduction to Modern Cryptography* (CRC Press, 2007), 3-4.

7 Nadia Heninger, Zakir Durumeric, Eric Wustrow, and J. Alex Halderman, "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices," *21st USENIX Security Symposium, Bellevue, WA, August 8-10*, 2012.

8 N. S. Agency, "Commercial National Security Algorithm Suite," National Cryptographic Solutions Management Office, 19 August 2015.

9 Daniel J. Bernstein, "How to Find Smooth Parts of Integers," 2004.

10 Don Coppersmith, "Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities," *Journal of Cryptology* 10, (1997): 233–260, https://doi.org/10.1007/s001459900030.

11 J.M. Pollard, "A Monte Carlo method for factorization," *BIT Numerical Mathematics* 15, (1975): 331–334, https://doi.org/10.1007/BF01933667.

12 Abderrahmane Nitaj, "Another Generalization of Wiener's Attack on RSA," in S. Vaudenay, ed., *Progress in Cryptology – AFRICACRYPT 2008. AFRICACRYPT 2008. Lecture Notes in Computer Science*, vol. 5023 (Berlin, Heidelberg: Springer, 2008), 174-190, https://doi.org/10.1007/978-3-540-68164-9_12.

About the Authors

Prof. **Tsonka Baicheva** obtained her PhD in Informatics in 1998 and her PhD in Mathematics in 2015, both at the Institute of Mathematics and Informatics, Bulgarian Academy of Sciences. Since 2012, she is a professor at the same institute. Her research interests are in Coding Theory, Computer Communications, and Data Protection. She has published 73 papers in scientific journals and proceedings of conferences and has 222 citations of her publications.

Miroslav Dimitrov has just started his PhD at the Bulgarian Academy of Sciences under the supervision of Prof. Tsonka Baicheva. He has graduated in Informatics and Information Security (both at Sofia University). His main research is on cryptanalysis and applications of symmetric and asymmetric cryptosystems.