Permanent Reencryption: How to Survive Generations of Cryptanalysts to Come [⋆]

Marcus Völp, Francisco Rocha, Jeremie Decouchant, Jiangshan Yu, and Paulo Esteves-Verissimo

CritiX
SnT — Interdisciplinary Center for Security, Reliability and Trust
University of Luxembourg
firstname.lastname@uni.lu

Abstract. The protection of long-lived sensitive information puts enormous stress on traditional ciphers, to survive generations of cryptanalysts. In addition, there is a continued risk of adversaries penetrating and attacking the systems in which these ciphers are implemented. In this paper, we present our work-in-progress on an approach to survive both cryptanalysis and intrusion attacks for extended periods of time. A prime objective of any similar work is to prevent the leakage of plaintexts. However, given the long lifespan of sensitive information, during which cryptanalysts could focus on breaking the cipher, it is equally important to prevent leakage of unduly high amounts of ciphertext. Our approach consists in an enclave-based architectural set-up bringing in primary resilience against attacks, seconded by permanently reencrypting portions of the confidential or privacy-sensitive data with fresh keys and combining ciphers in a threshold-based encryption scheme.

1 Introduction

Many application areas today produce enormous amounts of privacy or otherwise sensitive data and metadata while often underestimating its longevity. Human genotypes, sequenced for research and medicine, are a prominent example of such data as it combines long-time valuable information with a general desire to keep this data online accessible for medical research and other applications [1]. Current next generation machines sequence genotypes at an enormous pace of around 10^{11} raw base pairs per day. As a result, the costs of sequencing a human's DNA dropped below 1000 USD, and more and more genotypes are

[⋆] This work is partially supported by the Fonds National de la Recherche Luxembourg (FNR) through PEARL grant FNR/P14/8149128.

[1] In fact we have selected genomic information for precisely this combination, but for the remainder of this paper and in hindsight of the workshop, please consider it as onyl one example of information with such properties and possibly not the best one. The interested reader is here directed to the transcript of the talk and the controversial discussion it triggered.

being sequenced, with protective measures barely keeping pace. Besides indicators allowing unique identification of individuals in a group, DNA may carry also stigmatizing information such as an individual's susceptibility to illnesses and its sensitivity to drugs.

DNA inheritance is a perfect example of an often overlooked aspect of sensitive data: its longevity. Such long lived data, originally collected in a legal way (e.g., to diagnose a patient), may be used for less ethical reasons in the future. For example, one could imagine the political impact in a presidential election if indicators for a serious disease in the genotype of a candidate's ancestor are made public. These threats are neither science fiction nor do they go unnoticed. For example, Melissa Gymrek and colleagues [1] were able to re-identify 13.1% of the donors contributing to the 1000-genome project, even though their data was anonymized, and Backes et al. [2] warned about the potential erosion of genomic privacy.

Naturally, a wealth of solutions exist at the protocol level, including synchronous [3]–[5] and asynchronous [6], [7] proactive secret sharing approaches [8] to limit the time of share exposure in (t, n)-threshold ciphers [9]. The (t, n)-threshold thereby ensures that attackers, who are able to compromise up to $t-1$ out of n locations while reencrypting shares every epoch of size T prevents adversaries from recombining secrets learned at different epochs. Similarly, an efficient proactive encryption scheme [10] is also introduced to periodically update the secret keys for decryption. In particular, there is a unique public key in the system for data encryption, and this public key and all ciphertexts in the system do not need to be updated. All secret keys are updated periodically, in the way that a sufficient number of the secret keys in any single epoch can decrypt the data encrypted using the non-changed public key. So, all users in the system only need to remember a single public key, and use this single public key for encryption in the life time of the system, even though the corresponding secret keys (possessed by the corresponding decryption servers) are updated. However, with data whose life-spans exceed 50 years, we must also take into account two further threats:

T1. long-term cryptanalysis using leaked ciphertexts and
T2. revealed vulnerabilities of ciphers, both at the protocol level and below.

The contributions we are working on are therefore:

1. a system architecture to safely consolidate shares in a single location while limiting the probability of revealing plaintext but, equally important, of the rate at which ciphertext can be revealed; and
2. a heterogeneous (multi-cipher) proactive sharing scheme to mask vulnerabilities in individual ciphers, in their implementation or in the execution environment they execute in.

It is paramount to consider system-level attacks while constructing novel ciphers because the protection of many systems is flimsy at best. For an account of the protection state of sensitive systems, consider the recent testimony given in front of the Committee on Oversight and Government Reform [11] where US Department of Homeland Security revealed that it still relies on known to be insecure Windows 2003 server installations with a transition planned only for the fiscal year 2018. Combined with the increasing availability and sophistication of tools to penetrate and attack systems, this positions data loss as one of the most paramount risks long-lived information is facing, even if data is revealed partially or fully encrypted.

2 Threat Model

In this work, we consider threats mounted by external attackers on computer systems used to process data whose value remains high for several generations. Attackers may eventually escalate privileges and become "insiders", but must attain their goals through remote exfiltration of information. Thus, we exclude attackers with local and/or physical access, or any other kind of attacks whose attainable ciphertext extraction bandwidth exceeds the practical bounds of any reencryption system our system may use. Indeed, such insiders might simply steal or copy the disks. In particular, we therefore assume appropriate access restrictions and supervision of maintenance works. We assume a locally synchronous setting with reliable timing information available to all components and shielded from attacks. Network-only access and hypervisor-only adjustable timers justify this assumption.

We generally assume advanced and persistent threats and highly skilled adversaries capable of penetrating any exposed system. More precisely,

1. adversaries penetrate systems from the outside through any possible connection and proceed from there to penetrate further components;
2. adversaries can only exfiltrate information by remote means (including covert channels), with a bounded and known extraction bandwidth;
3. adversaries have unlimited external storage for storing plain- and ciphertexts; and
4. they will eventually learn about prospective vulnerabilities in the ciphers or their implementation.

However, we assume the impenetrability of trusted-trustworthy components and a sufficiently strong isolation of enclaves to limit data exchange between compromised entities to a bandwidth bounded by b_{max}.

Our working hypothesis, which remains to be validated as part of our future work, is that given an assumption about the minimal amount C of ciphertext needed for a successful cryptanalysis attack — even taking into consideration vulnerabilities, which may shrink the search space and diminish that amount — by restricting the amount of ciphertext material an adversary may extract to be less than C, we maintain confidentiality.

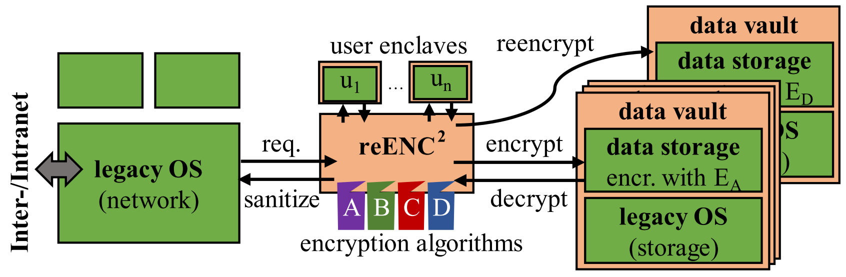

Fig. 1. Permanent reencryption architecture

3 Permanently Reencrypting Enclaves

To address on the one side potential future compromises of ciphers (both at the protocol and implementation levels) and, on the other side, breakthroughs in cryptanalysis and improvements of adversarial computing power, we take a more-shots-on-goal approach and use diversity, through a collection of replaceable encryption algorithms to protect the confidentiality (and likewise integrity and privacy) of stored data and metadata.

More precisely, on the architectural side, which we depict in Figure 1, we decouple the data vaults from the networked legacy operating system (OS) by channeling all accesses through our *reENC*^{2} component. Our goal is to limit the rate at which the ciphertexts of sensitive data are revealed to adversaries. We do not preclude covert channels and similar means to communicate between the data vaults and the networked legacy OS as long as the bandwidth between the two is lower than an upper bound b_{max}. In particular, we do not preclude the possibility of vulnerabilities in the legacy OSs (network and storage) and hence the possibility of compromise. *reENC*^{2} is the trusted-trustworthy permanent reencryption enclave in our hybrid system design [12], whose role is to mediate user access to the data vault for operations (ideally directly on the ciphertext [13], but if necessary also by decrypting the data, and by reencrypting it after these operations complete). The actual operations are thereby located in user-provided enclaves u_j, which receive the decrypted data after successfully passing an access control check. Moreover, u_j communication is sanitized by *reENC*^{2} to prevent unintended exposure of plaintext. We assume existence of sufficiently strong data sanitizers for all user operations and leave it as future work to construct such trusted-trustworthy sanitizers (possibly by requesting a proof of how u_j processes the data, which can then be matched against the data-release policy).

Ciphers plug into *reENC*^{2} and are provided the keys and data to process. The coupling has to be such that faults do not spread to other enclaves or to the legacy OS. Dedicated access to FPGA-based implementations or software implementations with dedicated message buffers are examples of such a decoupling.

Besides granting access and redirecting data into user enclaves, the second primary role of *reENC*^{2} is to permanently read data from the vault and reencrypt it with a fresh key and periodically also with a fresh cipher. To prevent a compromised data vault from keeping old data and, by this, from extending the time of ciphertext exposure, we redirect the output of reencryption to a fresh data vault and recycle the resources of old vaults once their data is transferred.

Let v be the number of data vaults. We anticipate vaults are executed in parallel. Hence, illegitimate channels between the vaults and the untrusted legacy OS have a maximal bandwidth of vb_{max}. Let c_i be the amount of ciphertext kept in vault i. Then adversaries are able to exfiltrate the tolerated fraction θ of c_i within

$$T_{leak} = \frac{\theta \ min_i(c_i)}{b_{max}}. \tag{1}$$

To prevent leakage of a large enough fraction of ciphertext, vaults have to be rejuvenated faster than T_{leak}. From this we can derive minimal bandwidth requirements between *reENC*^{2} and the old and new data vault of

$$b^{par}_{min} = \frac{2 \sum_i c_i}{T_{leak}} \tag{2}$$

in case of parallel re-encryption and of

$$b^{seq}_{min} = v \ b^{lpar}_{min} \tag{3}$$

for sequential re-encryption of one enclave at a time.

3.1 Long-Term Data Encryption

The remaining questions are now: "How can we encrypt data such that we drastically reduce the dependence on the correctness of a cipher?" and "How, by reencrypting the data, can we maintain this situation for the life-span of the data?". The properties which must be enforced are that:

1. all data that is encrypted with a given key needs to be reencrypted with a fresh key before the adversary gets hold of enough ciphertext material to succeed in the cryptanalysis of this data; and
2. all data that is encrypted with a given algorithm must be reencrypted with an alternative algorithm, before an adversary learns how to break the algorithm (e.g., once a vulnerability is discovered).

Of course, adversaries may learn about zero-day exploits before an algorithm is replaced. To keep our long-living data protected, we therefore propose a proactive two-stage encryption scheme, using techniques drawing from fault and intrusion tolerance, where we combine f_1-threshold crypto with f_2-cipher diversity, where f_1+1 and f_2+1, respectively the number of shares to be captured and the number of diverse ciphers to be broken, for a successful attack, define the parametric protection of the data. By putting the evaluation of these parameters in context with the above-mentioned C parameter, in a real system, we claim that we will be able to reduce the probability of a successful attack below a tolerable residual risk, over a long life span. We plan to evaluate our approach through a security analysis and field experiments.

References

[1] M. Gymrek, D. Golan, S. Rosset, and Y. Erlich, "Lobstr: A short tandem repeat profiler for personal genomes," *Genome Res.*, vol. 22, no. 6, pp. 1154–1162, 2012.

[2] M. Backes, P. Berrang, M. Humbert, X. Shen, and V. Wolf, "Simulating the large-scale erosion of genomic privacy over time," in *3rd International Workshop on Genome Privacy and Security (GenoPri)*, 2016.

[3] Y. Desmedt and S. Jajodia, "Redistributing secret shares to new access structures and its applications," Department of Information and Software Engineering, George Mason University, Tech. Rep. ISSE-TR-97-01, 1997.

[4] V. H. Gupta and K. Gopinath, "g^2_{its} vsr: An information theoretical secure verifiable secret redistribution protocol for long-term archival storage," in *International IEEE Security in Storage Workshop*, 2007, pp. 22–33.

[5] T. Wong, C. Wang, and J. Wing, "Verifiable secret redistribution for archive systems," in *International IEEE Security in Storage Workshop*, 2002.

[6] C. Cachin, K. Kursawe, A. Lysyanskaya, and R. Strobl, "Asynchronous verifiable secret sharing and proactive cryptosystems," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, ser. CCS '02, Washington, DC, USA: ACM, 2002, pp. 88–97.

[7] L. Zhou, F. B. Schneider, and R. Van Renesse, "Apss: Proactive secret sharing in asynchronous systems," *ACM Trans. Inf. Syst. Secur.*, vol. 8, no. 3, pp. 259–286, Aug. 2005.

[8] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," *CRYPTO - Lecture Notes in Computer Science*, vol. 963, pp. 339–352, 1995.

[9] A. Shamir, "How to share a secret," *Communication of the ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[10] J. Yu, M. Ryan, and L. Chen, "Authenticating compromisable storage systems," in *IEEE TrustCom'17*, 2017.

[11] D. A. Power, *Federal agencies need to address aging legacy systems*, Testimony before the Committee on Oversight and Government Reform, House of Representatives, May 2016.

[12] P. Sousa, N. F. Neves, and P. Verissimo, "Proactive resilience through architectural hybridization," in *ACM Symposium on Applied Computing (SAC'06)*, Dijon, France, Apr. 2006.

[13] C. Gentry, "A fully homomorphic encryption scheme," PhD thesis, Stanford, Sep. 2009.