Let Live and Let Die — Handling the State of Hash-based Signatures

Stefan-Lukas Gazdag[1], Denis Butin[2], and Johannes Buchmann[2]

[1] genua mbh, Germany
stefan-lukas_gazdag@genua.eu
[2] TU Darmstadt, Germany
{dbutin,buchmann}@cdc.informatik.tu-darmstadt.de

Abstract. Real-world use of digital signatures currently relies on algorithms that will be broken once quantum computers become available. Quantum-safe alternatives exist; in particular, hash-based schemes offer adequate performance and security and are seen as a fitting solution for post-quantum signatures. Unfortunately, they are not used at large because practical hurdles have not yet been overcome. In particular, their reliance on one-time signing keys makes it necessary to carefully keep track of a key index. We present strategies for handling the state of hash-based signatures for different use cases, ranging from infrequent software update authentication to high-frequency TLS connection initialization.

Keywords: Digital signatures, Hash-based, Integration

1 Motivation

Digital signatures are massively used online, notably for authentication, integrity checking and non-repudiation. The digital signature algorithms most commonly used in practice — RSA, DSA and ECDSA — rely on hardness assumptions about number theoretic problems, namely composite integer factorisation and the computation of discrete logarithms. In light of Shor's [19] algorithm, these arithmetical problems would be broken in the presence of quantum computing. While quantum computers are not yet available, their development is occurring at a swift pace [18]. However, post-quantum cryptography [3] provides a variety of quantum-resistant alternatives to classical digital signature schemes. Hash-based (or Merkle) signatures are one of the most promising of these alternatives and have received a lot of attention lately. Their construction combines the use of one-time signature schemes like Winternitz [13] with binary hash tree data structures. Hash trees relate a vast number of one-time signature to a single public key. Hash-based signatures offer an adequate trade-off between performance and security and have been the object of numerous successive improvements regarding speed and sizes as well as better security requirements (e.g. [9–11, 16]).

In this presentation We describe strategies to handle the state of hash-based signatures, including the access of private keys, error handling, integration into existing cryptographic software. Furthermore we show problems and possible solutions to issues like parameter selection, PKI and protocol integration.

2 Presentation Outline

Hash-based signature schemes are only recalled quickly. The concrete aspects of state handling that they require are the focus of the presentation. The following points will be discussed.

Statefulness of hash-based signatures When a hash-based signature scheme is used, private keys have to be updated during each signature generation. In particular, the index of the key pair for the next signing operation is set, the authentication path updated and — depending on the parameters and current index — nodes are stored. Thus, hash-based private keys have a state. This constraint creates a novel situation for key handling. Cryptographic software does not operate with changing keys usually. Normally, not only public keys are static; private keys stay unaltered as well. Dynamic private keys therefore contravene current usage. We investigate the practical issues originating from statefulness.

Coding and key handling Though many open-source cryptographic libraries offer generic method handling, interfaces do not always fulfil the requirements of hash-based signatures. Contemporary software usually assumes keys to be static. Therefore key sizes are sometimes also managed statically, causing problems when working with an updated key. As hash-based signature schemes can only yield a limited number of signature keys, users ought to be forewarned about the foreseeable end of usable signature keys. A fast key replacement method in case of Merkle tree depletion or key compromise is needed. Furthermore, software or hardware interrupts may occur at any time during key processing and any kind of damage ought to be prevented. We determine the stumbling blocks and discuss solutions.

Reducing the performance impact of key access Private key access frequency severely impacts runtime security and performance. Signature generation speed is crucial. However, since copies of the private key offer an attack vector to reveal old signature keys, any kind of backup must be prevented. If storage occurs after each signature, the old key has to be removed after reading and the updated one ought to be stored back to disk. As a safety precaution, the new key state is to be written to disk before returning the signature. Now, consider the case of a signing tool which is used multiple times and stores keys in virtual memory without writing private keys after every signature. Since the signing key changes after each signature, the updated key should be written to disk. Numerous read/write operations yield high latency, potentially unacceptable if signing is frequent. As mentioned above interrupts can always occur. We present different approaches

and concepts for securely working with keys while giving respect to the needs for different use cases. In addition, simultaneous private key access by different applications or instances must be avoided since multiple one-time signature key use is insecure. With that goal, we describe solutions that are included in our key access policies.

Adapting error detection and handling for statefulness Statefulness greatly increases the importance of error handling, even if the new key is stored after every change. Protection from unwanted key data changes is hard. Such changes can occur through malicious attacks or from bugs. Therefore, to protect keys, active checks are useful. We talk about the advantages and drawbacks of feasible error detection.

Stateless hash-based signatures A completely distinct approach is to modify the signature scheme to avoid statefulness altogether like proposed in the recently published work by Bernstein et al. [2]. We show why performance, security and new features legitimate the use of stateful schemes in a number of use cases.

Adapting Public Key Infrastructures A major use case for digital signatures is their application in PKI, which may be used for secure communication via data connection encryption or e-mail signing. The most commonly used PKI type is certificate-based authentication using X.509 [12]. Another possibility is the use of a *web of trust*. Here, a given public key must be trusted without any authority asserting the owner's identity. In this setting, public keys are often published on web homepages or via phonebook-like services, e.g. public key servers. Several studies of current PKI usage and future developments have been undertaken recently [1, 5–8, 15], also considering the needs for key evolving schemes like most hash-based signatures. An example is the question of how to keep third parties, such as certificate authorities, up-to-date about the dynamic key. Since no hash-based signature scheme has experienced widespread practical so far, the required implementations and improvements have not been performed yet. As these measures constitute significant modifications to existing PKI, gaining acceptance in this regard is crucial. The next major step is to standardise those protocols and add extensions to current standards. These developments are paramount for widespread diffusion. We present those ideas and show why and how they should be implemented.

Protocol integration Digital signatures find applications in many settings, including data traffic security, e-mail or software update authentication. To be used in practice, the new schemes must be integrated into familiar protocols. Popular protocols for securing data traffic are TLS and SSH. These depend on secure key exchange methods. A widely used key exchange scheme is the Diffie-Hellman protocol, but unfortunately it is not quantum-resistant. A post-quantum key-exchange alternative is therefore needed. Lately, several lattice-based post-quantum key exchange have been introduced (e.g. [4]). For post-quantum security, a quantum-safe signature scheme is often needed as well. We sketch strategies for integrating hash-based signatures into common security protocols.

Standardisation Another obstacle to the widespread use of hash-based signatures in practice is the lack of standardisation. So far, no official specification documents or format definitions exist for advanced variants. Recent contributions consist in McGrew and Curcio's IETF Internet-Draft for the basic Merkle scheme [17] and Housley's integration of the Merkle scheme in CMS [14]. For new implementations to cryptographic libraries and software, standardisation and alignment to protocol specifications is essential. If hash-based signatures are to find practical relevance, the main improvements of advanced schemes like XMSS^{MT} must be standardised.

Conclusion The presentation concludes with a synthesis of the key aspects of state handling discussed so far and an overview of our next steps to foster broader adoption of hash-based signatures.

References

1. Baier, H., Karatsiolis, V.: Validity Models of Electronic Signatures and Their Enforcement in Practice. In: Martinelli, F., Preneel, B. (eds.) Public Key Infrastructures, Services and Applications. Lecture Notes in Computer Science, vol. 6391, pp. 255–270. Springer (2010)
2. Bernstein, D.J., Hopwood, D., Hülsing, A., Lange, T., Niederhagen, R., Papachristodoulou, L., Schwabe, P., Wilcox O'Hearn, Z.: SPHINCS: practical stateless hash-based signatures (2014), `http://sphincs.cr.yp.to/sphincs-20141001.pdf`
3. Bernstein, D.J., Buchmann, J., Dahmen, E.: Post Quantum Cryptography. Springer (2008)
4. Bos, J.W., Costello, C., Naehrig, M., Stebila, D.: Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. Cryptology ePrint Archive, Report 2014/599 (2014)
5. Braun, J., Horsch, M., Hülsing, A.: Effiziente Umsetzung des Kettenmodells unter Verwendung vorwärtssicherer Signaturverfahren. In: Bundesamt für Sicherheit in der Informationstechnik (BSI) (ed.) Tagungsband zum 13. Deutschen IT-Sicherheitskongress. pp. 347–359. SecuMedia Verlag (2013)
6. Braun, J., Hülsing, A., Wiesmaier, A.: Schlanke Infrastrukturen für den digitalen Rechtsverkehr. Tech. rep., Technische Universität Darmstadt (2012)
7. Braun, J., Hülsing, A., Wiesmaier, A., Vigil, M.A.G., Buchmann, J.: How to avoid the Breakdown of Public Key Infrastructures — Forward Secure Signatures for Certificate Authorities. In: Public Key Infrastructures, Services and Applications — EuroPKI 2012. Lecture Notes in Computer Science, vol. 7868, pp. 53–68 (2012)

8. Braun, J., Kiefer, F., Hülsing, A.: Revocation and Non-repudiation: When the First Destroys the Latter. In: Katsikas, S.K., Agudo, I. (eds.) Public Key Infrastructures, Services and Applications — 10th European Workshop, EuroPKI 2013, Revised Selected Papers. Lecture Notes in Computer Science, vol. 8341, pp. 31–46. Springer (2013)
9. Buchmann, J., Dahmen, E., Hülsing, A.: XMSS — A Practical Forward Secure Signature Scheme Based on Minimal Security Assumptions. In: Yang, B.Y. (ed.) PQCrypto. Lecture Notes in Computer Science, vol. 7071, pp. 117–129. Springer (2011)
10. Buchmann, J., Dahmen, E., Klintsevich, E., Okeya, K., Vuillaume, C.: Merkle signatures with virtually unlimited signature capacity. In: Katz, J., Yung, M. (eds.) ACNS. Lecture Notes in Computer Science, vol. 4521, pp. 31–45. Springer (2007)
11. Buchmann, J., García, L.C.C., Dahmen, E., Döring, M., Klintsevich, E.: CMSS — An Improved Merkle Signature Scheme. In: Barua, R., Lange, T. (eds.) INDOCRYPT 2006. Lecture Notes in Computer Science, vol. 4329, pp. 349–363. Springer (2006)
12. Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., Polk, T.: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. Request for Comments, Internet Engineering Task Force (2008), RFC 5280
13. Dods, C., Smart, N., Stam, M.: Hash Based Digital Signature Schemes. In: Smart, N.P. (ed.) Cryptography and Coding, Lecture Notes in Computer Science, vol. 3796, pp. 96–115. Springer (2005)
14. Housley, R.: Use of the Hash-based Merkle Tree Signature (MTS) Algorithm in the Cryptographic Message Syntax (CMS). Request for Comments, Internet Engineering Task Force (2014), Internet-Draft: `https://datatracker.ietf.org/doc/draft-housley-cms-mts-hash-sig/`
15. Hülsing, A., Braun, J.: Langzeitsichere Signaturen durch den Einsatz hashbasierter Signaturverfahren. In: Informationssicherheit stärken — Vertrauen in die Zukunft schaffen. Tagungsband zum 13. Deutschen IT-Sicherheitskongress. pp. 565–576. Bundesamt für Sicherheit in der Informationstechnik (BSI) (2013)
16. Hülsing, A., Rausch, L., Buchmann, J.: Optimal Parameters for XMSS^{MT}. In: Cuzzocrea, A., Kittl, C., Simos, D.E., Weippl, E., Xu, L. (eds.) CD-ARES Workshops. Lecture Notes in Computer Science, vol. 8128, pp. 194–208. Springer (2013)
17. McGrew, D., Curcio, M.: Hash-Based Signatures. Request for Comments, Internet Engineering Task Force (2014), Internet-Draft: `https://datatracker.ietf.org/doc/draft-mcgrew-hash-sigs/`
18. Saeedi, K., Simmons, S., Salvail, J.Z., Dluhy, P., Riemann, H., Abrosimov, N.V., Becker, P., Pohl, H.J., Morton, J.J.L., Thewalt, M.L.W.: Room-Temperature Quantum Bit Storage Exceeding 39 Minutes Using Ionized Donors in Silicon-28. Science 342(6160), 830–833 (2013)
19. Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput. 26(5), 1484–1509 (1997)