Covert Channels in One-Time Passwords Based on Hash Chains

Jörg Keller*

FernUniversität in Hagen, Hagen, Germany
joerg.keller@fernuni-hagen.de

Steffen Wendzel

Worms University of Applied Sciences, Worms, Germany
wendzel@hs-worms.de

ABSTRACT

We present a covert channel between two network devices where one authenticates itself with Lamport's one-time passwords based on a cryptographic hash function. Our channel enables plausible deniability. We also present countermeasures to detect the presence of such a covert channel, which are non-trivial because hash values are randomly looking binary strings, so that deviations are not likely to be detected.

CCS CONCEPTS

• **Security and privacy** → **Hash functions and message authentication codes**; Mathematical foundations of cryptography; • **Theory of computation** → *Cryptographic primitives*.

KEYWORDS

Cryptographic Hash Function, Hash Chain, Plausible Deniability, Steganography, Covert Channel

ACM Reference Format:

Jörg Keller and Steffen Wendzel. 2020. Covert Channels in One-Time Passwords Based on Hash Chains. In *European Interdisciplinary Cybersecurity Conference (EICC 2020)*, November 18, 2020, Rennes, France. ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3424954.3424966>

1 INTRODUCTION

Covert channels (CCs) are unforeseen communication channels in a system design. While first CCs for local computers were described in the 1970's (cf. [7]), research of recent decades discovered a plethora of new and sophisticated CCs that aid the secret exchange of information between hosts, databases, network hosts, and IoT devices. Due to their stealthy and policy-breaking nature, CCs enable several actions related to cybercrime, such as the secret extraction of confidential information, barely detectable botnet command & control channels, and unobservable communication for cybercriminals. While legitimate use cases are imaginable as well, e.g. journalists using CCs for secure exchange of dissident-related information, criminal use seems foremost, so that presentation of new CCs also serves presentation of countermeasures.

We present the first CC that exploits cryptographic hash chains, which have become popular because some form (block chain) is

used in crypto currencies. Such CCs can be considered an example of plausible deniability: Alice communicates with Bob over the CC. Both can state that every possible hash value is equally likely to occur. By modifying (alternating) bits of hash values, Alice and Bob can thus plausibly deny the existence of the CC. We also sketch possible countermeasures, which are non-trivial because hash values are randomly looking bit strings.

2 BACKGROUND

A hash function is a function $h : U \rightarrow R$ that maps a possibly infinite universe, e.g. $U = \{0, 1\}^*$, onto a finite set of hash values, e.g. $R = \{0, 1\}^m$ [8]. A cryptographic hash function should have pre-image resistance, i.e. from a hash value $y \in R$ it is not possible (with reasonable effort) to compute a string $x \in U$ such that $h(x) = y$, 2nd pre-image resistance, i.e. from a given x with $h(x) = y$ it is not possible to compute $x' \in U, x' \neq x$ such that $h(x') = y$, and collision resistance, i.e. it is not possible to compute two strings $x, x' \in U, x' \neq x$ such that $h(x) = h(x')$. As $R \subset U$, a hash function can be applied repeatedly, i.e. for a seed $x_0 = s$, we can compute $x_{i+1} = h(x_i)$ for $i = 0, 1, 2, \dots, n-1$. This sequence is called a hash chain. While it is easy to compute a hash chain in forward direction, it cannot be computed in backward direction, starting from x_n , because of the pre-image resistance property.

A one-time password (OTP) is a password only used once to authenticate an entity. As such, it is secure as long as it is secret and cannot be guessed prior to its use. Lamport [6] presented a scheme to generate OTPs from a hash function with pre-image resistance. Entity A uses a random and secret seed s to compute a hash chain $x_0, \dots, x_n$, and transfers x_n to entity B via a secure channel. To authenticate, A sends $z = x_{n-1}$ over an insecure channel to B . B checks if $h(z) = x_n$. If yes, the authentication is approved, and B replaces x_n by $z = x_{n-1}$. If no, authentication has failed. For the next authentication, A sends $z = x_{n-2}$, and so on. Each password is only used once, and by the pre-image resistance, the next password cannot be guessed from the previous password. Also B need not keep x_n secret, as nothing can be inferred from this value. Haller's S/Key [5] and the TESLA protocol [9] use similar ideas.

A CC defines a parasitic communication channel nested inside a computing environment that allows at least two different legitimate processes (or nodes) to access a shared resource. The CC exploits the legitimate processes in a way that it allows the signaling of hidden information via the shared resource. The sender and receiver of a CC are called covert sender (CS) and covert receiver (CR), respectively. CS and CR are not necessarily identical with the overt sender and overt receiver (OS and OR) as CS and CR might act in an indirect manner or as man-in-the-middle. CCs inside computer networks follow a set of known hiding patterns [10] to transfer information, e.g. by placing secret data inside unused or reserved header bits of network packets, by modifying header fields with random values, or by modulating the timings between succeeding network packets.

*This work is supported by project SIMARGL (Secure intelligent methods for advanced recognition of malware, stegomalware & information hiding methods, <https://simargl.eu>), which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 833042.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

EICC 2020, November 18, 2020, Rennes, France
© 2020 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-7599-3/20/11...\$15.00
<https://doi.org/10.1145/3424954.3424966>

Related Work: Plausible deniability for steganography was proposed by several authors, especially for filesystems (e.g. [2]). PRNG-output was also applied for this purpose, e.g. in the context of video streams [4]. Abad [1] presents a CC in the check sum of IP packets, which however does not use a cryptographic hash function. Calhoun et al. [3] describe a CC in which OTPs are used as an element for generating random-data for a CC that exploits rate switching in 802.11b WiFi networks. In comparison, our approach allows an Internet-wide application that is not limited to wireless environments. Moreover does our approach represent an indirect CC based on hash chain exploitation. To the best of our knowledge, no other CC was published that exploits cryptographic hash chains.

3 COVERT CHANNELS IN HASH CHAINS

We present two variants of a CC that can be used in OTPs based on hash chains. We assume that each hash value is transmitted as part of network packets between A and B , so that a modification of the hash value can be hidden by re-computing the packet's check sum. In our scenario, CS is located close to A (or on OS) and CR close to B , i.e. both have at least indirect access to the communication between A and B .

Variant 1: The message that CS wants to send is broken into pieces of $\log_2 m$ bits each (we assume m to be a power of 2), i.e. represented as symbols over alphabet $\{0, 1, \dots, m-1\}$. To send a symbol j , CS flips bit j of the password x_i that is going to be transmitted. CR, who knows the previous password x_{i+1} , intercepts the modified password x'_i upon arrival and tests x'_i with bit k flipped, for $k = 0, 1, \dots$, until a hash results in x_{i+1} . Then CR stores symbol k and forwards the corrected password to B . This is repeated until the complete message is transmitted (assuming that the number of pieces is less than n , the length of the chain).

Variant 2: Here, CS only sends one bit with each transmitted password x_i . To send a 1, bit 0 of the password is flipped. CR intercepts the possibly modified password x'_i and checks if $h(x'_i) = x_{i+1}$. If yes, then CR stores a 0, and forwards the password to B . If no, then CR stores a 1, corrects the password by flipping bit 0, and forwards the corrected password to B . This is repeated until the complete message is transmitted. Obviously, one is not restricted to always use bit 0. The bit that is possibly flipped can be decided depending on i , synchronized timing events or other methods.

Both variants perform a manipulation of (pseudo-)random packet fields as described by the *random value* pattern [10]. Random value techniques have been implemented by several authors, rendering our proposed CC variants applicable in realistic environments.

Note that in both variants, CR needs no knowledge of x_n stored by B . In the first transmitted password x_{n-1} , no symbol is transmitted. Then CR knows x_{n-1} when intercepting and forwarding that password, and can refer to it later on, duplicating B 's work.

Variant 1 can be detected more easily than variant 2 on the receiver side, as variant 1 on average encompasses $m/2$ evaluations of h , which is a notable amount of computation. This gives rise to intermediate forms where only $m' < m$ bits can be used, i.e. the message is represented in symbols over alphabet $\{0, \dots, m'-1\}$. Parameter m' will be chosen as large as possible, and as small as necessary to avoid detectability. In contrast, variant 1 provides higher steganographic bandwidth than variant 2.

On the sender side, both variants have very low effort and are thus hardly detectable. During transmission, an isolated packet will look innocent, as the one-time passwords look like random bitstrings, so that a possible bit flip is not likely to leave any trace or signature. Hence, detection during transmission seems non-trivial.

4 COUNTERMEASURES

Possible countermeasures against variant 1 on the receiver side have been described in the previous section. Countermeasures during transmission require that a warden has knowledge of two successive packets with passwords x'_i and x'_{i-1} . In this case, the warden can check if $h(x'_{i-1}) = x'_i$. If either password has been modified by CS, this equality will not hold because of the hash function's properties.

Another possible countermeasure would be that A and B modify the passwords themselves, in a way that is not foreseeable by CS or CR. When A is about to send x_i , it already has knowledge about the following password x_{i-1} . Let us assume that both A and B have knowledge of a hash function $g : \{0, 1\}^m \rightarrow \{0, 1\}$. If $g(x_{i-1}) = 0$, then A sends x_i . If $g(x_{i-1}) = 1$, then A sends $\bar{x}_i$, i.e. x_i with all bits inverted. If g is a good hash function, then the second possibility will occur half of the time. If B receives a password x'_i , then it checks if either $h(x'_i)$ or $h(\bar{x}'_i)$ equals x_{i+1} . In the latter case, B stores $\bar{x}'_i$.

As only two instead of one password value will be accepted, this scheme is still secure. But life is now more complicated for CR. In variant 2, it cannot simply compare hash values, but must use all possible combinations of inverted and non-inverted passwords. More complicated schemes are possible. Even if they are not preventing this CC, they can restrict bandwidth and/or can increase detectability by forcing CR to perform more computation.

5 CONCLUSIONS

We have presented the first CC in hash chains, detailed different variants, and sketched non-trivial countermeasures against them. Instead of using network packets, the CCs could also use a local system to break a security policy if A authenticates against B over a shared medium that both, CS and CR, have access to under some timing constraints. Future work will comprise implementation and validation of the CC and the countermeasures.

REFERENCES

- [1] C. Abad. 2001. *IP Checksum Covert Channels and Selected Hash Collision*. Technical Report. Univ. of California, Los Angeles.
- [2] R. Anderson, R. Needham, and A. Shamir. 1998. The steganographic file system. In *International Workshop on Information Hiding*. Springer, 73–82.
- [3] T. E. Calhoun Jr, X. Cao, et al. 2012. An 802.11 MAC layer covert channel. *Wireless Communications and Mobile Computing* 12, 5 (2012), 393–405.
- [4] S. Craver, E. Li, and J. Yu. 2009. Protocols for data hiding in pseudo-random state. In *Media Forensics and Security*, Vol. 7254.
- [5] N. Haller. 1995. *The S/KEY One-Time Password System*. RFC 1760. RFC Editor. <https://www.rfc-editor.org/rfc/rfc1760.txt>
- [6] L. Lamport. 1981. Password authentication with insecure communication. *Commun. ACM* 24, 11 (1981), 770–772. <https://doi.org/10.1145/358790.358797>
- [7] B. W. Lampson. 1973. A Note on the Confinement Problem. *Commun. ACM* 16, 10 (Oct. 1973), 613–615. <https://doi.org/10.1145/362375.362389>
- [8] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. 1996. *Handbook of Applied Cryptography*. CRC Press.
- [9] A. Perrig, R. Canetti, et al. 2002. The TESLA Broadcast Authentication Protocol. *CryptoBytes* 5, 2 (2002), 2–13.
- [10] S. Wendzel, S. Zander, et al. 2015. Pattern-Based Survey and Categorization of Network Covert Channel Techniques. *Computing Surveys* 47, 3 (2015).