A Chosen Random Value Attack on WPA3 SAE Authentication Protocol

SHENG SUN, Huawei Technologies Canada, Canada

Simultaneous Authentication of Equals (SAE) is a password-authenticated key exchange protocol that is designed to replace the WPA2-PSK-based authentication. The SAE authenticated key exchange protocol supports the peer-to-peer authentication and is one of the major authentication mechanisms of the Authentication and Key Management Suite specified within Wi-Fi. The SAE authenticated key exchange protocol has been widely implemented in today's Wi-Fi devices as part of major security feature upgrades and is regarded as the third generation of Wi-Fi Protected Access. This article presents a way of attacking the weaker randomness generation algorithm within the SAE protocols, which can lead to successful impersonation types of attacks. We also suggest some protocol amendments for protection. It is recommended that SAE implementations should be upgraded to ensure protection against these attacks.

CCS Concepts: • **Security and privacy** → **Software and application security;**

Additional Key Words and Phrases: Wi-Fi, WPA3, authenticated key exchange, impersonation attacks

ACM Reference format:
Sheng Sun. 2022. A Chosen Random Value Attack on WPA3 SAE Authentication Protocol. *Digit. Threat.: Res. Pract.* 3, 2, Article 16 (February 2022), 8 pages.
https://doi.org/10.1145/3468526

1 INTRODUCTION

Simultaneous Authentication of Equals (SAE) [6] is a password-authenticated key exchange protocol [12] specified for use within IEEE 802.11 systems. In this article, we simply call it the SAE authentication protocol. The SAE authentication protocol is based on the Dragonfly [7, 8] protocol and employs discrete logarithm cryptography to perform an efficient key exchange. It performs mutual authentication using a password or a **pre-shared key (PSK)** between two authenticating parties. The SAE protocol was originally designed to support the P2P authentication and to replace the WPA2-PSK-based authentication. The SAE authentication protocol supports both the Finite Field Computation and Elliptical Curve Computation. These give the SAE authentication protocol the benefits of reusing the safe Diffie-Hellman Modular Exponential groups [10] specified in Internet Key Exchange protocol [9] or the safe elliptical curves, such as NIST P-256 specified in Reference [1]. The SAE authentication protocol also possesses the security property of Perfect Forward Secrecy [5], which is useful for scenarios such as IoT, Hotspot, and so on.

The SAE protocol has a number of components. One of these is the Dragonfly protocol, which is designated within the Internet Engineering Task Force as a candidate standard for IPsec, Transport Layer Security, and Extensible Authentication Protocol applications [13]. The SAE protocol is resistant to active attacks or online

Author's address: S. Sun, Huawei Technologies Canada, Ottawa, Ontario, Canada, K2K 3J1; email: rob.sun@huawei.com.

2576-5337/2022/02-ART16 $15.00
https://doi.org/10.1145/3468526

dictionary attacks, passive attacks or off-line dictionary attacks. The resistance to online dictionary attacks assures that an adversary cannot gain any advantage even by launching an active attack within key exchange procedure. Another technique with the SAE authentication protocol employs the **Zero Knowledge Proof (ZKP)** [14] to prevent the leak of domain parameters during the handshakes. There have been two independent cryptanalysis studies of the Dragonfly key exchange protocol [4, 11]. In Reference [4], Clarke and Hao pointed out that the Dragonfly protocol potentially is subject to sub-group attacks due to the lack of Public Key Validation. However, to validate a public key would require a full exponentiation over the finite group, which significantly decreases the protocol's efficiency. In Reference [11], the authors prove that the Dragonfly protocol can achieve the Forward Secrecy in Random Oracle Model and as secure as SPEKE protocol.

In this article, it is further shown that SAE authentication protocol is also vulnerable to impersonation attacks as results of weaker randomness generation algorithms. These could enable an adversary to bypass the authentication, by simply choosing a weaker random value during the key exchange protocol.

This article further reviews the security properties of the SAE authentication protocol and outlines some additional attacks using the chosen random value impersonation attack methodology. At the end of this article are outlined some of the potential protection schemes for these attacks. Section 2 outlines the SAE authentication protocol and its known attacks. Section 3 illustrates further impersonation attacks on SAE based on the choice of weak random numbers during the protocol exchange. Section 4 discusses implications of the impersonation attack. Finally, Section 5 outlines examples of additional measures that may be employed in SAE authentication protocol to protect against these attacks. It is recommended that future implementations make provision to guard against these attacks and that installed systems be upgraded with new protection.

2 THE SAE AUTHENTICATION PROTOCOL

This section provides a brief outline of the mechanism of the basic Simultaneous Authentication of Equals protocol. SAE authentication protocol is based on discrete logarithm cryptography and the ZKP system. An implementation of SAE authentication protocol can either use operations on a finite field or an elliptic curve group. No assumptions are made about the underlying group, other than that the computation of discrete logarithms is sufficiently computationally difficult for the level of security required. The SAE protocol is made up with two operational phases: (I) The first phase is an element operation that takes an input of two elements and outputs a third element and a scalar operation that takes an input of an element and a scalar and outputs an element. This password mapping to password element function is sometimes referred to as "hunting and pecking." In this process, the password is mapped into a group element. After a **Password Element (PE)** is generated at the "hunting and pecking" phase, both parties engage in the Dragonfly key exchange protocol to derive the session keys. (II) The second phase is the authenticated key exchange protocol, which will be discussed further in Section 2.1, as our attacking method is concentrating on the weakness of the key exchange protocol of phase II.

2.1 Outline of SAE Authentication Protocol

We take the finite field as an example. Let us define p as a large prime. We denote a finite cyclic group Q, which is a subgroup of Z_P^* of prime order q. Hence, $q \mid p - 1$. We denote the element operation A, B for elements A and B, and the scalar operation $b \cdot A$ for element A and scalar b. These notations are in line with those commonly used when working over a finite field.

The SAE authentication protocol requires both participants to have an identical representation of the password/passphase. *Alice* and *Bob* have a shared password from which each can deterministically generate a password element $PE \in Q$. The algorithms to map an arbitrary password to an element in Q are specified in References [6, 7]. The generation of the PE, however, is not within our attack, and we will not repeat the procedure here.

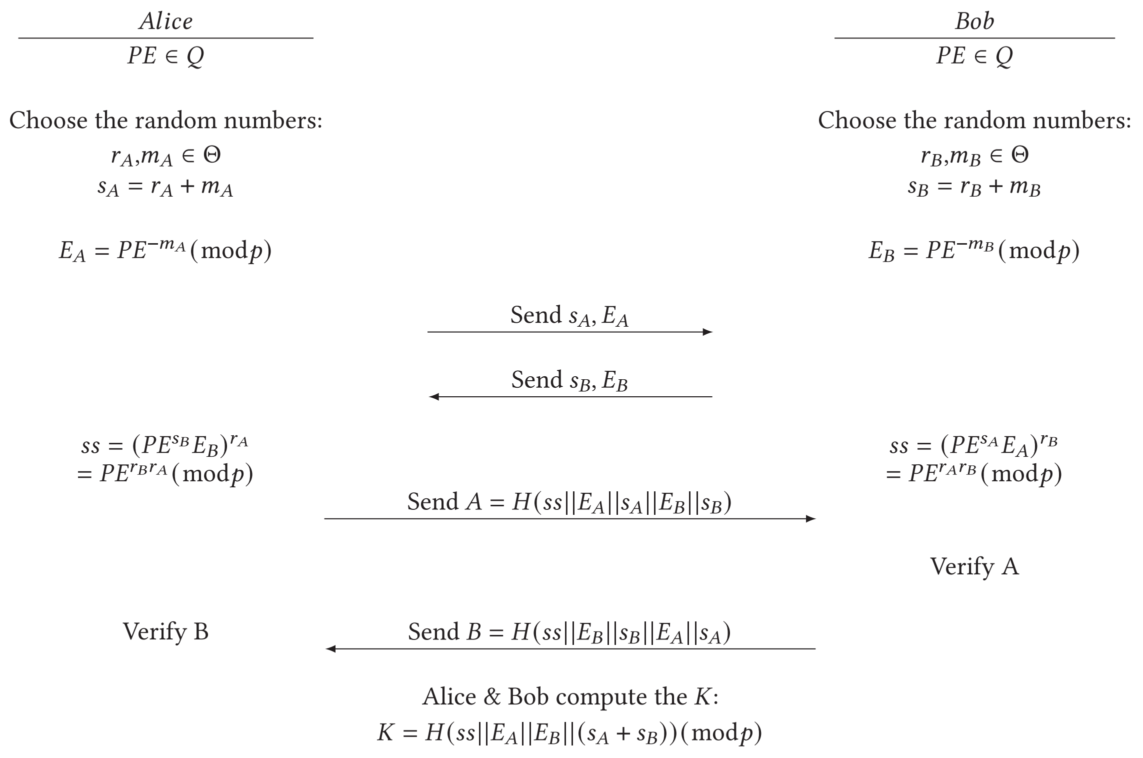
Alice
$PE \in Q$
Bob
$PE \in Q$
Choose the random numbers:
$r_A, m_A \in \Theta$
$s_A = r_A + m_A$
$E_A = PE^{-m_A}(\mathrm{mod}\, p)$
Choose the random numbers:
$r_B, m_B \in \Theta$
$s_B = r_B + m_B$
$E_B = PE^{-m_B}(\mathrm{mod}\, p)$
Send s_A, E_A
Send s_B, E_B
$ss = (PE^{s_B}E_B)^{r_A} = PE^{r_B r_A}(\mathrm{mod}\, p)$
$ss = (PE^{s_A}E_A)^{r_B} = PE^{r_A r_B}(\mathrm{mod}\, p)$
Send $A = H(ss||E_A||s_A||E_B||s_B)$
Verify A
Verify B
Send $B = H(ss||E_B||s_B||E_A||s_A)$
Alice & Bob compute the K:
$K = H(ss||E_A||E_B||(s_A + s_B))(\mathrm{mod}\, p)$

Fig. 1. SAE Authentication Protocol.

A brief outline of the SAE protocol is as follows:

(1) *Alice* randomly chooses two variables, scalars r_A and m_A from a random space $\Theta : \{1, \ldots, q\}$, calculates the scalar $s_A = r_A + m_A(\mathrm{mod}\, q)$ and the element $E_A = P^{-m_A}(\mathrm{mod}\, p)$, and sends s_A and E_A to *Bob*.
(2) *Bob* randomly chooses two variables, scalars r_B, m_B from the same random space $\Theta : \{1, \ldots, q\}$, calculates the scalar $s_B = r_B + m_B(\mathrm{mod}\, q)$ and $E_B = P^{-m_B}(\mathrm{mod}\, p)$, and sends s_B and E_B to *Alice*.
(3) *Alice* calculates the shared secret $ss = (PE^{s_B}E_B)^{r_A} = PE^{r_B r_A}(\mathrm{mod}\, p)$.
(4) *Bob* calculates the shared secret $ss = (PE^{s_A}E_A)^{r_B} = PE^{r_A r_B}(\mathrm{mod}\, p)$.
(5) *Alice* sends $A = H(ss||E_A||s_A||E_B||s_B)$ to *Bob*, where H is a predefined cryptographic hash function.
(6) *Bob* sends $B = H(ss||E_B||s_B||E_A||s_A)$ to *Alice*, where H is the same hashing function as Alice's.
(7) *Alice* and *Bob* check that the hashes are correct and if they are, then *Alice* and *Bob* can create a shared key $K = H(ss||E_A||E_B||(s_A + s_B))(\mathrm{mod}\, p)$.

This SAE authentication protocol is illustrated in Figure 1.

The SAE authentication protocol at this stage has successfully generated the shared key K, which can then be fed into the Pairwise Master Key for the subsequent IEEE 802.11 four-way handshake key installation.

3 A CHOSEN RANDOM VALUE ATTACK ON SAE KEY EXCHANGE PROTOCOL

In this section,we demonstrate an attacking method that enables an adversary, identified as *Eve* but impersonating as *Bob* in the protocol, to compromise the SAE key exchange by carefully choosing the random value, r_B and m_B.

3.1 Attack Methodology

The new attack is a form of a chosen random value attack. The SAE protocol was carefully designed to thwart dictionary attacks by utilizing groups containing small subgroups that obfuscate the group domain parameters during the handshakes [2]. The group is only identified by a group identifier that both participants agree to use. However, the SAE authentication protocol fails to fully specify how the random values, r_A, m_A, r_B, m_B must be chosen. The protocol simply specifies them as randomly generated from random space Θ. We observe that, with some probability, an adversary, *Eve*, is able to impersonate *Bob* during the SAE authentication by choosing weaker random values.

It is feasible for the adversary, *Eve* impersonating *Bob*, to carefully choose the random values r_B and m_B during the initialization of the SAE key exchange protocol. If *Eve* has knowledge of the proper group domain parameters to fully implement the SAE authentication protocol, then *Eve* only lacks knowledge of the pre-shared credentials of the valid party. In 2014, Clarke and Hao [4] properly pointed out a type of impersonation attack that could take advantage of the SAE protocol's lack of public key validation during the key exchange. To address this attack, the authors of Reference [4] recommended a remedy of performing a group element validation when receiving E_B. In our attacking model, we assumed the SAE authentication protocol has been extended to include the remedy in Reference [4] even though it could be a costly solution.

In the further attack, it is assumed that *Alice* and *Bob* share a common password and *Eve* is able to eavesdrop all the SAE protocol conversation and hijacks the session by spoofing $Bob's$ identity, which is transparent during the authentication protocol. In effect, *Eve* may impersonate *Bob* becoming "*Rob*." Without loss of generality, we assume *Alice* initiates an SAE session with *Bob* by sending S_A and E_A to *Bob*. When *Eve* intercepts the message during the transmission, it is feasible for *Eve* to intentionally respond with $r_B = 0$, and with m_B chosen arbitrarily from the random space Θ. The adversary's choice of $r_B = 0$ is masked by the obfuscating techniques of $s_B = r_B + m_B (\text{mod} q)$ specified in the SAE key exchange protocol. Thus, the carefully chosen $r_B = 0$ won't be detectable by *Alice* even assuming *Alice* enforces the public key validation at the time of reception.

One challenge with *Eve* is how to construct the E_B given *Eve* has no chance to guess the pre-shared password between *Alice* and *Bob*, hence no way to guess the Password Element during the "hunting and pecking" phase. However, the adversary, *Eve* can take advantage of the fact *Alice* first shared the s_A and E_A with *Bob*, which means the *Eve* can obtain the value of E_A and s_A. An approach would be that *Eve* could reconstruct E_B by setting $E_B = E_A^{m_B} = PE^{-m_A m_B} (\text{mod}\, p)$. At the end of construction, *Eve* launches the attack by sending the s_B and E_B to *Alice* (in effect, spoofing *Bob*). After receiving s_B and E_B, *Alice* then computes the $ss = (PE^{s_B} E_B)^{rA} = PE^{m_B(1-m_A)r_A} (\text{ mod } p)$, which is then used in computation of the verification value A. At $Eve's$ side, *Eve* just computes the $ss'= PE^{r_B r_A} (\text{mod} p) = 1 (\text{mod} p)$ and then constructs the verification value $B' = H(ss||E_B||s_B||E_A||s_A)$ where the $ss = 1(\text{mod} p)$.

Without loss of generality, depending on how *Alice* randomly chose the r_A, assuming with certain probability of p, the $r_A = 1$ was chosen or due to implementation flaw, this will cause the *Alice* to obtain the $ss = (PE^{s_B} E_B)^{rA} = PE^{mB(1-mA)rA} (\text{mod} p) = 1(\text{mod} p)$.

Thus, we can say that with probability of *prob*, *Eve* will bypass the SAE authentication and successfully launch an impersonation attack. As we have pointed out earlier, the chances of bypassing the SAE authentication lies in the possibility of Alice choosing the $m_A = 1$, $prob = \frac{1}{q}$, assuming m_A is generated uniformly from a random function.

This chosen random value attack on SAE authentication protocol is illustrated in Figure 2.

3.2 Extension of Chosen Random Value Attack with Dictionary

Similarly, the chosen random value attack could be extended to a dictionary type of attack if the random space is relatively small.

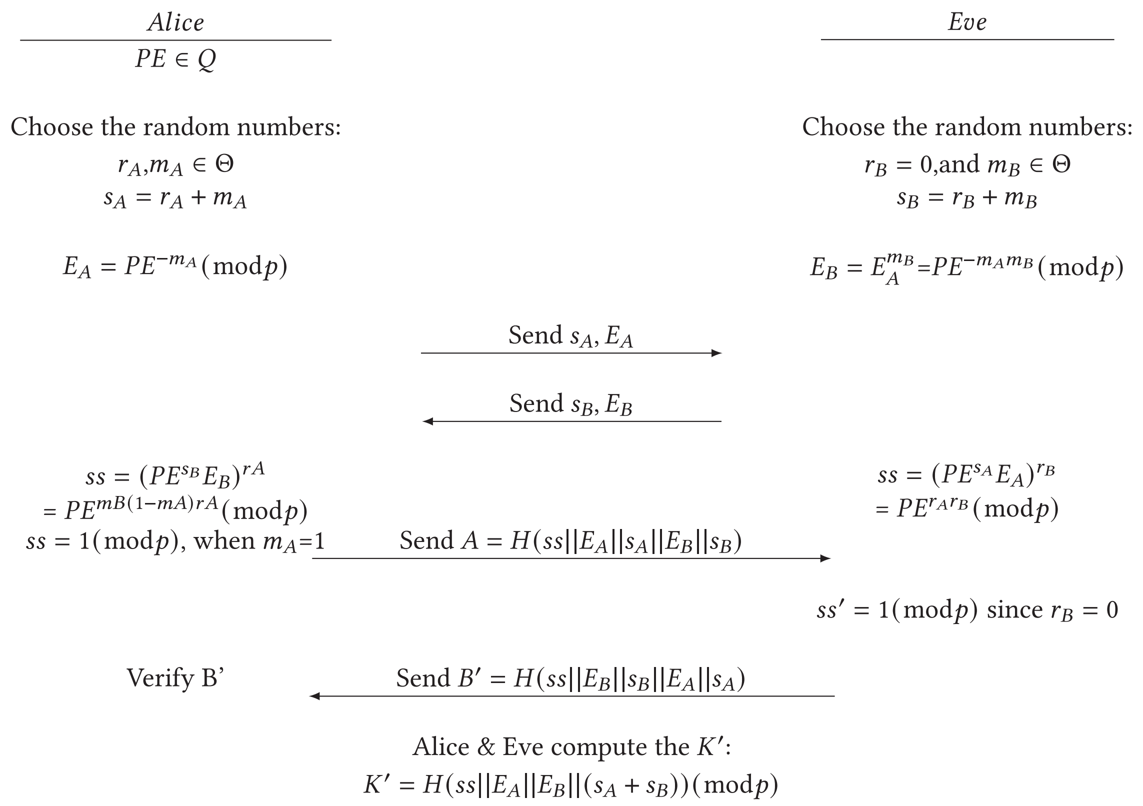
Alice
$PE \in Q$
Eve
Choose the random numbers:
$r_A, m_A \in \Theta$
$s_A = r_A + m_A$
$E_A = PE^{-m_A}(\mathrm{mod}\,p)$
Choose the random numbers:
$r_B = 0$, and $m_B \in \Theta$
$s_B = r_B + m_B$
$E_B = E_A^{m_B} = PE^{-m_A m_B}(\mathrm{mod}\,p)$
Send s_A, E_A
Send s_B, E_B
$ss = (PE^{s_B} E_B)^{rA}$
$= PE^{mB(1-mA)rA}(\mathrm{mod}\,p)$
$ss = 1(\mathrm{mod}\,p)$, when m_A=1
Send $A = H(ss||E_A||s_A||E_B||s_B)$
$ss = (PE^{s_A} E_A)^{r_B}$
$= PE^{r_A r_B}(\mathrm{mod}\,p)$
$ss' = 1(\mathrm{mod}\,p)$ since $r_B = 0$
Verify B'
Send $B' = H(ss||E_B||s_B||E_A||s_A)$
Alice & Eve compute the K':
$K' = H(ss||E_A||E_B||(s_A + s_B))(\mathrm{mod}\,p)$

Fig. 2. The chosen random value attack on SAE protocol.

If the adversary, then *Eve* could let E_B be equal to the square root of E_A over the group Q, which calculates the E_B= $|\sqrt{E_A}|$ = $|PE^{\frac{-m_A}{2}}|(\mathrm{mod}\,p)$. To make sure there exist square roots of E_A over Q, it is imperative to check if the received E_A is quadratic residue over Q by checking the legendre symbol, specifically check if $(\frac{E_A}{p}) = 1$. If E_A is a quadratic residue over p, then *Eve* will proceed with performing the off-line random value guessing of the scalar equivalence, which is from the random space Θ. The well-chosen random value and E_B can be used to obtain the intermediate key $ss = (PE^{s_B} E_B)^{rA} = PE^{(m_B - \frac{m_A}{2})r_A}(\mathrm{mod}\,p)$, with $r_B = 0$. There exist many pairs of m_B and $m_A \in \Theta$ that could result in the $ss = 1(\mathrm{mod}\,p)$ at *Alice*. We hereby denote the pairs as scalar equivalence. Assuming the scalar equivalence exist, *Alice* will obtain the $ss = 1(\mathrm{mod}\,p)$. When *Alice* transmits the s_A and m_A to *Bob*, The adversary, *Eve* launches the dictionary attack by finding the scalar equivalence in Θ that can obtain the intermediate key $ss = PE^{(m_B - \frac{m_A}{2})r_A}(\mathrm{mod}\,p) = 1(\mathrm{mod}\,p)$. Then *Eve* sends the s_B and m_B to *Alice*. With the expectation that *Alice* will obtain the $ss = 1(\mathrm{mod}\,p)$. She computes

$$m_B - \frac{m_A}{2} = 0, \tag{1}$$

since *Eve* received the $s_A = r_A + m_A$; hence,

$$m_B - \frac{s_A - r_A}{2} = 0, \tag{2}$$

$$s_A = 2 \cdot m_B + r_A(\mathrm{mod}\,q). \tag{3}$$

Assuming the random space Θ is not sufficiently large, to find the scalar equivalence $\{m_B, m_A\}$, *Eve* could take advantage of Equation (3) to build an off-line dictionary to find the proper $\{m_B, r_A\}$ pairs, which are also the scalar equivalence. Then, *Eve* sends the s_B and m_B to *Alice*. With expectation of *Alice* to obtain the $ss = 1(\bmod q)$, *Eve* calculates the $B' = H(ss = 1||E_B||s_B||E_A||s_A)$ and transmits to *Alice* for verification. *Eve* runs the dictionary attack of Algorithm 1:

ALGORITHM 1: Extension chosen random value attack on SAE protocol

```
Input   : $s_A$, $m_B$, $r_A$, $m_A$, $r_B = 0$
Output : $B'$, $K'$
while each $m_B$ and $r_A$ in $\Theta$ do
    find the pairs $m_B, r_A$ which satisfy:
    $s_A = 2 * m_B + r_A (\bmod q)$;
    return $m_B, r_A$ as new dictionary
end
while each $m_B$ in dictionary do
    $s_B = m_B$
    $E_B = |\sqrt{E_A}| = |PE^{\frac{-m_A}{2}}|(\bmod p)$
    $ss' = (PE^{s_B} E_B)^{rA} = PE^{(\frac{m_B - m_A}{2})r_A}(\bmod p)$
    $A' = H(ss = 1||E_A||s_A||E_B||s_B)$
    if $A = A'$ then
        $B' = H(ss = 1||E_B||s_B||E_A||s_A)$
        $K' = H(ss = 1||E_A||E_B||(s_A + s_B))$
        return $B'$, $K'$
    else
end
```

To perform the above attack, the adversary *Eve* needs to find out the scalar equivalence within the random space Θ of the dictionary. As long as the size of the random space Θ is manageable, it is not difficult for an adversary to determine random pairs or keys within the random space that are equivalent. This is particularly true when the SAE authentication protocol is applied to devices using **Personal Identification Numbers (PIN)** or short passwords. These applications often chose to limit the random space to improve the performance.

The extension chosen random value attack above lies in computing square roots over primes, which is hard. Nonetheless, it could be argued that square roots modulo a prime p can be computed faster by Sarkar's Square Roots algorithm [3].

Similar to this extension random value attack, *Eve* can also reuse the E_A, similar to reflection attacks, let $E_B = E_A$, then the scalar equivalence is to satisfy $s_A = m_B + r_A(\bmod q)$. The attack algorithm is similar. This construction is much simpler to implement with less computation required.

4 DISCUSSION

It is observed that the SAE authentication protocol in Reference [6] does not fully specify the random space Θ and its requirements. Unlike SPEKE protocol, SAE protocol relaxes the requirements that p must be a safe prime (i.e., $p = 2 \cdot q + 1$), which is also observed in Reference [4]. These factors enable practical attacks on SAE protocol that could result in widely deployed devices being compromised. In addition to the subgroup attack in Reference [4], if the E_B is carefully constructed based on the received E_A, then *Alice* will not be able to detect the chosen random value attack even with the public key validation techniques within the SAE protocol. The rationale of transmitting the scalars in the SAE authentication protocol, and the scalar construction using addition instead of other safer operations, such as multiplication is unknown. This technique may be related to obfuscating the random value r during the transmission to prevent the Man-in-the-Middle attacks and to help the protocol efficiency. However, it seems this technique actually reduces the security.

5 PREVENTING CHOSEN RANDOM VALUE ATTACKS ON THE SAE PROTOCOL

There are several ways to prevent the chosen random value attacks on SAE key exchange protocol.

(1) The chosen random value attack can be prevented by implementing a validation of the intermediate key ss within the SAE authentication protocol. If $ss = 1(\bmod p)$, then it is very likely an indication of a chosen random value attack. Such indication should cause *Alice* to abandon the SAE authentication process.
(2) There are possible other ways to construct the E_B by utilizing the intercepted s_A and E_A. It is always the best practice to implement the full public key validation as specified in NIST SP 800-56A [2], to check if the intermediate key ss belongs to the group $\in Q$. The public key validation as remedy was also recommended in Reference [4]. Nonetheless, it is well known that implementing the public key validation requires the exponentiation and would reduce the performance of the SAE authentication protocol.
(3) As we have observed from the chosen value attack, it is imperative to define the random space Θ as $\{2 \ldots, q-1\}$, instead of random space from $\{1, \ldots, q\}$ to avoid the weak random values.
(4) It is recommended to use the large random space Θ with large prime number $q \in Z_p^*$ to decrease the possibility of chosen random value attacks. If q is divisible by other numbers, then the extension attacks could become more efficient by creating subgroups of the random space Θ.

6 CONCLUSIONS

We demonstrated that the SAE protocol is vulnerable to a chosen random value attack and its extension attacks by computing the dictionary within the random space Θ, in which some weak random values could lead to the SAE protocol being bypassed by a third party without the knowledge of the password. These attacks are more practical to implement in scenarios where the random space Θ is relatively small, for instance, PIN- or short-password-based environments. It might be argued that the risk of such attacks could be mitigated by anti-clogging mechanisms and by thresholding unsuccessful impersonation attempts. Nonetheless, our analysis indicates the SAE authentication protocol could be compromised by online and offline dictionary attacks. For SAE, the adversary could gain information about the random value chosen by the valid users by guessing the scalar equivalence.

ACKNOWLEDGMENTS

The author thanks David Steer, Sophie Vrzic, Wen Tong, Peiying Zhu, and Rene Struik for their reviews and valuable comments.

REFERENCES

[1] National Institute of Standards and Technology. 2013. Digital Signature Standard (DSS). Retrieved from https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf.
[2] National Institute of Standards and Technology. 2013. Recommendation for Pair-wise Key Establishment Schemes Using Discrete Logarithm Cryptography. Retrieved from https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-56ar2.pdf.
[3] International Association for Cryptologic Research. 2021. Computing Square Roots Faster than the Tonelli-Shanks/Bernstein Algorithm. Retrieved from https://eprint.iacr.org/2020/1407.pdf.
[4] D. Clarke and F. Hao. 2014. Cryptanalysis of the Dragonfly key exchange protocol. *IET Info. Secur.* 8 (Nov. 2014). Retrieved from https://www.dcs.warwick.ac.uk/~fenghao/files/Dragonfly_final.pdf.
[5] Whitfield Diffie, Paul C. Van Oorschot, and Michael J. Wiener. 1992. Authentication and Authenticated Key Exchanges. Retrieved from https://people.scs.carleton.ca/~paulv/papers/sts-final.pdf.
[6] Dan Harkins. 2012. Simultaneous Authentication of Equals. IEEE. Retrieved from https://www.ieee802.org/11/#.
[7] Dan Harkins. 2015. RFC7664-Dragonfly Key Exchange. Retrieved from https://datatracker.ietf.org/doc/html/rfc7664.
[8] Dan Harkins. 2019. RFC8492-Secure Password Ciphersuites for Transport Layer Security (TLS). HP Enterprise. Retrieved from https://datatracker.ietf.org/doc/html/rfc8492.
[9] D. Harkins and D. Carrel. 1998. RFC2409-The Internet Key Exchange (IKE). Retrieved from https://datatracker.ietf.org/doc/html/rfc2409.

[10] T. Kivinen and M. Kojo. 2019. RFC3526-More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE). Retrieved from https://datatracker.ietf.org/doc/html/rfc3526.

[11] J. Lancrenon and M. Skrobot. 2015. On the Provable Security of the Dragonfly Protocol. In *Proceedings of the 18th International Information Security Conference (ISC'15)*.

[12] David P. Jablon. 1996. Strong password-only authenticated key exchange. *ACM SIGCOMM Computer Communication Review* 26, 5 (Oct 1996). https://doi.org/10.1145/242896.242897

[13] J. Salowey and S. Winter. 2013. RFC7057-Update to the Extensible Authentication Protocol (EAP) Applicability Statement for Application Bridging for Federated Access Beyond Web (ABFAB). RESTENA, Cisco. Retrieved from https://datatracker.ietf.org/doc/html/rfc7057.

[14] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. 1989. The knowledge complexity of interactive proof systems. Issue 1. https://doi.org/10.1137/0218012

Received December 2020; revised May 2021; accepted May 2021