Subliminal Channels in High-Speed Signatures

Alexander Hartl, Robert Annessi, and Tanja Zseby*
TU Wien, Vienna, Austria
alexander.hartl@student.tuwien.ac.at
robert.annessi@nt.tuwien.ac.at
tanja.zseby@tuwien.ac.at

Abstract

Subliminal channels in digital signatures can be used to secretly transmit information between two or more communication partners. If subliminal messages are embedded in standard signatures in network protocols, neither network operators nor legitimate receivers notice any suspicious activity. Subliminal channels already exist in older signatures, such as ElGamal and ECDSA. Nevertheless, in classical network protocols such signatures are used only sparsely, e.g., during authentication in the protocol setup. Therefore, the overall potential subliminal bandwidth and their usability as carrier for hidden messages or information leakage is limited. However, with the advent of high-speed signatures such as EdDSA and MQ-based signatures such as PFlash or MQQ-SIG, scenarios such as signed broadcast clock synchronization or signed sensor data export become feasible. In those scenarios large sequences of packets are each individually signed and then transferred over the network. This increases the available bandwidth for transmitting subliminal information significantly and makes subliminal channels usable for large scale data exfiltration or even the operation of command and control structures. In this paper, we show the existence of subliminal channels in recent high-speed signatures and discuss the implications of the ability to hide information in a multitude of packets in different example scenarios: broadcast clock synchronization, signed sensor data export, and classical TLS. In a previous paper we already presented subliminal channels in the EdDSA signature scheme. We here extend this work by investigating subliminal channels in MQ signatures. We present specific results for existing MQ signatures but also show that whole classes of MQ-based methods for constructing signature schemes are prone to the existence of subliminal channels. We then discuss the applicability of different countermeasures against subliminal channels but conclude that none of the existing solutions can sufficiently protect against data exfiltration in network protocols secured by EdDSA or MQ signatures.

Keywords: Information leakage, Insider threats, Subliminal channel, EdDSA

1 Introduction

Subliminal channels are hidden channels that allow an adversary to unnoticeably transmit information by exploiting the mathematical structure of cryptographic schemes. In particular, signature schemes are an attractive candidate for exploitation as subliminal channel. Unlike information hiding techniques like steganography or obfuscation, the sender of the subliminal message has no need to modify the overt message content. He therefore is able to use the channel even if he has no influence on the transmitted message or if a modification would raise suspicion.

Subliminal channels can be classified according to their bandwidth. Broadband channels allow the use of almost all the signature's bits that are not needed for security against forgery. Narrowband channels, on the other hand, yield a significantly lower subliminal bandwidth, typically just few bits [1].

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), 9:1 (March 2018), pp. 30-53
*Corresponding author: Gusshausstraße 25 / E389, 1040 Wien, Austria, Tel: +43-(1)-58801-38910

While subliminal channels were first described by [2], they did not receive much attention recently and were tolerated or possibly even ignored by protocol designers. Meanwhile new constructions for signature schemes that have several desired properties, have attracted interest in the scientific community. The EdDSA [3] signature scheme combines high-security with high-performant signing and verification operations and is already widely deployed. Signatures based on multivariate quadratic polynomials in finite fields (MQ signatures) use mathematical constructions that are very different from traditional signatures, which allows them to survive progress in quantum computing and also spawned high-speed signature schemes like PFlash [4].

At the same time, new application areas have emerged that require signing of a large number of messages per time unit, and thus can easily utilize the new high-speed signatures. But when transmitting a large number of signed messages, the problems that result from subliminal channels multiply. It is thus required to analyze if the new high-speed signatures allow the establishment of subliminal channels and reevaluate the consequences for those new application scenarios.

Scenarios are particularly susceptible to the establishment of a subliminal channel if authenticity, integrity or non-repudiation has to be guaranteed, but confidentiality is not as important, so data is sent unencrypted. This applies, for example, to group communication scenarios like broadcast clock synchronization or sensor data collection. When the subliminal channel is exploited, the leakage of sensible information or the establishment of covert malware communication becomes possible. Depending on the application field this can harm individuals, companies or even whole nations.

In this paper, we show how subliminal channels can be established using EdDSA and MQ signatures. After describing the operation of EdDSA, we describe how subliminal channels can be established using EdDSA, how the channels can be used in different scenarios and what methods can be used to prevent the subliminal communication. We accompany our argumentation by experiments in order to show the exploitation of the subliminal channel for specific use cases. We calculate the available subliminal bandwidth and point out difficulties of using the channel in practice. We then outline methods for constructing MQ signature schemes and describe if and how subliminal channels can be established for these signatures. We first show general possibilities for hiding information in the mathematical constructs of MQ signatures and several of their modifications. Then we investigate the achievable subliminal bandwidth for several existing MQ signatures such as QUARTZ, Gui-127, SFlash, PFlash, MQQ-SIG and Rainbow.

This paper is an extended version of our paper [5] (presented at the International Workshop on Managing Insider Security Threats MIST 2017) in which we already published the idea and the subliminal channel scenarios for the EdDSA signature. The major extension in this paper is the investigation and evaluation of subliminal channels in MQ-based signatures.

2 Related Work

Subliminal channels first appeared in a paper[2]. The same author extended his considerations to more practical signature schemes [6] and showed that a significant part of a signature's bits can be used to leak information without giving a legitimate receiver any means to discover the subliminal information exchange. Later, subliminal channels were found in all important signature schemes like RSA [7, 8, 9], DSA [1, 10] or ECDSA [11, 12, 13]. Ensuring that a signer does not actively exploit the subliminal channel turned out to be a difficult task.

The differentiation between narrowband and broadband channels was already introduced. In order to use a broadband channel, the subliminal receiver in most cases has to know the signer's secret key or at least parts of it. Noteworthy in this context is the Newton channel [14] which was found for the ElGamal signature scheme [10] specifically. When using the Newton channel, the signer unveils as many bits of information about the secret key to the subliminal receiver as afterwards should be usable as subliminal

bandwidth.

The concept of subliminal channels is related to SETUP (Secretly Embedded Trapdoor with Universal Protection) attacks that were introduced [15]. When performing a SETUP attack, an adversary replaces a cryptographic algorithm on a victim's device by an altered algorithm aiming to break its security. In the context of digital signatures this means that the modified signing algorithm leaks the secret key to the adversary. A. Young also introduced the realm of Kleptography which is defined as the "study of stealing information securely and subliminally" [15]. Recently attacks based on modifying cryptographic algorithms attracted anew research interest and are now called Algorithm-Substitution Attacks (ASAs) [16, 17] and Subversion Attacks (SAs) [17]. Most noteworthy, the paper [17] provides a detailed discussion about the requirements for a signature scheme to be susceptible to SAs.

3 The EdDSA Signature Scheme

EdDSA [3] is based on a signature scheme that was first described by C. P. Schnorr [18]. RFC 8032 [19] standardizes two variants: Ed25519 operates on the twisted Edwards curve Curve25519 [20] and yields a security level of 128. Ed448 uses the Edwards curve Curve448 [21] and yields 224 of security.

In the following, b and c are integers with $b = 256$ and $c = 3$ in the case of Ed25519 and $b = 456$ and $c = 2$ for Ed448, respectively. B is a public point on the elliptic curve with order L and H, H_a and H_r are cryptographic hash functions that produce values in $\mathbb{Z}_L$, the set of nonnegative integers smaller L.

The secret key consists of a b-bit string k. For signing, knowledge of $a = H_a(k)$ is sufficient. In the following, we will therefore term a the signing key. The public key consists of the point on the elliptic curve $A = aB$. For signing a message M, the signer computes a nonce value $r = H_r(k,M)$, the curve point $R = rB$ and the value $S = r + H(R,A,M)a \bmod L$. The signature then consists of the values R,S. For verification the equation $2^c SB = 2^c R + 2^c H(R,A,M)A$ is verified to hold [3].

Different issues have to be considered for the nonce value r [3]. The nonce r must remain secret for all signatures as otherwise a could be computed as $a = (S-r)/H(R,A,M) \bmod L$. Furthermore, the same nonce must never occur again for different messages M_1 and M_2, as also in this case a could be computed as $a = (S_1 - S_2)/(H(R,A,M_1) - H(R,A,M_2)) \bmod L$. By computing the nonce as described above, both issues can be addressed.

4 Subliminal Channels in EdDSA

For the reasons pointed out in the previous section, it is reasonable to use a cryptographic hash function as described in [3] for deriving the nonce value. However, as the calculation involves the secret key, a verifier has no means to test if the method indeed has been used. Therefore, an arbitrary (random) value could be used instead for r. This allows a broadband as well as a narrowband subliminal channel to be established in EdDSA signatures.

4.1 The Broadband Channel

Knowing the signing key a, the nonce can be recovered from a signature as $r = S - H(R,A,M)a \bmod L$. Hence, if subliminal information is directly encoded into r, it can be recovered by anyone who holds the signing key. This constitutes a broadband subliminal channel with a bandwidth of $\log_2 L \approx 252$ bit per signature for Ed25519 and $\log_2 L \approx 447$ bit per signature for Ed448. Fig. 1 depicts the setup for the subliminal channel.

The requirement for the subliminal receiver to know the signing key a is a major restriction, as knowledge of a allows the subliminal receiver to forge arbitrary signatures, which, however, may be

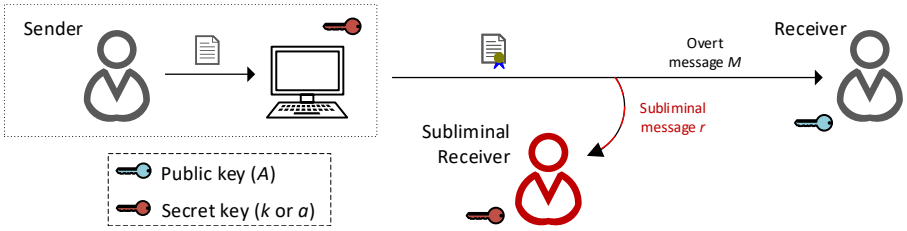

Figure 1: The broadband subliminal channel in EdDSA.

acceptable for a malicious signer who collaborates with an external receiver. But it requires upfront key sharing, which can take place offline or by using the narrowband subliminal channel described below.

A further requirement is that both the signature and the signed message are known to the subliminal receiver. Usability of the subliminal channel is thus ruled out if communication is encrypted after the signature has been attached and the decryption key is unknown to the subliminal receiver. Use of the encrypt-then-sign paradigm makes exploitation of the subliminal channel particularly easy. In this case the signed message is the ciphertext of the message and the signature itself is unencrypted. Hence, the nonce value can be recovered without the need to decrypt the ciphertext.

4.2 A Narrowband Channel

A very general approach for establishing a subliminal channel exploiting signature schemes that allow multiple valid signatures for a message, works as follows. The signer tries to make the encoded representation of the signature show a specific bit pattern that corresponds to the intended subliminal information. In our case, the signer could, for example, aim to make the last byte of the encoded representation of R equal to the subliminal information. As computing discrete logarithms is infeasible, he cannot directly find a nonce value, for which R has the desired properties. However, trying a large-enough number of different values for r, a suitable value can eventually be found by chance. On average the signer will have to try 2^{B_s} different values, where B_s denotes the subliminal bandwidth in bits. Due to the exponential growth of the number of required tries with the subliminal bandwidth B_s, only a small portion of the signature's bits can be used as subliminal channel, which makes this channel narrowband.

The benefits of this method as compared to the broadband channel are that the subliminal receiver does neither have to know the signing key a, nor the signed message. Note, furthermore, that this narrowband channel is exploitable even if communication is encrypted under the mild assumption that the subliminal receiver can locate the ciphertext of the signature in the encrypted data. The signer then tries to make the ciphertext of the encoded representation of R show the intended bit pattern. To reach this goal, he proceeds as described above.

5 Attack Scenarios

Digital signatures are used in a variety of different scenarios for securing communication. With the advent of high-speed signatures, applications become feasible that were not possible with signature schemes like RSA, DSA or ECDSA due to their high computational requirements for the signing and/or the signature verification process. We will now describe some scenarios for which a subliminal channel can pose a severe threat for information security.

For the scenarios we show the use of the EdDSA as signature scheme, because of its attractive properties giving rise to a widespread use for current and future deployments. But our conclusions can

[0]For projects that employ EdDSA see `https://ianix.com/pub/ed25519-deployment.html`.

be applied to other (high-speed) signatures as well if they yield a subliminal channel. For example, high-speed MQ signature schemes as described in Section 8 could provide authenticity for the same scenarios in the future if their security is approved. Therefore, the scenarios for EdDSA are in the same way applicable for these MQ signatures.

As described in Section 4, to be able to use the broadband channel of EdDSA, the subliminal receiver needs to know the signing key a and one of the following requirements have to be met: (1) the communication is unencrypted, (2) the communication is encrypted using the encrypt-then-sign paradigm or (3) the subliminal receiver also knows the decryption key. While (1) is a reasonable assumption for some application scenarios where confidentiality is of no importance, (3) can be achieved by leaking the decryption key together with the signing key upfront using, for example, the narrowband subliminal channel. In the following we will assume that the conditions are met.

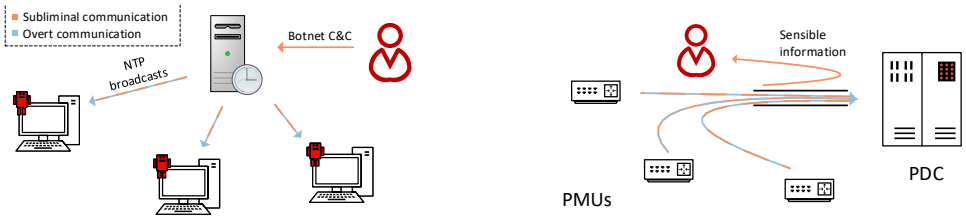

Figure 2: Subliminal channels in high-speed signatures: Command and Control communication using NTP broadcasts (left) and information leakage exploiting phasor measurement transmissions (right).

5.1 Clock Synchronization

Accurate time information becomes increasingly important for the proper functioning of critical infrastructure [22]. Consequently, protocols like the Network Time Protocol (NTP) or the Precision Time Protocol (PTP) are used for providing clock synchronization, which again needs to be secured. However, a method for authenticating NTP or PTP packets not only has to provide appropriate security, but also has to be fast to keep the influence on the error of the transmitted time information low. High-speed signature schemes have been proposed to be used for this purpose [22, 23]. Confidentiality is usually irrelevant for time information and, hence, encryption is likely to be omitted. The signatures used for securing NTP and PTP packets thus provide ideal conditions for exploitation as a subliminal channel.

For the impact of the subliminal channel it has to be differentiated if time synchronization happens in unicast or broadcast mode. In unicast mode a client exchanges few messages with the server. Signing each of these messages, the bandwidth of the resulting subliminal channel may suffice for leaking sensible information, which is particularly severe if synchronization is performed across the Internet, thus leaving the protected network of a company.

In broadcast mode, time is broadcast in regular intervals across a company's network. As these broadcasts occur in regular intervals, the amount of data that can be transferred using the subliminal channel, is large when observing a large-enough time span. Furthermore, for receiving the leaked data, an adversary does not have to take special measures for eavesdropping on the signed messages as any network access suffices.

Finally, also in scenarios where the adversary wants to reach a large number of network nodes the time broadcasts yield an attractive subliminal channel. As example, Fig. 2 depicts the operation of a botnet. If the adversary has managed to install malware on a large number of network nodes and also has infected the time server, he can use the subliminal channel for transmitting command and control messages to his bots. Approaches to discover the botnet by detecting the command and control communication then are foredoomed.

5.2 Smart Grid Communication

Smart grids enhance electrical grids by information technology to optimize grid operation. Power grids have to meet highest requirements in availability and quality of supply. As a consequence, for smart grids, high requirements for availability, integrity, authenticity and possibly also confidentiality, have to be considered.

Digital signatures are thus of fundamental importance for providing authentication. The use of high-speed signatures is favorable in many situations when signatures shall be processed on low-power hardware like sensor devices, if a large amount of data has to be signed or if low latency is important.

In such a setting there are several reasons for why a subliminal channel has a significant impact on information security. The communicating partners often store sensible data like maintenance schedules, configuration parameters or even key material. Among others, this data can be used for preparing an attack on critical infrastructure of the grid. Furthermore, real-time applications require data to be transmitted with a large frequency. Signing each of these packets individually, a vast subliminal bandwidth results for data exfiltration. Finally, due to the widespread deployment of sensor devices or other smart grid components, an adversary faces a widespread infrastructure for mounting attacks. The homogeneous hardware and configuration of many of these devices allows malware to spread more easily.

An example of smart grid applications where signatures can lead to a vast bandwidth for data exfiltration is the transmission of measurements by Phasor Measurement Units (PMUs). These devices measure the phasor of electrical current and voltage and transmit the measurements to Phasor Data Concentrators (PDCs), finally supporting control decisions for grid operation (see Fig. 2). The measurements have to be transmitted in real-time, where usually 60 to 120 measurements are performed each second. Authenticity and integrity is crucial as otherwise wrong control decisions might result. Confidentiality of phasor measurements is of little importance and, hence, encryption might be omitted. The use of high-speed signatures for signing transmissions with such a high rate of signed packets seems natural. Hence, also this scenario yields ideal conditions for the establishment of subliminal channels.

5.3 TLS Key Exchange

When using Diffie-Hellman in ephemeral mode during the key exchange phase of the handshake of the Transport Layer Security (TLS) protocol [24, 25], signatures are used for authenticating the server, ensuring integrity of the key exchange and, optionally, for authenticating the client. EdDSA can be used for this purpose for TLS in the current draft version TLS 1.3 [25]. It was also proposed for earlier versions [26]. It is hence possible to use the signature(s) as subliminal channel.

The signed data contains major parts of the handshake. As it also contains the random numbers that both client and server transmit to the respective other, identical signed data, which could lead to the detection of the subliminal channel as described in Section 7.4, will never occur.

It is noteworthy that the subliminal channel is more easy to exploit in TLS 1.2 and earlier versions than in TLS 1.3. This is because in TLS 1.3 the signatures and other important parts of the handshake are already encrypted, thus requiring the shared secret for decryption and use of the (broadband) subliminal channel.

In TLS and other security protocols signatures are only used during the initial key agreement phase. The subliminal bandwidth is thus significantly lower than in the two other scenarios described above. Also, the TLS protocol yields numerous further covert and subliminal channels [5, 27, 28, 29, 30, 31]. Also other signature schemes usable for this purpose yield subliminal channels [1, 11]. Nevertheless, we attribute the subliminal channel a great importance because of the tremendously wide deployment of TLS. Infecting a commonly used library with malware allows an adversary to clandestinely leak large amounts of sensible data.

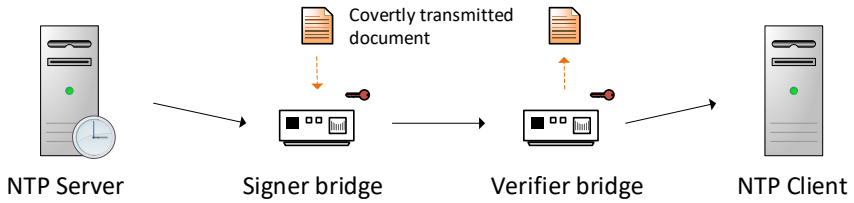

Figure 3: Setup for investigating a subliminal channel in signed NTP broadcasts .

5.4 Further Attack Scenarios

Due to the attractive properties of EdDSA, there are further scenarios, where a subliminal channel can have a significant impact on information security. We here give some examples of such scenarios, which we will not describe in more detail for reasons of space.

As highlighted in [11] a subliminal channel can be used for placing information in digital signatures of passports. The same applies to digital health insurance cards. The issuing instance can embed sensible information without the owner's knowledge. Furthermore, nested signatures are used for achieving path validation for the Border Gateway Protocol (BGP). In this case, a subliminal channel provides the possibility for clandestine information exchange between BGP routers. Signatures can also be used for providing security for the Domain Name System (DNS). In particular, the DNSSEC extension [32] and DNSCrypt[1] both support EdDSA. As DNS lookups occur frequently for some applications, a large subliminal bandwidth is possible. Finally, signatures are essential for cryptocurrencies to prove ownership of coins. If a subliminal channel exists, they can be exploited for unnoticeably transmitting and storing data on the blockchain.

6 Experimental Results

We performed experiments to investigate the broadband subliminal channel that EdDSA yields in practice. Our goals when performing the experiments were to prove the existence of the subliminal channel in practice, to get an impression of the difficulty of using the subliminal channel and to find the subliminal bandwidth with which data can be leaked in the above scenarios in practice.

6.1 Signed NTP Broadcasts

We used the experimental setup from [22] to investigate the subliminal channel that results from signing NTP broadcasts with EdDSA (see Fig. 3). Instead of modifying the source code of the NTP process and client, the tasks of signature generation and verification are performed by bridges located between NTP server and client. The subliminal information is thus embedded by the signer bridge. The subliminal receiver can be anywhere on the broadcast domain between signer and verifier bridge. In addition to signature verification we could thus recover the subliminal information on the verifier bridge. In this case, the secret key is required on the verifier bridge in contrast to just the public key in the usual situation.

Server, client and bridges were running Debian 'Jessie' as operating system and the insertion and removal of signatures was performed with iptables and nfqueue on the network bridges. We used the cryptographic routines from the NaCl[2] library. For the task of recovery of the subliminal information we had to enhance the library by a routine for performing field subtractions. Apart from this modification,

[1] `https://dnscrypt.org/`
[2] `https://nacl.cr.yp.to/`

substitution and recovery of the nonce value was straight-forward. For sake of simplicity, we transmitted 248 instead of 252 of subliminal information to avoid dealing with partial bytes.

With this setup the subliminal channel was proven functional with the expected subliminal bandwidth. Without modifying its source code, NTP allows broadcasting time information every 8 seconds. We thus observed a subliminal bandwidth of 3.9/.

Hence, also in a practical setting the subliminal channel is seen to be easily exploitable.

6.2 PMU Sensor Data Transmission

We used the signer and verifier bridges from the above setup to investigate signing sensor data transmissions by PMUs. Instead of an NTP server we thus used a 1133A Power Sentinel PMU by Arbiter Systems. When used with the manufacturer's proprietary PowerSentinelCSV protocol the device is able to send 10 UDP packets of measurement data per second. We were thus able to transmit 310/ of subliminal data, which constitutes a considerable bandwidth for data exfiltration.

Other protocols for the same task like IEEE C37.118 allow a comparable measurement frequency resulting in a similar subliminal bandwidth.

6.3 TLS Authentication

As described in Section 5.4 TLS uses signatures to ensure authenticity of the communicating partners and integrity of the key exchange. We performed an experiment to show the practical exploitability of these EdDSA signature(s) as subliminal channel. For this we used the nginx[3] webserver, a simple HTTPS client application and the BoringSSL[4] TLS library. We chose to use this library because of its support of both Ed25519 and the current TLS 1.3 draft. Also in this case we had to enhance the library by a function that performs field subtractions. In addition to this, just minor modifications were necessary to be able to exploit the subliminal channel for both TLS 1.2 and TLS 1.3.

Hence, also for the TLS handshake the subliminal channel was proven easily exploitable with a subliminal bandwidth of 31 per key exchange. If client authentication is used and thus signatures are exchanged in both directions, the subliminal channel can be exploited in both directions.

7 Preventing Subliminal Communication

There are some approaches that aim to prevent subliminal communication while retaining compatibility with the usual signature verification algorithm. These approaches use the warden scenario (Fig. 4) that was introduced by G. J. Simmons [2]. The sender may only communicate with a warden to transmit a message to a receiver. The warden is the logical instance that monitors the communication and filters inappropriate messages. The warden thus has to forward the message but will only do so if he approves with the message's contents. Obviously, the warden insists on the communication being unencrypted. Furthermore, if the sender uses signatures to authenticate his messages, the warden will only forward them if he can be sure that the sender has not embedded subliminal information in the signatures. Hence, the question arises how the signer can prove subliminal-freeness to the warden. It goes without saying that the warden must not be able to forge signatures on behalf of the sender.

Table 1 compares the different methods described below with respect to their benefits and drawbacks.

[3] `http://nginx.org/`

[4] `https://boringssl.googlesource.com/boringssl/`

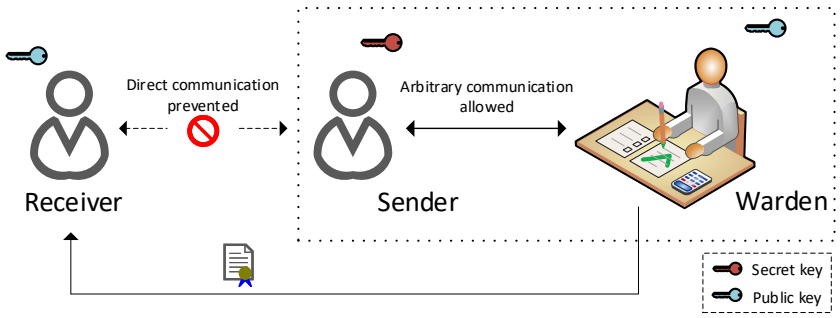

Figure 4: The warden scenario for preventing subliminal communication.

7.1 Preannounced Nonce Points

In some situations the following simple mitigation strategy might be sufficient. Instead of computing the nonce r from the message for each signature, the signer computes a number of nonces and corresponding nonce points R during key generation and announces them in an ordered list to the warden. Then, during signing, the signer uses the nonces in the same order as announced before, leaving just one valid signature for a given message.

Due to its drawbacks, the use cases for this method are very limited, though. The number of signatures that can be generated is limited by the number of announced nonce points. Furthermore, high storage requirements result for the warden and a large subliminal bandwidth results during key generation. Also, the approach introduces a state into signing, which might cause security issues.

7.2 An Interactive Approach

[33] describe a scheme for rendering Schnorr signatures provably subliminal-free by exchanging a total of six messages between signer and warden. Hence, the warden actively contributes in signature generation. The scheme is applicable to EdDSA as well, as EdDSA is based on Schnorr signatures.

Drawbacks of the method are the need for bidirectional communication between signer and warden, increased computational requirements and additional latency. These requirements conflict with application scenarios where high-speed signature schemes like EdDSA are typically used.

Table 1: Methods for preventing subliminal communication when using EdDSA.

Preannounced Nonce Points	**Interactive Method [33]**	**Non-Interactive Method [11]**
+ Simple	+ Small bandwidth requirements	+ Simple communication pattern
+ Low computational requirements	− Participation of warden required	+ Feasible for offline scenarios
− Limited number of transmitted messages	− Several messages need to be exchanged	− Huge proof size
− Subliminal channel during list computation	− Need for bidirectional communication	− Significant computational requirements
− Storage requirements for warden		

7.3 A Non-Interactive Approach

A method for making ECDSA subliminal-free with proof was described by [11]. The method works by proving that the nonce was computed deterministically from the message without disclosing the nonce value itself. Thus, it can equally be used for EdDSA.

For generating a nonce value, first a hash value $\boldsymbol{h} \in \{0,1\}^m$ is computed from the message. The nonce is then derived using M. Noar's pseudo random function [34] as

$$r(\boldsymbol{h}) = g^{a_{m+1}\prod_{1\leq i\leq m, h_i=1} a_i} \bmod p \bmod L.$$

In this equation, p is a prime number and g is the generator of a cyclic group of prime order q. The vector $\boldsymbol{a} \in \mathbb{Z}_q^{m+1}$ is an additional secret for signature generation.

The signer computes commitments for $\boldsymbol{a}$ and shows them to the warden during key generation. During signing the signer can then compute zero-knowledge proofs that show that the nonce has in fact been computed in the correct way. Subliminal-freeness is proven only for the signature itself. The proof could thus contain a subliminal channel and must be stripped off by the warden after verification.

Beneficial about this method is the simple unidirectional communication pattern which even qualifies for certain offline scenarios. In fact, the authors proposed the method for proving that the issuing instance of passports has not embedded any subliminal message. On the downside, the method has significant bandwidth demands due to the huge proof size of several megabytes. Also the computational requirements are large.

The method is suitable, furthermore, for situations where is suffices to test a random sample of generated signatures for subliminal-freeness. In this case the signer does not have to produce proofs for all signatures but only has to use M. Noar's pseudo random function for generating r. When the warden wants to check a signature he has intercepted for subliminal-freeness, he requests a proof for the signed message. As the nonce is computed deterministically from the message, the signer can recreate the same signature and compute a proof.

7.4 Detection Techniques

As it is not possible to prevent the subliminal channel without introducing significant drawbacks, the question arises if subliminal communication can at least be detected with a certain probability when investigating a set of signatures. This is possible only under very specific conditions.

Due to the deterministic signing process a legitimate signer will generate the same signatures when signing the same message multiple times. However, when the subliminal channel is exploited with varying subliminal information, different signatures result. The subliminal channel can thus be detected in this case, because in normal operation an equal signature would be expected. With enough effort the signer can circumvent this detection technique, however by ensuring to include subliminal messages only in messages that differ.

Furthermore, detection is easy by someone in a special administratory position who holds the secret key k. In this case the secret key can simply be used to test if the nonce was generated in the usual way.

Apart from these techniques detection is only possible if the subliminal sender naively encodes the information into r. In this case a guessable subliminal message or transmitting the same subliminal message multiple times results in disclosure of the signing key a as described in Section 3. A smart adversary will apply encryption with a suitable operational mode, so the nonces will be indistinguishable from random data.

8 MQ Signature Schemes

Signature schemes need to be based on problems that are mathematically hard to solve. The problem of solving multivariate quadratic polynomials in finite fields (MQ problem) qualifies for this and can be used for constructing signature schemes. One advantage of the MQ problem is that it allows to generate signature schemes that resist quantum computers. Furthermore, several MQ signature schemes can be used as high-speed signatures, e.g., in the scenarios described in Section 5. However, the existence of subliminal channels can lead to a vast bandwidth for data exfiltration.

In this section we analyze MQ signature schemes for the possibility for subliminal data transmission. After describing the basic principles of MQ signature schemes, we show the exploitable subliminal channels and describe which subliminal bandwidths can be achieved in some existing MQ signature schemes.

8.1 Basics

Fig. 5 depicts the basic functioning of MQ signature schemes [35]. The secret key usually consists of two bijective affine mappings $S : \mathbb{F}_q^m \to \mathbb{F}_q^m$ and $T : \mathbb{F}_q^n \to \mathbb{F}_q^n$ and a central quadratic mapping $F : \mathbb{F}_q^m \to \mathbb{F}_q^n$. Here, $\mathbb{F}_q$ denotes the Galois field of order q and $m,n \in \mathbb{N}$. When needed, we will write S and T as $S(\boldsymbol{s}) = \boldsymbol{S}^{(l)}\boldsymbol{s} + \boldsymbol{S}^{(a)}$ and $T(\boldsymbol{y}) = \boldsymbol{T}^{(l)}\boldsymbol{y} + \boldsymbol{T}^{(a)}$, where $\boldsymbol{S}^{(l)}$ and $\boldsymbol{T}^{(l)}$ denote the linear parts and $\boldsymbol{S}^{(a)}$ and $\boldsymbol{T}^{(a)}$ denote the affine parts of S and T, respectively. The public key consists of the composition $P = T \circ F \circ S$.

For signing and verification a cryptographic hash function H is applied to the message, yielding the value $\boldsymbol{h} = H(M)$, where $\boldsymbol{h} \in \mathbb{F}_q^n$. In the course of signature generation the signer has to find a vector $\boldsymbol{s} \in \mathbb{F}_q^m$, so that $\boldsymbol{h} = P(\boldsymbol{s})$. In order to find such a vector, the signer first computes $\boldsymbol{y} = T^{-1}(\boldsymbol{h})$. In the next step he tries to find a vector $\boldsymbol{x} \in \mathbb{F}_q^m$, for which $\boldsymbol{y} = F(\boldsymbol{x})$ holds. F is a quadratic function. Hence, if it consisted of polynomials with random coefficients, this problem would be as hard as solving $\boldsymbol{h} = P(\boldsymbol{s})$ in the first place and infeasible to solve. However, F is constructed with a particular structure that allows inversion in a straight-forward manner as discussed in Section 8.2. Finally, knowing $\boldsymbol{x}$, the signer is able to find $\boldsymbol{s}$ as $\boldsymbol{s} = S^{-1}(\boldsymbol{x})$.

Knowing a signature $\boldsymbol{s}$, verification is simple. By applying $P = T \circ F \circ S$ to $\boldsymbol{s}$, the signed message hash $\boldsymbol{h}$ can be regained and compared to the actual hash $H(M)$.

An attacker who tries to forge a signature only has P and is confronted with the problem of finding an $\boldsymbol{s}$ that solves $\boldsymbol{h} = P(\boldsymbol{s})$. If these quadratic polynomials had random coefficients, this would be hard, corresponding to the MQ problem. The attack approaches developed so far for many MQ signature schemes target at exploiting the structure of F that is meant to be hidden by the use of S and T.

Unfortunately, attacks have been found against all trapdoors found so far. Instead of devising completely new constructions it has thus become common to modify existing (broken) schemes to yield a more secure signature scheme. The most promising candidate for constructing a secure signature scheme is the Hidden Field Equations (HFE) trapdoor (see Section 8.2) in conjunction with the minus and vinegar variables modifications (see Section 8.3).

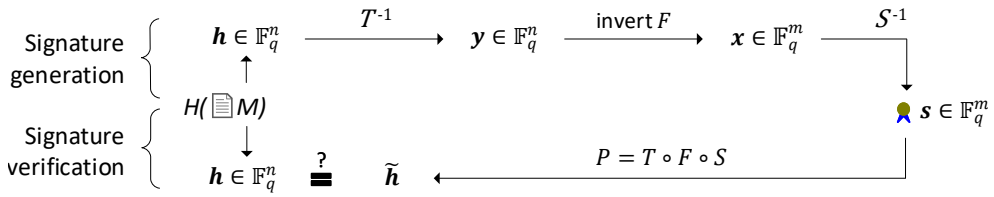

Figure 5: The basic operation principle of MQ signature schemes.

8.2 Trapdoors

There are different approaches for constructing the central mapping F. The survey paper [35] gives a good overview of the most important ones. In this section the approaches will be discussed but not described in too much detail. Without any modifications attacks have been found against all of the below trapdoors.

In addition to the trapdoors below, Stepwise Triangular Systems (STS) have been proposed, that use a layered approach for constructing F [36]. While these schemes have been broken when used on their own, they can be combined with other trapdoors like it was done for Rainbow [37].

Unbalanced Oil and Vinegar The initial idea of schemes based on vinegar variables was [38]. In the course of signature generation the signer has to invert the equations

$$F_i(\hat{\boldsymbol{x}}, \boldsymbol{z}) = \sum_{j=1}^{n}\sum_{k=1}^{v} \gamma_{ijk}\hat{x}_j z_k + \sum_{j=1}^{v}\sum_{k=1}^{v} \lambda_{ijk} z_j z_k + \sum_{j=1}^{n} \xi_{ij}\hat{x}_j + \sum_{j=1}^{v} \xi'_{ij} z_j + \delta_i, \tag{1}$$

where $\gamma_{ijk}, \lambda_{ijk}, \xi_{ij}, \xi'_{ij}, \delta_i \in \mathbb{F}_q$ are chosen randomly by the signer during key generation. Here, $\boldsymbol{z} \in \mathbb{F}_q^v$ denote the $v \in \mathbb{N}$ vinegar variables that are chosen at random by the signer for each signature. Fixing these variables the equations (1) turn into an affine equation system in the variables $\hat{x}_i$, that are referred to as oil variables. The equation system is nonsingular with non-negligible probability and, when this is the case, can easily be inverted by the signer. Otherwise he trys different values for the vinegar variables until he finds a solution. Finally, the signature is formed by transforming both the oil and the vinegar variables using the inverse of the secret affine mapping $S : \mathbb{F}_q^{n+v} \to \mathbb{F}_q^{n+v}$, thus hiding the vinegar variables. Note that for the notation introduced in Fig. 5, we set $m = n + v$ and $\boldsymbol{x}^T = (\hat{\boldsymbol{x}}^T \boldsymbol{z}^T)$.

While we have as many oil variables as vinegar variables in the case of the initial, balanced scheme [38], we have more vinegar than oil variables for Unbalaned Oil and Vinegar schemes [39], which were proposed after the cryptanalysis of balanced UOV to yield more security.

Matsumoto-Imai Scheme The Matsumoto-Imai Scheme (C^*) was first described in [40, 41]. It uses an extension field $\mathbb{E}$ of $\mathbb{F}_q$ to define the mapping F. The central polynomials F are expressed as $\Phi \circ \tilde{F} \circ \Phi^{-1}$, where $\Phi : \mathbb{E} \to \mathbb{F}_q^n$ is a secret bijection between an n-dimensional vector of the ground field and the extension field $\mathbb{E}$ of degree n. In the case of C^*, $\tilde{F}$ is the monomial

$$\tilde{F}(\tilde{x}) = \tilde{x}^{q^\lambda + 1}, \tag{2}$$

where $\tilde{x} \in \mathbb{E}$ and λ is an integer, so that $q^n - 1$ and $q^\lambda + 1$ are coprime. The latter condition ensures that equation (2) can easily be solved for $\tilde{x}$.

Hidden Field Equations Similarly to C^*, Hidden Field Equations (HFE) uses an extension field $\mathbb{E}$ to define F. After the cryptanalysis of C^* the trapdoor has been generalized to HFE by [42]. Instead of a monomial it uses a polynomial

$$\tilde{F}(\tilde{x}) = \sum_{\substack{0 \le i,j \le d \\ q^i + q^j \le d}} C_{i+1,j+1}\tilde{x}^{q^i + q^j} + \sum_{\substack{0 \le i \le d \\ q^i \le d}} B_{i+1}\tilde{x}^{q^i} + A \tag{3}$$

with $C_{i,j}, B_i, A \in \mathbb{E}$ and a degree $d \in \mathbb{N}$. In theory, the mapping could be constructed to be a bijection by choosing a permutation polynomial for $\tilde{F}(\tilde{x})$. However, J. Patarin assumed this to be very difficult, so the coefficients of $\tilde{F}(\tilde{x})$ are usually chosen randomly. Hence, the most important difference to C^* in this

context is that in general F is no bijection anymore. This means that for some message there may be multiple signatures and for others there may be none at all. In order to be able to sign all messages, it is necessary to include randomness in the signed data using, for example, the minus or vinegar variables modifications described in Section 8.3. At present Hidden Field Equations (HFE) is one of the most preferred candidates for constructing secure signature schemes by applying appropriate modifications.

Multivariate Quadratic Quasigroups D. Gligoroski proposed a trapdoor based on Multivariate Quadratic Quasigroups (MQQ) in [43, 44]. We here describe the slightly modified variant described in [45].

In MQQ the mapping $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is constructed from a quasigroup $(G, *)$ of order 2^d with $d \in \mathbb{N}$ a divisor of n as follows: First the input vector $\boldsymbol{x} \in \mathbb{F}_2^n$ is divided into n/d vectors in $\mathbb{F}_2^d$, which are interpreted as n/d group elements $X_1, \ldots, X_{n/d} \in G$. The variables $Y_1, \ldots, Y_{n/d} \in G$ are then obtained according to

$$Y_i = \begin{cases} X_i, & \text{if } i = 1 \\ X_{i-1} * X_i, & \text{if } i > 1 \wedge i \text{ odd} \\ X_i * X_{i-1}, & \text{if } i > 1 \wedge i \text{ even} \end{cases} . \tag{4}$$

Quasigroups that are used in MQQ show the bilinearity property and allow the operation $*$ to be executed in terms of simple matrix multiplications.

Finally, the variables Y_i are interpreted as n/d vectors in $\mathbb{F}_2^d$, which are concatenated to form the output $\boldsymbol{y} \in \mathbb{F}_2^n$ of the mapping $F(\boldsymbol{x})$.

8.3 Modifications

Attacks have been found for all (unmodified) trapdoors described above. It has thus become common to apply modifications to these trapdoors to attain a (more) secure signature scheme. In this section only modifications are listed that aim to improve the schemes' security and not to improve their performance or key sizes. Fig. 6 provides an illustration of signature generation and verification if the vinegar and minus modifications are used, like it is the case for, for example, HFEv-.

The Minus Modification One of the most popular methods is simply removing some of the public key polynomials [35] allowing only a part of the message hash to be recovered from the signature using the public key. Despite being very simple the modification prevents important attacks against MQ schemes and has been used in many proposed signature schemes.

Vinegar Variables The HFE trapdoor can be enhanced by the use of vinegar variables [39]. In this case the signer picks $v \in \mathbb{N}$ vinegar variables $\boldsymbol{z} \in \mathbb{F}_q^v$ at random and the polynomial (3) is adapted according to

$$\tilde{F}(\tilde{x}, \boldsymbol{z}) = \sum_{\substack{0 \le i,j \le d \\ q^i + q^j \le d}} C_{i+1,j+1} \tilde{x}^{q^i + q^j} + \sum_{\substack{0 \le i \le d \\ q^i \le d}} B_{i+1}(\boldsymbol{z}) \tilde{x}^{q^i} + A(\boldsymbol{z}), \tag{5}$$

so B_i and A are now functions of the vinegar variables. Similar to Unbalanced Oil and Vinegar (UOV), $\hat{\boldsymbol{x}} = \Phi(\tilde{x})$ and $\boldsymbol{z}$ are transformed together by the affine mapping $S : \mathbb{F}_q^{n+v} \to \mathbb{F}_q^{n+v}$ to form the signature. To retain MQ shape, $B_i(\boldsymbol{z})$ have to be affine functions and $A(\boldsymbol{z})$ has to be a quadratic function.

Further Modifications Further modifications that can be used are: Projection (or "fixing") described in [46], the plus Modification [47] or internal perturbation [35, 48].

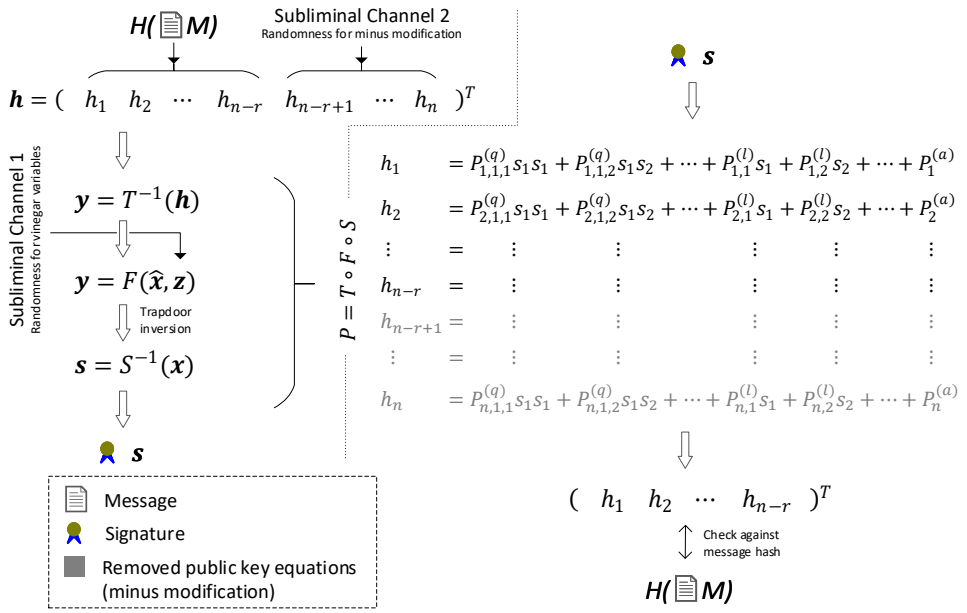

Figure 6: Signature generation (left) and verification (right) when deploying the vinegar variables and minus modifications.

9 Subliminal Channels in MQ Signatures

9.1 Randomness in MQ Signatures

The basic operation principle of MQ signature schemes depicted in Section 8.1 creates the impression that these schemes deterministically map a given message to a signature, leaving no space for subliminal information. Nevertheless, almost all MQ signature schemes devised so far explicitly use randomness throughout the signing process. Among the schemes that are currently considered secure, all use randomness.

Reasons for introducing randomness in the signing process can be classified into two groups. The first group of methods explicitly includes random data in the signature to achieve its security. The second group does not directly introduce randomness but reduces the probability of the central mapping to be invertible for a message. Hence, randomness has to be included in the signature to make sure that the signer is able to find a suitable signature.

9.1.1 Explicit Randomness

The group of methods that explicitly introduce randomness consists of the UOV trapdoor, the minus modification and the vinegar variables modification. When using schemes with vinegar variables $\boldsymbol{z}$, the signer chooses random values for $\boldsymbol{z}$ and tries to invert the central mapping F with this choice. Exploiting $\boldsymbol{z}$ to encode information, the vinegar variables can be used as subliminal channel. However, most constructions require multiple runs of the inversion of F with different choices of vinegar variables to be able to find one signature with large-enough probability. Hence, few of the vinegar variables have to remain free in order to be able to vary them to find a valid signature.

9.1.2 Loss of Surjectivity

There are many methods for constructing MQ signature schemes that cause the trapdoor to not be surjective. This means that, without any further measures, there would be messages for which no signature exists.

This group includes the HFE trapdoor, the projection modification, the internal perturbation modification and the plus modification. As described in Section 8.2 the HFE trapdoor would require a permutation polynomial for $\tilde{F}(\tilde{x})$ to constitute a bijective mapping. In order to guarantee the existence of a signature with large-enough probability for every message, randomness has to be included during signature generation. Usually this is accomplished by using either the minus modification or the vinegar variables modification.

It is interesting to note that the modifications that lead to the loss of surjectivity, can not only cause the existence of a subliminal channel but may also *reduce* the subliminal bandwidth that results from, for example, the minus or vinegar variables modifications. This is because the probability of finding a signature with a particular choice of random data is reduced, requiring the signer to be able to vary more variables to find a signature with large-enough probability. Yet, use of these modifications for this sole purpose is usually not justified, as they come with significant drawbacks in terms of signing speed.

9.1.3 Subliminal-Free Trapdoors

Comparing the methods mentioned in the prior two sections with Section 8.2 we see that only the C^*, STS and MQQ trapdoors in unmodifed form and not a single modification are suitable to construct subliminal-free signatures. This is because these trapdoors are specifically constructed to be bijective. But all signature schemes using these unmodified trapdoors suitable for subliminal-free signatures, have been broken.

For the sake of completeness, for the particular case of STS we note that bijectivity only applies to regular STS, which is constructed to have a bijective mapping in each layer. If, for example, a layer of a general STS construction yields more unknowns than equations, multiple signatures can be found for a message.

9.2 MQ Subliminal Channel 1: Recovering Vinegar Variables

Both the UOV trapdoor and the vinegar varibles modification have in common that, for the purpose of signature generation, the signer first picks v vinegar variables at random and afterwards inverts the central mapping $F(\hat{\boldsymbol{x}}, \boldsymbol{z}) = \boldsymbol{y}$ using this particular choice of vinegar variables. Knowing $\hat{\boldsymbol{x}}$ and $\boldsymbol{z}$ the signer can then compute the signature by using the inverse of the affine mapping S, i.e. the signer computes $\boldsymbol{s} = \boldsymbol{S}^{(l)^{-1}}(\boldsymbol{x} - \boldsymbol{S}^{(a)})$. Evidently, knowing S it is easy for the signer to recover the vinegar variables as

$$\boldsymbol{x} = \begin{pmatrix} \hat{\boldsymbol{x}} \\ \boldsymbol{z} \end{pmatrix} = \boldsymbol{S}^{(l)}\boldsymbol{s} + \boldsymbol{S}^{(a)}, \tag{6}$$

or, partitioning $\boldsymbol{S}^{(l)}$ as $\boldsymbol{S}^{(l)} = \begin{pmatrix} \boldsymbol{S}_x \\ \boldsymbol{S}_z \end{pmatrix}$ with $\boldsymbol{S}_x \in \mathbb{F}_q^{n \times m}$ and $\boldsymbol{S}_z \in \mathbb{F}_q^{v \times m}$, the vinegar variables can be computed as $\boldsymbol{z} = \boldsymbol{S}_z\boldsymbol{s} + \boldsymbol{S}^{(a)}$. Hence, it suffices if the signer passes the subliminal receiver the affine mapping consisting of $\boldsymbol{S}_z$ and $\boldsymbol{S}^{(a)}$ upfront to allow him to recover the vinegar variables and exploit them as a very efficient subliminal channel.

It is not surprising, however, that by sharing $\boldsymbol{S}_z$ and $\boldsymbol{S}^{(a)}$ the signature scheme's security is significantly reduced considering attacks performed by the subliminal receiver. To see that, we note that by setting $\boldsymbol{z}$ to some arbitrary value the subliminal receiver is able to obtain a linear equation system in s_i

Table 2: Subliminal bandwidths of proposed MQ signature schemes.

Scheme	Trapdoor	Broken	Signature length	Subliminal bandwidth	
QUARTZ [49]	HFEv-	no	128	~12	(9%)
Gui-127 [50]	HFEv-	no	163	~24	(15%)
Rainbow [37]	UOV-, STS	yes	264	~46	(17%)
SFlash [51]	C^*-	yes	469	77	(16%)
PFlash(GF16,94,30,1) [4]	pC^*-	no	372	~108	(29%)
MQQ-SIG (256) [45]	MQQ-	yes	256	128	(50%)

from $\boldsymbol{z} = \boldsymbol{S}_z\boldsymbol{s} + \boldsymbol{S}^{(a)}$. He can, hence, express v of the variables s_i and substitute them into the public key equations. The resulting altered public key is the public key of the original key with fixed vinegar variables. However, by fixing the vinegar variables, the HFEv signature system Eq. 5 reverts to the original HFE shape Eq. 3. The new public key therefore corresponds to unmodified HFE and can be attacked with key recovery attacks that have been found for HFE. For the UOV trapdoor the central mapping Eq. 1 even turns into an affine mapping, making it trivial for the subliminal receiver to forge signatures for arbitrary messages.

Hence, to retain a certain level of security against attacks performed by the subliminal receiver, the signer must not pass on all lines of $\boldsymbol{S}_z$ and, therefore, cannot use all vinegar variables to encode subliminal information. If there is not sufficient trust among the subliminal sender and the subliminal receiver, the sender thus has to find a tradeoff between achievable subliminal bandwidth and security against attacks performed by the subliminal receiver.

9.3 MQ Subliminal Channel 2: Using the Minus Modification

The minus modification is a popular method for improving an MQ signature scheme's security. The public key of an unmodified MQ signature scheme consists of n multivariate equations $h_i = P_i(\boldsymbol{s})$, where $i = 1,\ldots,n$.

When deploying the minus modification we remove r of the equations from the public key. The corresponding portion of the message hash $\boldsymbol{h}$ can be filled with arbitrary data by the signer. In fact, the signer even has to fill it with random data as otherwise an adversary is able to reconstruct the removed public key equations as soon as he has observed enough signed messages [45].

The public key of the modified scheme now consists of the equations $h_i = P_i(\boldsymbol{s})$ with $i = 1,\ldots,(n-r)$ and the removed equations read $\rho_i = P_{n-r+i}(\boldsymbol{s})$, where $i = 1,\ldots,r$ and $\boldsymbol{\rho} \in \mathbb{F}_q^r$ denotes the random data. Hence, using the unmodified public key, $\boldsymbol{\rho}$ can easily be recovered from the signature, making it possible to use $\boldsymbol{\rho}$ as subliminal channel. To use this subliminal channel, the signer has to share the removed equations $P_{n-r+1}(\boldsymbol{s}),\ldots,P_n(\boldsymbol{s})$ with the subliminal receiver upfront.

Obviously, by receiving the removed equations the modified scheme reverts to the unmodified scheme for the subliminal receiver. Hence, for the subliminal receiver attacks become possible that were meant to be prevented by the use of the minus modification. The signer can retain a certain level of security, however, by using only part of the removed equations as subliminal channel and keeping the remaining ones secret. Hence, also in this case a tradeoff between subliminal bandwidth and security against attacks performed by the subliminal receiver can be achieved.

9.4 Examples for Subliminal Channels in Existing MQ Signature Schemes

To get an understanding of how much data can be exfiltrated using subliminal channels in MQ signature schemes we analyze algorithms for which an implementation exists or at least a practical set of parameters has been proposed. It shall be stressed that we aim to analyze techniques that are used for constructing MQ signature schemes rather than concrete algorithms. Hence, even though many of the schemes described below have been broken, the results have a certain relevance for signature schemes that are going to be developed and are likely to be constructed in a similar way. Table 2 shows the results.

In the following we do not distinguish between truly random data and pseudorandom data that was deterministically derived from the message using a cryptographic hash function and a secret key (see e.g. the calculation of r in Section 3).

QUARTZ [49] uses the HFE trapdoor in conjunction with the minus modification and vinegar variables modification. To avoid birthday attacks, QUARTZ uses 4 rounds and uses the trapdoor in each round. The corresponding portion of the message hash and the vinegar variables are filled with random data. Hence, subliminal information can be embedded and recovered as described in Section 8.3. This way 7 bits of subliminal information (4 for vinegar variables and 3 due to the minus modification) could be embedded each round. It has to be noted, however, that, as described in Section 8.2 the trapdoor cannot be inverted for each possible choice of random data. The signer therefore has to be able to vary it to a certain degree to perform multiple attempts to eventually find a signature, varying the random data each time. In fact, the probability for the trapdoor to not be invertible in one round for a particular choice of random data is $1/e$ [49]. Reserving B_r bits per round for subliminal information we can try 2^{7-B_r} different values for the random data in each round. Having 4 rounds we then have a probability of

$$P_f = 1 - \left(1 - \left(\frac{1}{e}\right)^{2^{7-B_r}}\right)^4 \tag{7}$$

for a signature to fail. For a (arbitrarily chosen) value of $P_f = 10^{-6}$ we obtain $B_r = 3.07 \approx 3$ bit per round and a total subliminal bandwidth of 12bit.

The Gui signature scheme [50] is an improved version of QUARTZ and has a very similar design. The authors propose three different parametrizations of the scheme. Gui-127, which achieves a security level of 123, operates in 4 rounds like QUARTZ but it uses in each round 6 vinegar variables and removes 4 equations, hence achieving 10 of subliminal bandwidth per round when neglecting the need to invert the trapdoor multiple times. Using the same calculation as above, we obtain $B_r = 6.07 \approx 6$ bit per round and a total subliminal bandwidth of 24.

Rainbow [37] uses the UOV trapdoor in a layered approach. We can consider it as a combination of the UOV and STS trapdoors. In a total of 4 layers the scheme uses 6, 12, 17 and 22 vinegar variables, respectively. Here the vinegar and oil variables of layer i are the vinegar variables of layer $i+1$, which is why randomness is only used for the initial 6 vinegar variables. Having the appropriate equations these variables can be fully reconstructed from a signature. Hence, as the scheme operates in $\mathbb{F}_{256}$, this corresponds to a maximum subliminal bandwidth of 48bit. Also in this case the invertibility of the trapdoor is not guaranteed (see Section 8.2) narrowing the exploitable bandwidth. The probability for the matrix forming a UOV trapdoor to be invertible is $\prod_{n=1}^{N-1}(1-q^{-n})$, where q denotes the order of the Galois field and N the dimension of the matrix [38]. In our case we therefore obtain a probability of

$$P_S = \prod_{n=1}^{11}\left(1-\frac{1}{256^n}\right)\prod_{n=1}^{16}\left(1-\frac{1}{256^n}\right)\prod_{n=1}^{21}\left(1-\frac{1}{256^n}\right)\prod_{n=10}^{32}\left(1-\frac{1}{256^n}\right) \tag{8}$$

for the signature generation to succeed for a particular choice of vinegar variables. Reserving B_s bits for subliminal information we thus obtain a probability of $(1-P_S)^{2^{48-B_s}}$ for no signature to exist. Again

targeting a probability of 10^{-6} we obtain

$$B_s = 48 - \log_2 \frac{-6}{\log_{10}(1-P_S)} \approx 46 \text{ bit} \tag{9}$$

as subliminal bandwidth.

Furthermore, we analyzed two schemes that use the minus modification and use trapdoors with garantued invertability, independed of the choice of the random variables. Hence, all random bits can be used for subliminal information. One is the SFlash scheme [51]. The scheme was recommended by the NESSIE project for low cost smart cards (and broken shortly afterwards), where efficient operation is of high importance, but the size of the public key is not an issue [52]. The scheme uses the C^* trapdoor and removes 11 equations from the public key. As it operates in $\mathbb{F}_{128}$ we obtain a subliminal bandwidth of 77.

The PFlash signature scheme [4] extends SFlash by the use of projection, fixing one of the signature variables. The probability of finding a signature with a particular choice of random data thus drops to q^{-1}, where for PFlash $q = 16$. Setting the probability of not being able to find a signature when using the subliminal channel to $(1-q^{-1})^{2^{4r-B_s}} = 10^{-6}$, we obtain for an exemplary parametrization with $n = 94$ and $r = 30$ a subliminal bandwidth of 108.

Finally, the MQQ trapdoor which was proposed in [43, 44] was enhanced by the minus modification after being broken. Hence, the authors propose to remove half of the public key equations. If we, for example, use the signature scheme to produce a signature with a length of 256bit an exploitable subliminal bandwidth of 128bit results.

9.5 MQQ-SIG: A Practical Experiment

In order to verify the practical feasibility of the above results, we investigated a subliminal channel in the MQQ-SIG signature scheme by D. Gligoroski and developed a proof-of-concept. The scheme was found in 2011 and broken 3 years later in [53]. MQQ-SIG uses the MQQ trapdoor and the minus modification to improve its security.

The scheme uses the usual composition $P = T \circ F \circ S$ for signature creation and verification. Here, T and S are a linear and an affine mapping, respectively, that are constructed from a random nonsingular matrix $\boldsymbol{S} \in \{0,1\}^{n\times n}$ and a random vector $\boldsymbol{v} \in \{0,1\}^n$ according to

$$T(\boldsymbol{y}) = \boldsymbol{S}\boldsymbol{y} \qquad \text{and}$$
$$S(\boldsymbol{s}) = \boldsymbol{S}\boldsymbol{s} + \boldsymbol{v}.$$

The signature scheme uses the MQQ trapdoor, i.e. the mapping $F : \{0,1\}^n \to \{0,1\}^n$ is constructed by interpreting the input vector as $n/8$ elements X_i of a quasigroup of order 2^8, which are mapped according to equation 4.

Following this construction P consists of n quadratic polynomials. The signer is able to find $D(\boldsymbol{y}) = P^{-1}(\boldsymbol{y})$. The second half of the polynomials of P forms the mapping $E(\boldsymbol{s})$, which is used as public key.

To create a signature, two cryptographic hash functions of length $n/2$ are applied to the message. The results are prefixed with random bits to have a length of n bits resulting in the vectors $\boldsymbol{h}_1, \boldsymbol{h}_2 \in \{0,1\}^n$. The signature consists of the vectors $\boldsymbol{s}_1 = D(\boldsymbol{h}_1)$ and $\boldsymbol{s}_2 = D(\boldsymbol{h}_2)$. For verification of a signature, $E(\boldsymbol{s})$ is applied to both $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ and the result is checked to be equal to the original hash values of the message. Fig. 7 illustrates the functioning of MQQ-SIG.

The scheme as described here uses the minus modification, which was introduced after the cryptanalysis of unmodified MQQ. Instead of all n polynomials just $n/2$ are published as public key and the corresponding parts of $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ are padded with random data. This way key recovery attacks like

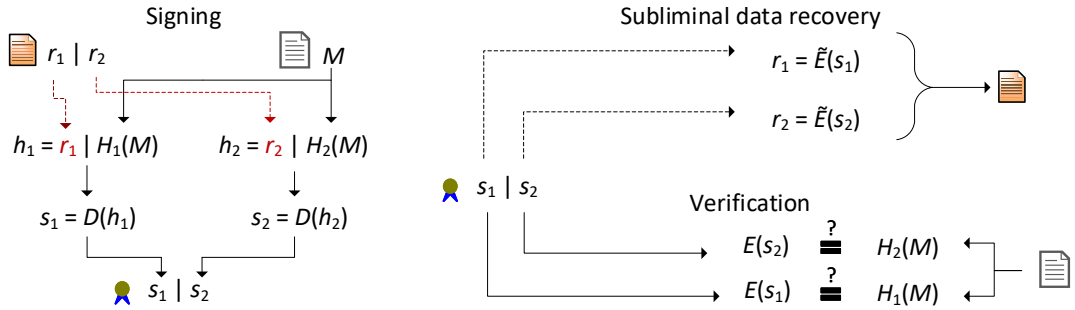

Figure 7: Embedment of subliminal channels into MQQ-SIG signatures.

[54, 55] shall be prevented. However, as described in Section 9.3, it is remarkably easy to use the signature as a subliminal channel. The signer just has to pass on the missing public key equations, which then can be used as a function $\tilde{E}(\boldsymbol{s})$ that recovers the subliminal information.

To verify that this subliminal channel works in practice, we used the reference implementation of the signature algorithms available from the SUPERCOP project[5] with a signature length of 256bit. We modified the key generation code to not only output half of the equations P as public key but also output a key for recovery of the subliminal information. Furthermore, we modified the signing algorithm to use the subliminal information as random data for signing. Then recovery of the subliminal information is straight-forward: The same algorithm as for verification can be reused, feeding it with the subliminal recovery key instead of the public key and outputting the subliminal information before comparison with the message hash. In this setting the subliminal channel was proven operational with the predicted subliminal bandwidth. By default, the implementation removed just 1/4 of the public key equations instead of half of them as described above. Hence, a subliminal bandwidth of 64bit was possible.

10 Discussion on MQ Signatures

Among all methods for constructing MQ signature schemes only very few showed to be subliminal-free. In particular, the approaches that are considered most secure today allow a large subliminal bandwidth. Furthermore, the embedding and recovery process works remarkably easy and efficiently in all algorithms that were considered.

It is noteworthy that the subliminal channel in this case may have particularly attractive properties for an attacker: In contrast to broadband subliminal channels of DSA-like signature schemes the possibility to decode the subliminal information does not directly coincide with the possibility of signing arbitrary messages. If the total available bandwidth is used, however, attacks become possible for the subliminal receiver allowing signature forging. On the other hand, by using just part of the bandwidth, a tradeoff can be achieved between subliminal bandwidth and security against attacks performed by the subliminal receiver. Furthermore, by passing on different parts of the set of hidden equations, it is possible to transmit different subliminal information to multiple receivers, who thereby are unable to decode the information that is not intended for them.

Even if a subliminal-free MQ signature scheme was found a further problem has to be considered: The subliminal-freeness can only be garantueed if the secret and public key are generated in the proper way. Here, the public key consists of a bijective mapping $P : \mathbb{F}_q^n \to \mathbb{F}_q^n$. Bijectivity ensures that a given message only has a single valid signature, rendering any subliminal communication impossible. However, without knowing how P was constructed, it cannot be verified if it indeed is a bijection and, hence,

[5] https://bench.cr.yp.to/supercop.html

if signing is indeed subliminal-free. This is similar to the RSA signature scheme where uniqueness of signatures can only be guarantueed if the signer's modulus is known to be the product of two primes [56, 57].

Finally, we note that mitigation strategies that use zero-knowledge proofs as described in Section 7 are significatly more difficult to construct for MQ signatures, if security in a post-quantum era is a reason for using these schemes. This is because many zero-knowledge proofs rely fundamentally on the hardness assumption of discrete logarithms. For post-quantum cryptography this assumption can no longer be assumed to hold.

11 Conclusion

We analyzed several high-speed signature schemes for the possibility of establishing subliminal channels and found that almost all considered schemes yield subliminal channels that may lead to the exfiltration of serious amounts of data.

Most notably, EdDSA signatures, which are already widely deployed, allow the subliminal transmission of 252 per signature, which is substantial. We showed several methods for ensuring subliminal-free EdDSA signatures, but have to conclude that none of them is practical for most real-world scenarios.

A further class of recent signatures that can offer attractive properties, such as high performance and post-quantum security, are MQ signatures. We outlined several methods that were proposed for constructing signatures based on MQ cryptography with significant future potential and demonstrated how basic constructions as well as modifications can be exploited for establishing subliminal channels.

In order to demonstrate the impact of our findings, we presented scenarios where the use of high-speed signatures allows a considerable bandwidth for data leakage. These scenarios include in particular new application areas for signatures like clock synchronization and smart grid sensor data collection. We backed our argumentation by experiments that showed that subliminal channels can easily and efficiently be exploited with the expected subliminal bandwidth.

Security engineers, network operators and protocol designers should be aware of the subliminal channels described in this paper when deciding about suitable signatures in their application areas. In cases where subliminal channels can not be tolerated and where a potential clandestine information exchange or data leakage can pose a substantial threat, they have to fall back on different, subliminal-free signature schemes, even if this means that they loose several attractive properties of the modern high-speed signatures analyzed in this paper.

References

[1] G. J. Simmons, "Subliminal communication is easy using the DSA," in *Proc. of the 10th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'93), Lofthus, Norway*, ser. Lecture Notes in Computer Science, vol. 765. Springer-Verlag, May 1993, pp. 218–232.

[2] G. J. Simmons, "The prisoners' problem and the subliminal channel," in *Advances in Cryptology - Proc. of the 1983 International Cryptology Conference (CRYPTO'83), Boston, Massachusetts, USA*, ser. Lecture Notes in Computer Science. Springer, Boston, Massachusetts, January 1984, vol. 1440, pp. 51–67.

[3] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," in *Proc. of the 13th International Workshop on Cryptographic Hardware and Embedded Systems (CHES'11), Nara, Japan*, ser. Lecture Notes in Computer Science, vol. 6917. Springer-Verlag, September 2011, pp. 124–142.

[4] M.-S. Chen, B.-Y. Yang, and D. Smith-Tone, "PFLASH - Secure asymmetric signatures on smart cards," in *Proc. of the Lightweight Cryptographic Workshop 2015, Gaithersburg, Maryland, USA*. NIST, July 2015.

[5] A. Hartl, R. Annessi, and T. Zseby, "A subliminal channel in EdDSA: Information leakage with high-speed signatures," in *Proc. of the 9th ACM CCS International Workshop on Managing Insider Security Threats (MIST'17), Dallas, Tesxas, USA*. ACM, October 2017, pp. 67–78.

[6] G. J. Simmons, "The subliminal channel and digital signatures," in *Proc. of the 1984 Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'84), Paris, France*, ser. Lecture Notes in Computer Science. Springer-Verlag, April 1984, vol. 209, pp. 364–378.

[7] X. Zhao and N. Li, "Reversible watermarking with subliminal channel," in *Proc. of the 10th International Workshop on Information Hiding (IH'08), Santa Barbara, California, USA*, ser. Lecture Notes in Computer Science, vol. 5284. Springer-Verlag, May 2008, pp. 118–131.

[8] J.-M. Bohli and R. Steinwandt, "On subliminal channels in deterministic signature schemes," in *Proc. of the 7th International Conference on Information Security and Cryptology (ICISC'04), Seoul, Korea*, ser. Lecture Notes in Computer Science, vol. 3506. Springer-Verlag, December 2005, pp. 182–194.

[9] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, February 1978.

[10] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Advances in Cryptology - Proc. of the 1984 International Cryptology Conference (CRYPTO'84), Santa Barbara, California, USA*, ser. Lecture Notes in Computer Science, vol. 196. Springer-Verlag, August 1984, pp. 10–18.

[11] J.-M. Bohli, M. I. G. Vasco, and R. Steinwandt, "A subliminal-free variant of ECDSA," in *Proc. of the 8th International Workshop on Information Hiding (IH'06), Alexandria, Virginia, USA*, ser. Lecture Notes in Computer Science, vol. 4437. Springer-Verlag, July 2006, pp. 375–387.

[12] Q. Dong and G. Xiao, "A subliminal-free variant of ECDSA using interactive protocol," in *Proc. of the 2010 International Conference on E-Product E-Service and E-Entertainment (ICEEE'10), Henan, China*. IEEE, November 2010.

[13] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *International Journal of Information Security*, vol. 1, no. 1, pp. 36–63, August 2001.

[14] R. Anderson, S. Vaudenay, B. Preneel, and K. Nyberg, "The Newton channel," in *Proc. of the 1st International Workshop on Information Hiding (IH'96), Cambridge, U.K.*, ser. Lecture Notes in Computer Science, vol. 1174. Springer-Verlag, May-June 1996, pp. 151–156.

[15] A. Young and M. Yung, "The dark side of "black-box" cryptography or: Should we trust capstone?" in *Advances in Cryptology - Proc. of the 16th Annual International Cryptology Conference (CRYPTO'96), Santa Barbara, California, USA*, ser. Lecture Notes in Computer Science, vol. 1109. Springer-Verlag, August 1996, pp. 89–103.

[16] M. Bellare, K. G. Paterson, and P. Rogaway, "Security of symmetric encryption against mass surveillance," in *Advances in Cryptology – Proc of the 14th International Cryptology Conference (CRYPTO'14), Santa Barbara, California, USA*, ser. Lecture Notes in Computer Science, vol. 8616. Springer-Verlag, August 2014, pp. 1–19.

[17] G. Ateniese, B. Magri, and D. Venturi, "Subversion-resilient signature schemes," in *Proc. of the 22nd ACM Conference on Computer and Communications Security (CCS'15), Denver, USA*. ACM, October 2015, pp. 364–375.

[18] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Advances in Cryptology - Proc. of the 1989 International Cryptology Conference (CRYPTO'89), Santa Barbara, California, USA*, ser. Lecture Notes in Computer Science, vol. 435. Springer-Verlag, August 1989, pp. 239–252.

[19] S. Josefsson and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)," IETF RFC 8032, January 2017, http://www.ietf.org/rfc/rfc8032.txt.

[20] D. J. Bernstein, "Curve25519: New Diffie-Hellman speed records," in *Proc. of the 9th International Conference on Theory and Practice in Public-Key Cryptography (PKC'06), New York, USA*, ser. Lecture Notes in Computer Science, vol. 3958. Springer-Verlag, April 2006, pp. 207–228.

[21] M. Hamburg, "Ed448-Goldilocks, a new elliptic curve," *IACR Cryptology ePrint Archive*, June 2015.

[22] R. Annessi, J. Fabini, and T. Zseby, "SecureTime: Secure multicast time synchronization," May 2017.

[23] E. Itkin and A. Wool, "A security analysis and revised security extension for the precision time protocol," *IEEE Transactions on Dependable and Secure Computing*, September 2017, online published, doi:

10.1109/TDSC.2017.2748583.

[24] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2," IETF RFC 5246, August 2008, http://www.ietf.org/rfc/rfc5246.txt.

[25] E. Rescorla, "The Transport Layer Security (TLS) protocol version 1.3," IETF Internet-draft (work in progress), March 2018, https://tools.ietf.org/html/draft-ietf-tls-tls13-26.

[26] Y. Nir, S. Josefsson, and M. Pegourie-Gonnard, "Elliptic Curve Cryptography (ECC) cipher suites for Transport Layer Security (TLS) versions 1.2 and earlier," IETF Internet-draft (work in progress), May 2017, http://www.ietf.org/internet-drafts/draft-ietf-tls-rfc4492bis-17.txt.

[27] E.-J. Goh, D. Boneh, B. Pinkas, and P. Golle, "The design and implementation of protocol-based hidden key recovery," in *Proc. of the 6th International Conference on Information Security (ISC'03), Bristol, U.K.*, ser. Lecture Notes in Computer Science, vol. 2851. Springer-Verlag, October 2003, pp. 165–179.

[28] J. Merrill and D. Johnson, "Covert channels in SSL session negotiation headers," in *Proc. of the 13th International Conference on Security and Management (SAM'15), Las Vegas, USA*. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), July 2015, pp. 70–72.

[29] C. Scott, "Network covert channels: Review of current state and analysis of viability of the use of X.509 certificates for covert communications," Roal Holloway, University of London, Tech. Rep., January 2008.

[30] Z. Gołbiewski, M. Kutyłowski, and F. Zagórski, "Stealing secrets with SSL/TLS and SSH – Kleptographic attacks," in *Proc. of the 5th International Conference on Cryptology and Network Security (CANS'06), Suzhou, China*, ser. Lecture Notes in Computer Science, vol. 4301. Springer-Verlag, December 2006, pp. 191–202.

[31] A. L. Young and M. M. Yung, "Space-efficient kleptography without random oracles," in *Proc. of the 9th International Workshop on Information Hiding (IH'07), Saint Malo, France*, ser. Lecture Notes in Computer Science, vol. 4567. Springer-Verlag, June 2007, pp. 112–129.

[32] O. Sury and R. Edmonds, "Edwards-Curve Digital Security Algorithm (EdDSA) for DNSSEC," IETF RFC 8080, February 2017, http://www.ietf.org/rfc/rfc8080.txt.

[33] Y. Zhang, H. Li, X. Li, and H. Zhu, "Provably secure and subliminal-free variant of schnorr signature," in *Proc. of the 2013 International Conference on Information and Communication Technology (ICT-EurAsia'13), Yogyakarta, Indonesia*, ser. Lecture Notes in Computer Science, vol. 7804. Springer-Verlag, March 2013, pp. 383–391.

[34] M. Naor and O. Reingold, "Number-theoretic constructions of efficient pseudo-random functions," in *Proc. of the 38th Annual Symposium on Foundations of Computer Science (FOCS'97), Miami Beach, Florida*. IEEE, October 1997, pp. 458–467.

[35] C. Wolf and B. Preneel, "Taxonomy of public key schemes based on the problem of Multivariate Quadratic equations," *IACR Cryptology ePrint Archive*, March 2005.

[36] C. Wolf, A. Braeken, and B. Preneel, "Efficient cryptanalysis of RSE(2)PKC and RSSE(2)PKC," in *Proc. of the 4th International Conference on Security in Communication Networks (SCN'04), Amalfi, Italy*, ser. Lecture Notes in Computer Science, vol. 3352. Springer-Verlag, September 2004, pp. 294–309.

[37] J. Ding and D. Schmidt, "Rainbow, a new multivariable polynomial signature scheme," in *Proc. of the 3rd International Conference on Applied Cryptography and Network Security (ACNS'05), New York, USA*, ser. Lecture Notes in Computer Science, vol. 3531. Springer-Verlag, June 2005, pp. 164–175.

[38] J. Patarin, "The oil and vinegar signature scheme," in *Proc. of the Dagstuhl Workshop on Cryptography*, September 1997.

[39] A. Kipnis, J. Patarin, and L. Goubin, "Unbalanced Oil and Vinegar signature schemes," in *Proc. of the 1999 International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT'99), Prague, Czech Republic*, ser. Lecture Notes in Computer Science, vol. 1592. Springer-Verlag, May 1999, pp. 206–222.

[40] H. Imai and T. Matsumoto, "Algebraic methods for constructing asymmetric cryptosystems," in *Proc. of the 3rd International Conference on Algebraic Algorithms and Error-Correcting Codes (AAECC'86), Grenoble, France*, ser. Lecture Notes in Computer Science, vol. 229. Springer-Verlag, July 1986, pp. 108–119.

[41] T. Matsumoto and H. Imai, "Public quadratic polynomial-tuples for efficient signature-verification and message-encryption," in *Proc. of the 1988 Workshop on the Theory and Application of Cryptographic Tech-*

niques (EUROCRYPT'88), Davos, Switzerland, ser. Lecture Notes in Computer Science, vol. 330. Springer-Verlag, May 1988, pp. 419–453.

[42] J. Patarin, "Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two new families of asymmetric algorithms," in *Proc. of the 1996 International Conference on the Theory and Application of Cryptographic Techniques (EUROCRYPT'96), Saragossa, Spain*, ser. Lecture Notes in Computer Science, vol. 1070. Springer-Verlag, May 1996, pp. 33–48.

[43] D. Gligoroski, S. Markovski, and S. J. Knapskog, "A public key block cipher based on Multivariate Quadratic Quasigroups," *IACR Cryptology ePrint Archive*, July 2008.

[44] D. Gligoroski, S. Markovski, and S. J. Knapskog, "Multivariate Quadratic trapdoor functions based on Multivariate Quadratic Quasigroups," in *Proc. of the 1998 American Conference on Applied Mathematics (MATH'08), Cambridge, USA*. WSEAS Press, March 2008, pp. 44–49.

[45] D. Gligoroski, R. S. Ødegård, R. E. Jensen, L. Perret, J.-C. Faugère, S. J. Knapskog, and S. Markovski, "MQQ-SIG: An ultra-fast and provably CMA resistant digital signature scheme," in *Proc. of the 3rd International Conference on Trusted Systems (INTRUST'11), Beijing, China*, ser. Lecture Notes in Computer Science, vol. 7222. Springer-Verlag, November 2012, pp. 184–203.

[46] J. Ding, B.-Y. Yang, C.-M. Cheng, C.-H. O. Chen, and V. Dubois, "Breaking the symmetry: a way to resist the new differential attack," *IACR Cryptology ePrint Archive*, September 2007.

[47] J. Patarin, L. Goubin, and N. Courtois, "C-+* and HM: Variations around two schemes of T. Matsumoto and H. Imai," in *Proc. of the1998 International Conference on the Theory and Application of Cryptology and Information Security(ASIACRYPT'98), Beijing, China*, ser. Lecture Notes in Computer Science, vol. 1514. Springer-Verlag, October 1998, pp. 35–50.

[48] J. Ding and D. Schmidt, "Cryptanalysis of HFEv and internal perturbation of HFE," in *Proc. of the 8th International Workshop on Public Key Cryptography (PKC'05), Les Diablerets, Switzerland*, ser. Lecture Notes in Computer Science, vol. 3386. Springer-Verlag, January 2005, pp. 288–301.

[49] N. Courtois, L. Goubin, and J. Patarin, "Quartz, an asymmetric signature scheme for short signatures on PC – Primitive specification and supporting documentation," October 2001.

[50] A. Petzoldt, M.-S. Chen, B.-Y. Yang, C. Tao, and J. Ding, "Design principles for HFEv- based multivariate signature schemes," in *Proc. of the 21st International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'15), Auckland, New Zealand*, ser. Lecture Notes in Computer Science, vol. 9452. Springer-Verlag, November-December 2015, pp. 311–334.

[51] N. Courtois, L. Goubin, and J. Patarin, "SFLASHv3, a fast asymmetric signature scheme," *IACR Cryptology ePrint Archive*, October 2003.

[52] "Portfolio of recommended cryptographic primitives," NESSIE consortium, February 2003.

[53] J.-C. Faugère, D. Gligoroski, L. Perret, S. Samardjiska, and E. Thomae, "A polynomial-time key-recovery attack on MQQ cryptosystems," in *Proc. of the 18th IACR International Conference on Practice and Theory in Public-Key Cryptography (PKC'15), Gaithersburg, USA*, ser. Lecture Notes in Computer Science, vol. 9020. Springer-Verlag, March-April 2015, pp. 150–174.

[54] M. S. E. Mohamed, J. Ding, J. Buchmann, and F. Werner, "Algebraic attack on the MQQ public key cryptosystem," in *Proc. of the 8th International Conference on Cryptology and Network Security (CANS'09), Kanazawa, Japan*, ser. Lecture Notes in Computer Science, vol. 5888. Springer-Verlag, December 2009, pp. 392–401.

[55] J.-C. Faugère, R. S. Ødegård, L. Perret, and D. Gligoroski, "Analysis of the MQQ public key cryptosystem," in *Proc. of the 9th International Conference on Cryptology and Network Security (CANS'10), Kuala Lumpur, Malaysia*, ser. Lecture Notes in Computer Science, vol. 6467. Springer-Verlag, December 2010, pp. 169–183.

[56] M. Bellare and M. Yung, "Certifying permutations: Noninteractive zero-knowledge based on any trapdoor permutation," *Journal of Cryptology*, vol. 9, no. 3, pp. 149–166, June 1996.

[57] S. A. Kakvi, E. Kiltz, and A. May, "Certifying RSA," in *Proc. of the 18th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT'12), Beijing, China*, ser. Lecture Notes in Computer Science, vol. 7658. Springer-Verlag, December 2012, pp. 404–414.

Author Biography

Alexander Hartl is a student of electrical engineering in the Telecommunications master of TU Wien. He received his B.Sc in electrical engineering and is currently working on his master's thesis in the area of network steganography. His research interests in the area of network security include covert communication techniques, smart grid security and cryptography.

Robert Annessi is working on his doctoral thesis on secure group communication for critical infrastructures. He received his B.Sc and M.Sc degrees in computer engineering from TU Wien in 2011 and 2014 respectively. His research interests span various areas of secure digital communications, such as secure group communication, anonymous communication, covert communication, and subliminal communication

Tanja Zseby is a full professor of communication networks and head of the Institute of Telecommunications at TU Wien . She received her diploma degree (Dipl.-Ing.) in electrical engineering and her doctoral degree (Dr.-Ing.) from TU Berlin, Germany. She worked as scientist and head of the Network Research Group at the Fraunhofer Institute for Open Communication Systems, Berlin, Germany and as a visiting scientist at the University of California, San Diego (UCSD).