Security Analysis of a Multi-Factor Authenticated Key Exchange Protocol

Feng Hao, Dylan Clarke

School of Computing Science
Newcastle University
{feng.hao, dylan.clarke}@ncl.ac.uk

Abstract. This paper shows several security weaknesses of a Multi-Factor Authenticated Key Exchange (MK-AKE) protocol, proposed by Pointcheval and Zimmer at ACNS'08. The Pointcheval-Zimmer scheme was designed to combine three authentication factors in one system, including a password, a secure token (that stores a private key) and biometrics. In a formal model, Pointcheval and Zimmer formally proved that an attacker had to break all three factors to win. However, the formal model only considers the threat that an attacker may impersonate the client; it however does not discuss what will happen if the attacker impersonates the server. We fill the gap by analyzing the case of the server impersonation, which is a realistic threat in practice. We assume that an attacker has already compromised the password, and we then present two further attacks: in the first attack, an attacker is able to steal a fresh biometric sample from the victim without being noticed; in the second attack, he can discover the victim's private key based on the Chinese Remainder theorem. Both attacks have been experimentally verified. In summary, an attacker actually only needs to compromise a single password factor in order to break the entire system. We also discuss the deficiencies in the Pointcheval-Zimmer formal model and countermeasures to our attacks.

1 Introduction

Authenticated Key Exchange (AKE) is a fundamental security protocol for almost all secure communication systems. Depending on how the "authentication" is defined, AKE schemes are generally divided into two categories: Password-based Authenticated Key Exchange (PAKE) and PKI-based Authenticated Key Exchange [7]. In the former case, the authentication is based on the knowledge of a shared password, without requiring any Public Key Infrastructure (PKI). In the latter case, each party possesses a unique pair of the public and private keys. The authentication is based on the possession of the private key, which is usually stored in a tamper resistant device. A PKI is needed to securely distribute authentic public keys to all users [18].

While the above two AKE categories correspond to something you know (i.e., a password) and something you have (i.e., a secure token that stores a private key) respectively, there is a third authentication factor: namely, something you

are (i.e., biometrics). Biometrics are an advanced authentication mechanism, which works by measuring a person's unique behavioral or physical characteristics [2]. The security of biometrics largely depends on whether a trusted path exists, which ensures the biometric sample is freshly obtained from the live subject. Such a trusted path can be realized, for example by enforcing supervision in a controlled environment (e.g., airport). In an unsupervised environment, the security will have to depend on the liveness detection features embedded with the biometric scanning equipment [18]. Assuming a trusted path already in place, researchers have made progress in designing biometrics-based AKE schemes [3, 5, 6].

So far all the above-mentioned AKE schemes are based on a single factor. In recent years, Multi-Factor Authenticated Key Exchange (MF-AKE) has emerged as an active research topic [10–14, 18–23]. The rationale is to improve single factor based AKE by combining two or even more factors in one system. This is a worthy goal, but extra caution should be taken. The past thirty years of research in the area of authenticated key exchange has proved that it is incredibly difficult to get even a single factor based AKE scheme right [4]. Designing a multi-factor AKE protocol can only be harder.

Many MF-AKE protocols have been proposed – and subsequently broken. For example, in 2010, Lee et al. proposed a two-factor AKE protocol that combines a smartcard and a password [11]. But a year later, their protocol was found insecure: the compromise of the smartcard factor breaks the entire scheme [21]. Xu et al. proposed a similar two-factor AKE protocol based on a smartcard and a password with "formal security proofs" in [23]. Within a year, their protocol was broken and a patched protocol was proposed in [19]. Yet the patched protocol was shortly found insecure [20]. Li and Hwang proposed a different type of two-factor AKE protocol, which consists of biometrics and a smart card [13]. In less than a year, their scheme was broken [14]. These examples show that MK-AKE is still a young field; more research is very much needed.

Recently at ACNS'08, Pointcheval and Zimmer proposed the first Multi-Factor Authenticated Key Exchange protocol that combines all three factors in one system: a password, a smartcard and biometrics [18]. Furthermore, the authors defined a formal model and formally proved that an adversary had to break all three factors in order to win. Unfortunately, we find their protocol vulnerable, as we explain below.

2 Pointcheval-Zimmer protocol

In this section, we will describe Pointcheval-Zimmer's Multi-Factor Authenticated Key Exchange scheme. We will follow the original notations in [18] as closely as possible.

2.1 Notation

The client $\mathcal{C}$ owns a tuple $t_{\mathcal{C}} = (W'_{\mathcal{C}}, \mathrm{sk}_{\mathcal{C}} = \mathrm{x}_{\mathcal{C}}, \mathrm{pwd}_{\mathcal{C}})$ where $W'_{\mathcal{C}}$ is a biometric, $\mathrm{sk}_{\mathcal{C}}$ a private key and $\mathrm{pwd}_{\mathcal{C}}$ a password. The iris code is used in [18] as a specific example for biometrics.

The server $\mathcal{S}$ holds a list of tuples for each client $t_{\mathcal{S}} =< t_{\mathcal{S}}[\mathcal{C}] >$, where $t_{\mathcal{S}}[\mathcal{C}]$ is a transformed-tuple of $t_{\mathcal{C}}$. More specifically, $t_{\mathcal{S}}[\mathcal{C}]$ contains the following information about the client $\mathcal{C}$:

- The client's public key $h = g^{x_{\mathcal{C}}}$.
- An encrypted copy of the iris-code template that was enrolled during registration. The template is denoted as $W_{\mathcal{C}} = (W_i)_{i \leq N}$, where W_i is the i-th bit of $W_{\mathcal{C}}$ and N is the number of bits of an iris code. The ciphertext is obtained by using the El Gamal encryption algorithm; the result is $(g^{r_i}, h^{r_i} \cdot g^{W_i})_i$, where r_i is a random element in $\mathbb{Z}_q$.
- The client's password $\mathrm{pwd}_{\mathcal{C}}$.

2.2 Description of protocol

The protocol is based on a cyclic group with parameters (p, g, q). The p and q are big prime numbers, and $q \,|\, p - 1$. Let $\mathbb{G}_q$ be a subgroup in $\mathbb{Z}_p^*$ with prime order q, and g be its generator. In addition, the protocol defines two random elements in $\mathbb{G}_q$, namely u and v. Figure 1 specifies how the protocol works. The symbols used in the figure should be self-explanatory[1].

In [18], the authors also suggest practical parameters for a real-world implementation. They assume an iris scan has $N = 1024$ bits. A value of $t = 300$ is defined as the threshold, so two iris codes with less than t disparate bits (i.e., Hamming distance) are considered belonging to the same eye; otherwise, they are regarded from different eyes. Furthermore, the authors use l to denote the bit lengths of $\alpha, \alpha', \beta, \beta'$ (see line 5 and 7 in Figure 1). That is: $l = ||\alpha'|| = ||\beta'|| = ||\alpha|| = ||\beta||$.

The value of l is critical to the correctness of the protocol. For small values of l, say $l = 1$, then the protocol will be guaranteed to fail even between two honest players. To ensure a successful honest execution of the protocol, the value l must not be small. Pointcheval and Zimmer recommend $l = 24$, and they estimate that with this parameter, the probability for a successful execution between two honest players is $1 - 2 \cdot t \cdot 2^{-24} = 1 - 2^{-14}$.

Pointcheval and Zimmer also define a formal model to prove the security of the protocol [18]. The model assumes the adversary is able to corrupt a client $\mathcal{C}$ in the following ways: by compromising the password $\mathrm{pwd}_{\mathcal{C}}$, by stealing the private key $\mathrm{sk}_{\mathcal{C}}$, or by spoofing biometrics $W_{\mathcal{C}}$. Under this model, the authors formally prove that an attacker has to compromise all three factors in order

[1] The original specification in in [18] does not explicitly explain the meaning of lsb in Line 12 and 14. We assume lsb refers to "least significant bits" and we interpret it as a key derivation function that derives a session key from raw keying materials. This ambiguity does not affect our security analysis however.

	Client		Server
	$\mathcal{C}$: $(W_c^{'} = (W_i^{'})_i,\ sk_{\mathcal{C}} = x_{\mathcal{C}},\ \mathrm{pwd}_{\mathcal{C}})$		$\mathcal{S}$: $((g^{r_i}, h^{r_i} g^{W_i})_i,\ h = g^{x_{\mathcal{C}}},\ \mathrm{pwd}_{\mathcal{C}},\ \mathcal{C})_{\mathcal{C}}$
1	$b \xleftarrow{\$} \mathbb{Z}_q$ and $B = g^b$, $B^* = B \cdot v^{\mathrm{pwd}_{\mathcal{C}}}$	$\xrightarrow{\mathcal{C},\ B^*}$	For $1 \le i \le N$
2			$r_i' \xleftarrow{\$} \mathbb{Z}_q$ and compute
3			$g^{s_i} = g^{r_i'} \cdot g^{r_i}$, $h^{s_i} g^{W_i} = h^{r_i'} \cdot h^{r_i} \cdot g^{W_i}$
4	For $1 \le i \le N$	$\xleftarrow{\mathcal{S},\ (g^{s_i})_i,\ A^*}$	$a \xleftarrow{\$} \mathbb{Z}_q$, $A = g^a$, $A^* = A \cdot u^{\mathrm{pwd}_{\mathcal{C}}}$
5	compute $H(K_i') = \alpha_i' \| \beta_i' \| k_i'$ with:		
6	$K_{\mathcal{C}} = (\frac{A^*}{u^{\mathrm{pwd}_{\mathcal{C}}}})^b$, $K_{\mathcal{C}}^i = (g^{s_i})^{x_{\mathcal{C}}} \cdot g^{W_i'}$		
7	$K_i' = \mathcal{S} \| \mathcal{C} \| (g^{s_i})_i \| A^* \| B^* \| K_{\mathcal{C}}^i \| K_{\mathcal{C}} \| \mathrm{pwd}_{\mathcal{C}} \| i$	$\xrightarrow{(\alpha_i')_i}$	For $1 \le i \le N$, $H(K_i) = \alpha_i \| \beta_i \| k_i$ with:
8			$K_{\mathcal{S}} = (\frac{B^*}{v^{\mathrm{pwd}_{\mathcal{C}}}})^a$, $K_{\mathcal{S}}^i = h^{s_i} \cdot g^{W_i}$
9			$K_i = \mathcal{S} \| \mathcal{C} \| (g^{s_i})_i \| A^* \| B^* \| K_{\mathcal{S}}^i \| K_{\mathcal{S}} \| \mathrm{pwd}_{\mathcal{C}} \| i$
10			
11			If $\#\{i : \alpha_i \neq \alpha_i'\} \le t$
12			Then acc = 1, $K = \mathrm{lsb}_{\mathrm{k}}(\|_{\mathrm{i}:\alpha_{\mathrm{i}} = \alpha_{\mathrm{i}}'} \mathrm{k_i})$
13	If $\#\{i : \beta_i \neq \beta_i'\} \le t$	$\xleftarrow{(\beta_i)_i}$	Else acc = 0, $K \xleftarrow{\$} \{0,1\}^k$, $\beta_i \xleftarrow{\$} \{0,1\}^l$
14	Then acc = 1, $K' = \mathrm{lsb}_{\mathrm{k}}(\|_{\mathrm{i}:\beta_{\mathrm{i}} = \beta_{\mathrm{i}}'} \mathrm{k_i'})$		
15	Else acc = 0, $K' \xleftarrow{\$} \{0,1\}^k$		

Fig. 1. Pointcheval-Zimmer protocol

to win. However, the formal model only considers the unilateral authentication from the client to the server. In other words, it implicitly assumes the server is honest. This is acknowledged in [18] as "a strong limitation", but the authors do not explicitly explain what will happen if the attacker is able to impersonate the server to the client (e.g., in a man-in-the-middle attack). We fill the gap by analyzing this threat in the following section.

3 Attacks

When the attacker is able to impersonate the server (which is a realistic threat in practice), we show the Pointcheval-Zimmer protocol is insecure. First of all, we assume that the attacker has compromised the client's password (say by phishing). Then, we show how the attacker is able to subsequently compromise the other two factors: the biometrics and private key, hence breaking the entire system.

3.1 Stealing biometrics

First, we show how the compromise of the password factor will lead to the breach of the biometrics factor. Figure 2 shows how the attack works. As shown in Line 2, the attacker selects random values for s_i and computes g^{s_i} accordingly. This is a clear deviation from the original specification, because if the server had honestly followed the specification in computing $g^{s_i} = g^{r_i'} \cdot g^{r_i}$, it will not have knowledge of the exponent of g^{s_i}. But in this case, the server (attacker) knows

	Client (Victim)		Server (Attacker)
	$\mathcal{C}$: $(W'_c = (W'_i)_i,\ sk_\mathcal{C} = x_\mathcal{C},\ \mathrm{pwd}_\mathcal{C})$		$\mathcal{M}$: $(h = g^{x_\mathcal{C}},\ \mathrm{pwd}_\mathcal{C})$
1	$b \xleftarrow{\$} \mathbb{Z}_q$ and $B = g^b$, $B^* = B \cdot v^{\mathrm{pwd}_\mathcal{C}}$	$\mathcal{C},\ B^* \longrightarrow$	For $1 \le i \le N$
2			$s_i \xleftarrow{\$} \mathbb{Z}_q$ and compute g^{s_i}
3	For $1 \le i \le N$	$\mathcal{S},\ (g^{s_i})_i,\ A^* \longleftarrow$	$a \xleftarrow{\$} \mathbb{Z}_q$, $A = g^a$, $A^* = A \cdot u^{\mathrm{pwd}_\mathcal{C}}$
4	compute $H(K'_i) = \alpha'_i \| \beta'_i \| k'_i$ with:		
5	$K_\mathcal{C} = (\frac{A^*}{u^{\mathrm{pwd}_\mathcal{C}}})^b$, $K^i_\mathcal{C} = (g^{s_i})^{x_\mathcal{C}} \cdot g^{W'_i}$		
6	$K'_i = \mathcal{S} \| \mathcal{C} \| (g^{s_i})_i \| A^* \| B^* \| K^i_\mathcal{C} \| K_\mathcal{C} \| \mathrm{pwd}_\mathcal{C} \| \mathrm{i}$	$(\alpha'_i)_i \longrightarrow$	
7		$(\beta_i)_i \longleftarrow$	$((\beta_i)_i,\ (k_i)_i,\ (\mathrm{W''_i})_\mathrm{i}) \leftarrow \mathrm{SearchBiometrics}((\alpha')_i)$
8	If $\#\{i : \beta_i \neq \beta'_i\} \le t$		Compute $K = \mathrm{lsb}_k(\|_i k_i)$
9	Then acc = 1, $K' = \mathrm{lsb}_\mathrm{k}(\|_{\mathrm{i}:\beta_\mathrm{i}=\beta'_\mathrm{i}} \mathrm{k'_i})$		
10	Else acc = 0, $K' \xleftarrow{\$} \{0,1\}^k$		

Fig. 2. Attack 1: stealing fresh biometrics from the client without being detected

the exponent (which allows carrying out the subsequent attack). This deviation is undetectable to the client.

One principle in designing robust security protocols is "never let yourself be used as an oracle by your opponent" [1]. Unfortunately, in this case, the client has made itself an oracle to the attacker. After receiving the data from the attacker, the client proceeds to compute values of $(\alpha'_i)_i$ and sends them over to the server. Those values will allow the attacker to discover the biometric sample from the user, as described in Algorithm 1.

The recovered sample from Algorithm 1 is high-quality biometric data. By "high-quality", we mean the recovered sample $(W''_i)_i$ is extremely close to the client's sample $(W'_i)_i$, which is freshly acquired in a favorable supervised condition. The exact difference between $(W''_i)_i$ and $(W'_i)_i$ depends on the parameter l (which is the bit length of $\alpha, \alpha', \beta, \beta'$). If we take $l = 24$ as recommended in [18], the probability of each bit in W''_i being correct (i.e., it equals the bit in W_i) is $p = 1 - 2^{24}$. Hence, the probability of all $N = 1024$ bits in W''_i being correct is $p^N = 99.994\%$. With an almost identical biometric sample, it is trivial for the attacker to compute $(\beta_i)_i$ so that he can successfully finish the rest of the protocol (see Algorithm 1). In reality, having two identical biometric samples normally suggests a replay attack, so the attacker may randomly corrupt some (up to t) of the β_i values to artificially make the biometric matching look "fuzzy". The same attack also applies to the three-party extension of Pointcheval-Zimmer's protocol, which was proposed in [15].

3.2 Disclosing private key

In the second attack, we show how the attacker is able to recover the client's private key, based on a compromised password and stolen biometrics (obtained from the first attack).

Algorithm 1 SearchBiometrics algorithm

Input: $(\alpha'_i)_i$
Output: $(\beta_i)_i$, $(k_i)_i$, $(W''_i)_i$

1: **for** $i = 1, \ldots, N$ **do**
2: *Compute* $H(K'_i) = \alpha'_i||\beta'_i||k'_i$ with
3: $K_{\mathcal{C}} = (\frac{B^*}{v^{\text{pwd}_{\mathcal{C}}}})^a$, $K^i_{\mathcal{C}} = (g^{x_{\mathcal{C}}})^{s_i}$
4: $K'_i = \mathcal{S}||\mathcal{C}||(g^{s_i})_i||A^*||B^*||K^i_{\mathcal{C}}||K_{\mathcal{C}}||\text{pwd}_{\mathcal{C}}||i$
5: **if** $\alpha_i = \alpha'_i$ **then**
6: $(W''_i)_i = 0$
7: $(\beta_i)_i = \beta'_i$
8: $(k_i)_i = k'_i$
9: **else**
10: $(W''_i)_i = 1$
11: *Recompute* $H(K'_i) = \alpha'_i||\beta'_i||k'_i$ with
12: $K'_i = \mathcal{S}||\mathcal{C}||(g^{s_i})_i||A^*||B^*||K^i_{\mathcal{C}} \cdot g||K_{\mathcal{C}}||\text{pwd}_{\mathcal{C}}||i$
13: $(\beta_i)_i = \beta'_i$
14: $(k_i)_i = k'_i$
15: **end if**
16: **end for**

The attack is possible because the client is not required to perform public key validation on the data received from the server[2] (Line 4-7 in Figure 1). Many protocols omit the step of public key validation in order to increase the protocol efficiency. But this is often done at the expense of security. One well-known example is HMQV, which "provably" drops public key validation based on a formal model and security proofs [9]. However, Menenzes *et al.* subsequently pointed out the flaws in the formal model and attacks on HMQV [17]. Their paper highlights the importance of performing the public key validation. In [18], Pointcheval and Zimmer defined a formal model and provided formal security proofs for their MK-AKE protocol. However, the formal model in [18] implicitly assumes the server is *honest*, which is not a valid assumption.

The attack works as follows. Instead of computing g^{s_i} as in the original protocol (Line 3 in Figure 1), the attacker selects small subgroup elements in $\mathbb{Z}^*_p$ and sends them to the client. We use $\mathbb{G}_s$ to denote a small subgroup of prime order s. Let us take $i = 1$ as an example. Let b_1 be a generator of $\mathbb{G}_s$ (i.e., an arbitrary non-identity element). After receiving b_1, the client proceeds to compute α'_1 as specified in the original protocol and sends it to the server. We know $W_1 = W'_1$ with a high probability (say 90%). For simplicity of illustration, we first assume $W_1 = W'_1$. The attacker knows all the concatenated items in K'_i (Line 7 in Figure 1), except $K^i_{\mathcal{C}}$. Based on $W_1 = W'_1$ and the fact that $b_1^{x_{\mathcal{C}}} \bmod p$ falls within a small range, the attacker can easily obtain the value of $a_1 = b_1^{x_{\mathcal{C}}} \bmod p$ by exhaustive search (i.e., against the value of α'). In the

[2] In the three-party extension of the Pointcheval-Zimmer protocol [15], there is no public key validation either. So the same attack applies.

subsequent step, the attacker can compute $x_{\mathcal{C}} \bmod s$ – once again by exhaustive search (based on a_1, b_1 and p). Since b_1 is a generator in $\mathbb{G}_s$, $b_1^{x_{\mathcal{C}}}$ also falls within the same small subgroup. Through exhaustive search, the attacker can obtain $x_{\mathcal{C}} \bmod s$. By repeating the same procedure for different subgroups, the attacker can recover more secret bits of the private key. Depending on the group setting, it is possible to recover a full copy of the private key based on the Chinese Remainder theorem (e.g., see a concrete implementation of the attack in Appendix A).

In the above analysis, we have assumed $W_1 = W_1'$, but in reality the equality only holds for a probability say 90%. This problem can be easily addressed by exploiting the large amount of redundancies in an iris code. Instead of sending one b_i value, the attacker could send several b_i values from the same small subgroup $\mathbb{G}_s$. For example, with 30 b_i values, there will be on average $30 \times 0.9 = 27$ results that give the same $x_{\mathcal{C}} \bmod s$. This removes any uncertainty due to the fuzzy nature of biometrics.

3.3 Combining two attacks

It is easy to combine the two attacks together. For that, we need to modify the first attack slightly: after successfully stealing a biometric sample, the adversary sends to the client random β_i' values to trigger a "rejection". Because the β_i' values are random, the matching at the client will fail. However, the failure will hardly raise any suspicion from the client, as any biometric system has a non-zero false rejection rate in the real-world operation[3]. Most likely, the user will be prompted to try again (and be ready to be more cooperative with the iris photo-taking). In the second attempt, the attacker will send small subgroup elements to discover the user's private key. Subsequently, the attacker will send correct β' values to trick the client into believing that the second attempt is successful.

Thus, with a stolen password, the attacker has successfully compromised the user's two other factors, hence breaking the entire system.

4 Discussion

Between the two attacks, the second attack might look more damaging, but it is easier to fix. We can address it by adding public key validation in the protocol. However, this will significantly decrease the computational efficiency of the original protocol, as it normally takes a full exponentiation to verify if the received public key is a valid element in the correct subgroup of $\mathbb{Z}_p^*$. Nonetheless, our attack shows that this step is necessary (more explanation about the importance of public key validation can be found in [17]).

[3] Even with two perfectly matching biometric samples, a false reject may still occur in the Pointcheval-Zimmer scheme (due to the deficiency in the engineering design) with a probability 2^{-14} for $l = 20$ (see [18]).

The first attack indicates a more fundamental flaw with the protocol. The question concerns the exact role of biometric authentication in the Pointcheval-Zimmer protocol. Assume the attacker has compromised the password and the private key, but he does not have user's biometric data. What kind of security assurance can the remaining biometrics factor provide? Very little – based on the following observations. First, the client cannot safeguard the biometrics. As shown in Section 3.1, the protocol allows a remote attacker to steal the user's freshly obtained biometric sample without the user's awareness at all. Second, the encryption of biometrics at the server does not really preserve the privacy of biometrics as claimed in [18]. The attacker will target the weaker part in the system – the client – to steal a biometric sample in the plaintext. Third, although by design biometric matching is done at the server, the result of the matching cannot be securely sent back to the client. As demonstrated in Section 3.1, the attacker is able to arbitrarily manipulate the matching outcome. All these suggest a fundamental problem with the protocol, which seems not easily fixable with the current structural design of the protocol. More research on this is much needed (e.g., some techniques in biometrics-based AKE [3, 5, 6] may be applicable to address this issue).

The above flaw was missed by the formal analysis in [18], because the authors assume biometrics as "fully public". Their justification is that the "the opposite assumption is not reasonable in practice", which refers to the treatment of biometrics as "fully secret" in [3,5,6]. However, we think that "biometric applications lie between the extremes of secret data and fully public data" [8]. Indeed, it is possible to steal users' fingerprints that were left on the keyboard, or surreptitiously photograph their irises using hidden cameras. But, samples stolen this way tend to be of poor-quality [2]. Getting high-quality biometric samples usually requires user cooperation and a favorable environment: e.g., proper posture, distance to the camera and lighting etc. Stealing such samples without being detected by the user is not easy. When designing biometrics-based protocols, it is prudent not to rely on the secrecy of biometrics, but on the other hand, it is obviously a security flaw to give away freshly acquired biometric samples to a remote attacker.

5 Conclusion

In this paper, we described two attacks on Pointcheval-Zimmer's Multi-Factor Authenticated Key Exchange (MF-AKE) protocol. In the first attack, we showed how an attacker could make use of a stolen password to subsequently compromise the biometrics factor without being detected. This violates the privacy of biometric data. In the second attack, we showed how an attacker could further discover the user's private key by exploiting the small subgroup confinements. This attack breaches the presumed tamper-resistance of a secure token that stores the private key. In summary, the attacker only needs to compromise a single password factor in order to compromise Pointcheval-Zimmer's three-factor AKE protocol.

Acknowledgement

We sincerely thank Pointcheval and Zimmer for quickly and frankly acknowledging our attacks and the deficiencies in their formal model, and also for confirming our suggested countermeasures.

Appendix A: An implementation of small subgroup confinement attack on Pointcheval-Zimmer's protocol

We provide implementation details about the second attack below. (We have also experimentally verified the first attack, but since that implementation is straightforward, we do not include the details in the paper.)

Group parameters

Let us assume a cyclic group $\mathbb{Z}_p^*$ that has a subgroup of prime order q. Hence, $q \,|\, p-1$. Let g be a generator, $g^q \bmod p = 1$. As a specific example, we define a 512-bit p, 160-bit q, 512-bit g with the following values (in the hex format).

$$\begin{aligned} p &= f95b8b2f45b3016efb6ec51d342931aea4a5f4516d15c4ed2cf79e4d318\ldots \\ &\quad e28837989bedcbe4ce8693f68de6b72b1f74c8e109bc9155f5d2d65e9f6d\ldots \\ &\quad 091e7f79b \\ q &= e80f99e4981ee1eac37d8f0bf707b2067f6fe8cf \\ g &= 33c65b25ad4c47ac067083b7f2acf53ed3a053dbe508acbabe179029dad\ldots \\ &\quad 77a04c0953c1dbce02ce2f8cf5b030a36de7868b7434194816dbe7da920\ldots \\ &\quad 13bc4696d \end{aligned}$$

Note that this (artificial) example is for illustration only. In practice, the bit lengths of p and q are normally much longer. In addition, we assume $\mathbb{Z}_p^*$ has several small subgroups of primer order s_i, so $s_i \,|\, p-1$. The hex values of s_i are given in Table 1. Except s_1, all other s_i are 21-bit long.

i	1	2	3	4	5	6	7	8	9
s_i	2	15a661	1182bb	12b357	1fa9e7	1f1c9f	1c58b7	16b6b3	1727c3

Table 1. Orders of small subgroups of $\mathbb{Z}_p^*$

Small subgroup confinement attack

To demonstrate how an adversary can recover the private key, we first define a random 160-bit private key in the range of $[0,\, q-1]$:

$$x = 538b2f452c20f9cd7e356455e2ae66e9924ddd5d$$

By exploiting the small subgroup confinement, the adversary can obtain $a_i = b_i^x \bmod p$ for each of the small subgroups with prime orders s_i. Furthermore, he can obtain $c_i = x \bmod s_i$ through exhaustive search. Table 2 shows the results of a_i and c_i for each of the small subgroups $\mathbb{G}_{s_i}$. On average, it takes about 68.5 seconds to find c_i using exhaustive search, on a 2.93 GHz desktop PC with 4 GB memory.

i	$a_i = b_i^x \bmod p$	$c_i = x_i \bmod p$	Time for exh search (ms)
1	f95b8b2f45b3016efb6ec51d342931aea4a5f4516d15c4ed2cf79e4d318e2883 7989bedcbe4ce8693f68de6b72b1f74c8e109bc9155f5d2d65e9f6d091e7f79a	1	0
2	791778359b242e573617fff6735703be986dd9a6271be8b413381f4211ffbc0cb 6cb2da17c880a115e223752cb431708c4a64d68cbd8109c7a2e31e434839682	122156	124854
3	133023e42630ff22b9f9fb3e6a1ceeddf2d7fc1014fcab33eeedc1af416951c2c9 099d8fef275408d1f82c7090cd72744a260685381a15cc8e58bdd052bb1928	517f1	31481
4	93358cdb4153463b93f0525d6819426b7b4c0fcb5905cf10db5f39eeea68a879 c8ad548cef72476671ff2805e1cbf8644f39f2d4fc71570522e4e89784a0a0aa	42cb2	25271
5	f517a5a96bd4a92af483727dd0c73c251ac159056ec5b2eea90854379ba8344 cc41e641d9e84ec319fd4ab545e038cba799972a2db93b2bf315fb62b08402cb	e0125	93050
6	e1895eb12f66645c2359ad6185d85fb02f39ddd3b2e80392c3f53ffc5ebcb23a9 81eb1d9cfd7a8eeea8a13e83af81a726280fcd7d545450fb6871786f9e2ebe6	15d34	7544
7	8dc566a29c84977ceb1e1a466321859fe3022f7ab3adae44ead9d8b7c2dc5461 b1ea9441b19425c13da5b7c998ea7fbe41aaea70177118b37438c5f36cad85d2	121427	121793
8	d73e16cf041ecd7c9c73d6414ad01b0ce85deaedbcf6834591f373091a51903 1cbb4aebe31fd56afab5750226584a762eaf7ffef0f5d7e2f940e27ea8d8b0d41	ef2c6	99664
9	dbdb75657da4a1e4312f17b7519eff2fe2c0b0b2fbe225482e3f78f73530c545a 3c8cd4d6fa0abd3f27058090cd1992263e72f16e47b7256f916d4a20201e5c0	10c700	112565

Table 2. Exhaustive search among the small subgroup confinements

With the c_i values, the private key can be recovered based on the Chinese Remainder theorem [16]. For example, we could apply Gauss's algorithm to solve the simultaneous congruences problem as follows. Let $n = s_1 \cdot s_2 \cdot s_3 \cdot s_4 \cdot s_5 \cdot s_6 \cdot s_7 \cdot s_8 \cdot s_9$. Then $x = \sum_i c_i N_i M_i \bmod n$ where $N_i = n/s_i$ and $M_i = N_i^{-1} \bmod s_i$. In this specific example, n (169-bit) $> q$ (160-bit), so we are able to recover the full private key. It takes merely 1 millisecond to obtain the following result using Gauss's algorithm.

$$\begin{aligned} x' &= 538b2f452c20f9cd7e356455e2ae66e9924ddd5d \bmod n \\ &= 538b2f452c20f9cd7e356455e2ae66e9924ddd5d \bmod q \end{aligned}$$

References

1. R.J. Anderson, R. Needham, "Robustness Principles for Public Key Protocols," Proceedings of the 15th Annual International Cryptology Conference on Advances

in Cryptology, LNCS 963, pp. 236–247, 1995.
2. R.J. Anderson, *Security Engineering: A Guide to Building Dependable Distributed Systems*, 2nd edition, Wiley, 2008.
3. X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky, and A. Smith, "Secure Remote Authentication Using Biometric Data," Proceedings of Eurocrypt'05, LNCS 3494, pp. 147-163, 2005. J.K. Lee and S.R. Ryu, "Fingerprint-based Remote User Authentication Scheme Using Smart Cards". Electronics Leters, 38(12):554–555, June 2005.
4. C. Boyd, A. Mathuria, *Protocols for Authentication and Key Establishment*, Springer-Verlag, 2003.
5. X. Boyen, "Reusable Cryptographic Fuzzy Extractors," ACM CCS'04, pp. 82-91, 2004.
6. Y. Dodis, J. Katz, L. Reyzin and A. Smith, "Robust Fuzzy Extractors and Authenticated Key Agreement from Close Secrets," Eurocrypt'04, Vol. 3027, pp. 523-540, 2004.
7. F. Hao, "On Robust Key Agreement Based on Public Key Authentication," proceedings of the 14th International Conference on Financial Cryptography and Data Security, Tenerife, Spain, LNCS 6052, pp. 383-390, 2010.
8. F. Hao, R. Anderson, J. Daugman, "Combining crypto with biometrics effectively," IEEE Transactions on Computers, Vol. 55, No. 9, pp. 1081-1088, 2006.
9. H. Krawczyk, "HMQV: A High-Performance Secure Diffie-Hellman Protocol," CRYPTO 2005. LNCS, Vol. 3621, pp. 546-566, 2005.
10. M. Hwang, S. Chong, T. Chen, "DoS-Resistant ID-Based Password Authentication Scheme Using Smart Cards," *Computer Journal of Systems and Software*, Vol. 7, No. 50, pp. 147-150, 2009.
11. Y. Lee, S. Kim, D. Won, "Enhancement of Two-Factor Authenticated Key Exchange Protocols in Public Wireless LANs," *Computers and Electrical Engineering*, Vol. 36, No. 1, pp. 213-223, 2010.
12. I.E Lian, C.C. Lee, M.S. Hwang, "A Password Authentication Scheme Over Insecure Networks," Journal of Computer System Sciences, Vo. 72, pp. 727-740, 2006.
13. C.T. Li, M.S. Hwang, "An Efficient Biometrics-Based Remote User Authentication Scheme Using Smart Cards," Journal of Network and Computer Applications, Vol. 33, No. 1, pp. 1-5, 2010.
14. X. Li, J.W. Niu, J. Ma, W.D. Wang, "Cryptanalysis and Improvement of a Biometrics-Based Remote User Authentication Scheme Using Smart Cards," Journal of Network and Computer Applications, Vol. 34, No. 1, pp. 73-79, 2011.
15. Y. Liu, F. Wei, and C. Ma, "Multi-Factor Authenticated Key Exchange Protocol in the Three-Party Setting," Inscrypt 2010, LNCS 6584, pp. 255-267, 2011.
16. A.J. Menezes, P.C. van Oorschot and S.A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.
17. A. Menezes, B. Ustaoglu, "On The Importance Of Public-Key Validation In The MQV And HMQV Key Agreement Protocols," INDOCRYPT 2006, LNCS, Vol. 4329, pp. 133-147, 2006.
18. D. Pointcheval and S. Zimmer, "Multi-Factor Authenticated Key Exchange," Proceedings of Applied Cryptography and Network Security (ACNS'08), pp. 277-295, LNCS 5037, 2008.
19. R. Song, "Advanced Smart Card Based Password Authentication Protocol," Computer Standards & Interfaces, Vol. 32, pp. 321-325, 2010.
20. J.E. Tapiador, J.C. Hernandez-Castro, P. Peris-Lopez, J.A. Clark, "Cryptanalysis of Song's Advanced Smart Card Based Password Authentication Protocol," Technical report available at `http://arxiv.org/pdf/1111.2744`, 2011.

21. S. Wu, Y. Zhu, "Improved Two-Factor Authenticated Key Exchange Protocol," The International Arab Journal of Information Technology, Vol. 8, No. 4, pp. 430-439, 2011.
22. T. Xiang, K. Wong, X. Liao, "Cryptanalysis of A Password Authentication Scheme Over Insecure Networks," Journal of Computer System Sciences, Vol. 74, pp. 657-661, 2008.
23. J. Xu, W.T. Zhu, D.G. Feng, "An Improved Smart Card Based Password Authentication Scheme with Provable Security," Computer Standards & Interfaces, Vol. 31, pp. 723-728, 2009.