Security Proof for Password Authentication in TLS - Verifier-based Three-Party Group Diffie-Hellman

Olivier Chevassut[1], Joseph R. Milner[1], and David Pointcheval[2]

[1] Computational Research Division, Lawrence Berkeley National Laboratory, USA
[2] Technology Transfer and Intellectual Property Management Department, Lawrence Berkeley National Laboratory
[3] Computer Science Department, École normale supérieure, France

Abstract. The internet has grown greatly in the past decade, by some numbers exceeding 47 million active web sites and a total aggregate exceeding 100 million web sites. What is common practice today on the Internet is that servers have public keys, but clients are largely authenticated via short passwords. Protecting these passwords by not storing them in the clear on institutions's servers has become a priority. This paper develops password-based ciphersuites for the Transport Layer Security (TLS) protocol that are: (1) resistant to server compromise; (2) provably secure; (3) believed to be free from patent and licensing restrictions based on an analysis of relevant patents in the area.

Keywords: Encrypted Key Exchange, Group Diffie-Hellman Key Exchange, TLS.

1 Introduction

The internet has grown greatly in the past decade, by some numbers exceeding 47 million active web sites and a total aggregate exceeding 100 million web sites. Internet transactions encompass all forms of commercial transactions, often where secure information such as credit cards, social security numbers, and bank accounts are necessary to a set of transactions. Ultimately, businesses and banks become hacking targets of the confidential data relating to their customers. Such data is tempting to steal as a jumping off point for identity theft or direct credit card usage. A much easier target than literally robbing a bank is to target the internet transactions of a bank, which may be done in a variety of technical means. Thus, it becomes increasingly important that passwords never be directly transmitted to a financial institution, as they may be compromised during transit.

What is common in practice today is that servers have public keys, but end-users (clients) will largely be authenticated based on their human-memorizable passwords. One reason for this is that security infrastructures aim to maintain and respect existing local security architectures of its users, and these are largely password-based. Another is that users find passwords easier to use. Indeed, even if a user has a public key, its matching secret key is stored on a server and the user accesses it via a password. For all these reasons, a central problem is how a client can establish a secure channel between itself and a server based on a password. As is standard, the central element here is to execute an authenticated exchange of a session key [1]. This is a well-studied problem, an notably from the seminal EKE work (which stands for Encryption Key Exchange) [BM92,BPR00,BMP00]; however, what differentiates the present paper from previous work is that we want to mitigate the damage down by system compromise, by making the server to store a transformation of the password only [BM93,ACP05,GMR06].

Previous works have assumed servers or other parties to be honest, but we know that system compromise is a reality which can put secret-holding servers in the hands of adversaries. System compromise is a reality that should not be ruled out. It is unrealistic to expect full security in the presence of this threat: if a hacker breaks into a server that stores client passwords, the

[1] The secure channel is then implemented via symmetric encryption and authentication under the session key.

passwords are immediately lost. This is what happens in the *pw-pw* model that have been considered in [BM92,BPR00,BMP00,BCP03,BCP04,ABC+07]; where the server and client both share the password.

In this paper we propose to instead consider the *pw-f(pw)* model, where the server stores the image of the password under a one-way function f [BM93,ACP05,GMR06]. Now if an adversary breaks into a server, it obtains only *f(pw)*. If *pw* is well-selected, meaning not in a too small dictionary, the adversary will be unable to recover efficiently *pw*. If it is a poor password, meaning in a small dictionary, then the adversary can mount a dictionary attack and recover *pw* in the time for a number of computations of f equal to the size of the dictionary, but at least this means it has to work harder, slowing down the attack and giving the servers administrator time to react appropriately and inform her clients [2]. The *pw-f(pw)* model thus provides greater security in the face of system compromise, particularly for those users savvy enough to choose good passwords, but also to some extent for others, particularly if a salt is used and the break-ins are detected in a reasonable amount of time. One refers as *verifier-based* the *pw-f(pw)* scenario.

Organization of the paper. The rest of this short paper is organized as follows. In Section 1 we describe the cryptographic protocol and its integration in TLS. This is a necessary step toward an implementation in an open-source cryptographic library such as OpenSSL. In Section 2, we show that TLS-V3SOKE is provably secure under reasonable computation assumptions. We then finally conclude the paper.

2 The Patent Issue

With the proliferation of internet applications and business, there is much more monetary value associated with the corruption of passwords. Individuals and businesses compete in the internet marketplace using traditional method in intellectual property protection: trade secret, copyright, trademark, and patents. One difficulty in the area of password-based key exchange is the existence of patents covering the two cases [ABC+07]. The seminal patent in the area of two party secure communications is that of Steven M. Bellovin and Michael J. Merritt, "Cryptographic Protocol for Secure Communications", which issued on August 31, 1993 as United States patent 5,241,599 (the '599 patent), based on [BM92]. This patent discloses a method which permits computer users to authenticate themselves to a computer system, with password-based authentication. A method was devised that is believed to be practicable for two party secure communications without infringing the '599 patent. This method has a pending United States patent application, which was published by the United States Patent and Trademark Office as US-2005-0157874-A1 on July 21, 2005.

Similarly, the same Bellovin and Merritt were coinventors for "Cryptographic Protocol for Remote Authentication" in United States patent 5,440,635 (the '635 patent) based on [BM93], which issued on August 8, 1995. The '635 patent discloses a cryptographic communication system that employs a combination of public and private key cryptography, allowing two players, who share only a relatively insecure password, to bootstrap a computationally secure cryptographic system over an insecure network. The '635 patent system is secure against active and passive attacks, and has the property that the password is protected against offline "dictionary" attacks. This patent tackles the issue of verifier-based password key exchange.

Although the '635 patent was only issued in the United States, it still remains an obstacle to unfettered remote authentication. Therefore, an alternative protocol is proposed here that differs

[2] The work the attacker must do can also be increased by making the function f slow to compute. We can make the dictionary attack even harder by salting, where the server stores *f(salt, pw)*, for some random salt that is public but differs from user to user.

from the claimed '635 invention. From a United States patent law perspective, the determination of whether a claim is infringed based on the "all elements" approach to the doctrine of equivalents.

"Under the 'all elements' rule, there can be no infringement under the doctrine of equivalents if even one limitation of a claim or its equivalent is not present in the accused device or method."

Thus, if one element or its equivalent of the '635 is missing, then there can be no infringement of that claim. In the '635 claim, there are only independent claims 1 and 9. In patent law, if an independent claim is not infringed, then a claim depending from the independent claim (with still more limitations, hence narrower) is also not infringed. Thus, one only needs to initially analyze claims 1 and 9. If neither of these claims is infringed, then the entire patent is not infringed.

3 Cryptographic Protocol and Its Integration in TLS

Password derivation. The actual common information between the client and the server is $v = U^{pw}$. However, the client likely wants to see the group being used before typing his password string *password*, which should actually be hashed before being used as an exponent pw. Furthermore, one may use (C, S, pw) instead of just *password* to derive the exponent, we obtain a security improvement in practice in that a client can use the same *password* (string) with multiple servers and yet the respective secrets (effective passwords) will be different: One may thus use $pw = \mathcal{G}(C \parallel S \parallel U \parallel pw) \bmod q$, and $v = U^{pw}$.

Algorithm (see Fig 1). The Client and the Server first agree on the ciphersuite to use which in our case is TLS-V3SOKE. Once the cipersuite is agreed upon, the Server can initiate the 3-party (dynamic) group Diffie-Hellman key exchange [ABC+06]. The Server's cryptographic accelerator generates a value x_1 at random and computes the output g^{x_1} which is transmitted to the Server's CPU. The Server's CPU then picks at random a first value x_2' which is used to compute the verifier as $U^{x_2'}$, and a second value x_2 in order to compute the three Diffie-Hellman values $g^{x_1}, g^{x_2}, g^{x_1 x_2}$. The Server then forms the output flow consisting of its identity, the three Diffie-Hellman values, and the verifier. In TLS language, this flow is called the `ServerKeyExchange` message.

Once the Client receives the `ServerKeyExchange` message, the Client generates a random value x_3 and computes the Diffie-Hellman value $g^{x_1 x_3}$ and sees the Server's CPU has having left the group —for details see Bresson et al. [BCP02]. The Client encrypts the Diffie-Hellman value and sends its to the Server. In TLS language, this flow is called the `ClientKeyExchange` message.

The Client and the Server compute the group shared secret key $g^{x_1 x_2 x_3}$ and the common verifier V^{pw}. They then both use the latter value as part of the calculation of the `PreMasterSecret`. The rest of the handshake is identical to the handshake presented by Abdalla et al. in [ABC+06].

4 Security Theorem and its Proof

Theorem 1 (FS-AKE Security). *Let us consider protocol from Figure 1 over a group of prime order q, where* Password *is a dictionary of size N, equipped with the uniform distribution. Let $\mathcal{A}$ be an adversary running within a time bound t that makes less than q_{active} active sessions with the parties, and asks q_{hash} hash queries. Let us assume that the CDH problem is hard in this group, and the PRFs and the MAC are secure, then we have,*

$$\mathsf{Succ}^{\mathsf{auth}_S}_{\mathsf{VSOKE}}(\mathcal{A}) \leq \frac{q_{\mathsf{active}}}{N} + \frac{q_{\mathsf{hash}} N}{q} + negl() \quad \mathsf{Adv}^{\mathsf{ake-fs}}_{\mathsf{VSOKE}}(\mathcal{A}) \leq \frac{4q_{\mathsf{active}}}{N} + \frac{4q_{\mathsf{hash}} N}{q} + negl().$$

Proof. We are interested in the event S, which occurs if the adversary correctly guesses the bit b involved in the Test-queries. We furthermore consider server (unilateral) authentication: event A is

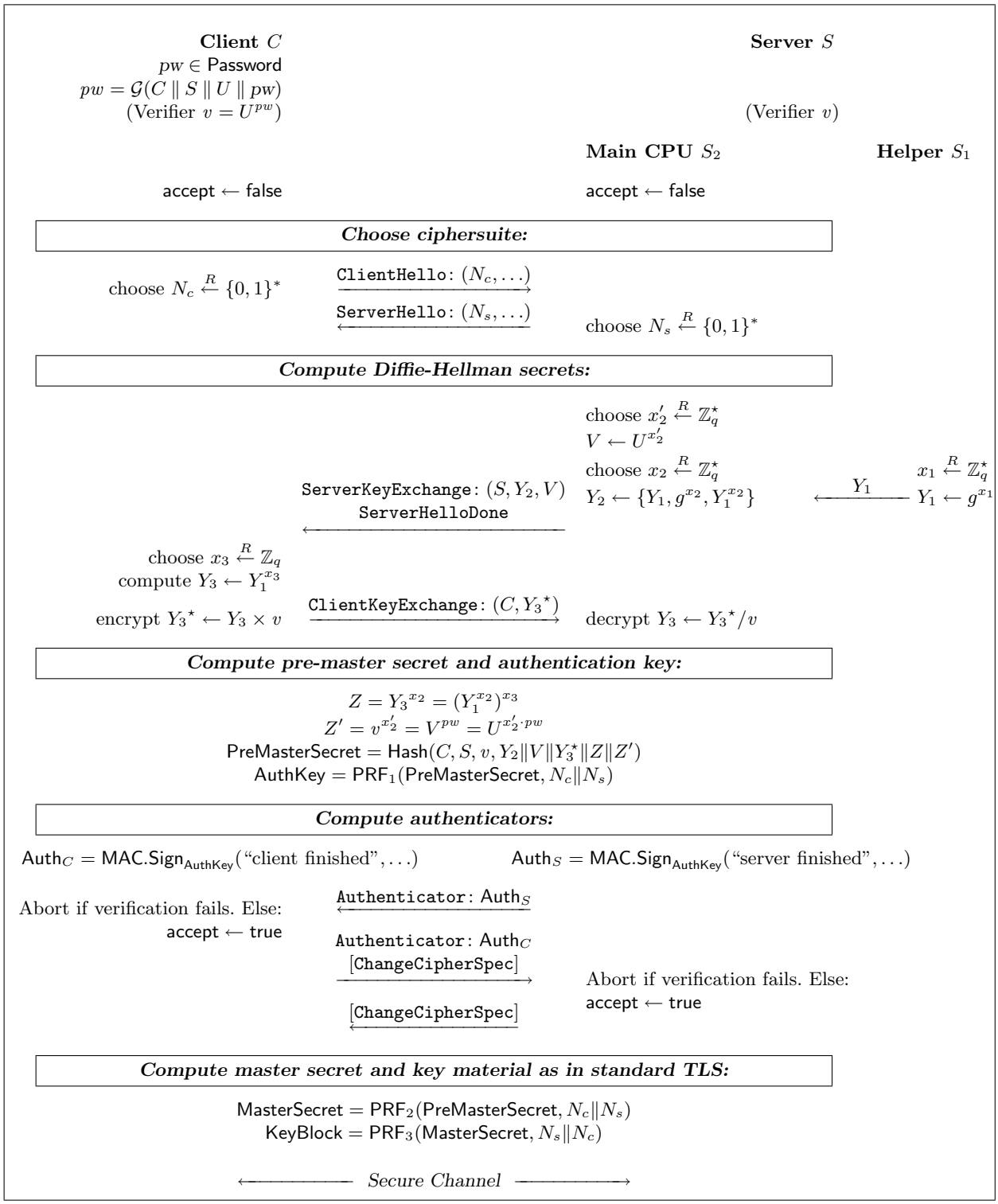

Fig. 1. The TLS-V3SOKE ciphersuites.

set to true if a client instance accepts, without any server partner. Let us remember that in this attack game, the adversary is allowed to use Corrupt-queries.

Game $\mathbf{G}_0$: This is the real protocol, in the random-oracle model:

$$\mathsf{Adv}^{\mathsf{ake-fs}}_{\mathsf{VSOKE}}(\mathcal{A}) = 2\Pr[\mathsf{S}_0] - 1 \quad \mathsf{Adv}^{\mathsf{auth}_S}_{\mathsf{VSOKE}}(\mathcal{A}) = \Pr[\mathsf{A}_0].$$

Let us furthermore define the event $\mathsf{S}_{w/t}\mathsf{A} = \mathsf{S} \wedge \neg\mathsf{A}$, which means that the adversary wins the *Real-Or-Random* game without breaking authentication.

Game $\mathbf{G}_1$: In this game, we simulate the hash oracles (Hash, but also an additional hash function $\mathsf{Hash}' : (\{0,1\}^\star)^3 \to \{0,1\}^\ell$ that will appear in the Game $\mathbf{G}_4$) as usual by maintaining hash lists Λ_{Hash} and $\Lambda_{\mathsf{Hash}'}$ with all the queries-answers asked to the hash functions. We also simulate all the instances, as the real players would do, for the Send-queries and for the Execute, Test and Corrupt-queries.

Game $\mathbf{G}_2$: We cancel games in which some collisions appear on the transcripts $(C, S, Y_2, V, Y_3^\star)$, and on the master secrets. Regarding the transcripts, the distance follows from the birthday paradox since at least one element of each transcript is generated by an honest participant (at least one of them in each of the q_{active} active attacks, and all of them in the q_{passive} passive attacks). Likewise, in the case of the master keys, a similar bound applies since Hash is assumed to behave like a random oracle (which outputs ℓ-bit bit-strings): $\Pr[\mathsf{Coll}_2] = \mathit{negl}()$.

Game $\mathbf{G}_3$: We cancel games in which for some hash query Hash on $(C, S, v, Y_2\|V\|Y_3^\star\|Z\|Z')$, there exists $pw \in \mathsf{Password}$ such that $v = U^{\mathcal{G}(C\|S\|U\|pw)}$, but the $\mathcal{G}$-query has not been asked by the adversary. We denote by $\mathsf{NotAskPW}_3$ this bad event: $\Pr[\mathsf{NotAskPW}_3] \leq \frac{q_{\mathsf{hash}} N}{q}$. This is the main difference from previous proofs [ABC+07], since the reduction needs to make the exhaustive search on the dictionary, hence N must be large, but not too large: say 40-bit entropy.

Game $\mathbf{G}_4$: In this game, we show that the success probability of the adversary is negligible in passive attacks via Execute-queries. To do so, we modify the way in which we compute the pre-master secret PreMasterSecret in passive sessions that take place before or after a Corrupt-query. More precisely, whenever the adversary asks an Execute-query, we compute the pre-master secret PreMasterSecret as $\mathsf{Hash}'(C, S, Y_2\|V\|Y_3^\star)$ using the private oracle Hash' instead of the oracle Hash. As a result, it holds that any value of PreMasterSecret computed during a passive session becomes completely independent of Hash, Z and Z', which are no longer needed in these sessions. Please note that the oracle Hash is still being used in active sessions.

The games $\mathbf{G}_4$ and $\mathbf{G}_3$ are indistinguishable unless the adversary $\mathcal{A}$ queries the hash function Hash on $(C, S, v, Y_2\|V\|Y_3^\star\|Z\|Z')$, for such a passive transcript: this (bad) event is denoted AskH-Passive-Exe. In order to upper-bound the probability of this event, we consider an auxiliary game $\mathbf{G}_4$', using a $\mathsf{CDH}_{g,\mathbb{G}}$-instance (U_1, U_2) as input, still knowing the password pw, in which the simulation of the players changes—but the distributions remain perfectly identical (therefore, $\Pr[\mathsf{AskH\text{-}Passive\text{-}Exe}_4] = \Pr[\mathsf{AskH\text{-}Passive\text{-}Exe}'_4]$), and at the same cost (5 exponentiations): We first set $U = U_1$. Since we do not need to compute Z for the simulation of Execute-queries, we can simulate Y_2 as $\{Y_1 = g^{x_1}, U_2^{x^\star}, U_2^{x^\star \cdot x_1}\}$, $V = U^{x'_2}$ as usual, and $Y_3^\star$ as $g^{y^\star}$, for known values of x_1, $x^\star$ and $y^\star$. This implicitly sets x_2 to $x^\star \cdot \log_g U_2$. If event AskH-Passive-Exe occurs, the value $Z = (g^{y^\star}/v)^{x^\star \cdot \log_g U_2}$ can be extracted from Λ_{Hash}, by simply choosing at random among the q_{hash} elements. Since the values $x^\star$, $y^\star$ and pw are known, we get $\mathsf{CDH}_{g,\mathbb{G}}(U_1, U_2)$: $\Pr[\mathsf{AskH\text{-}Passive\text{-}Exe}_4] \leq \mathit{negl}()$.

Game $\mathbf{G}_5$: In this game, we consider passive attacks via Send-queries, in which the adversary simply forwards the messages it receives from the oracle instances. More precisely, we replace Hash

by Hash' when computing the value of $\mathsf{PreMasterSecret}$ whenever the values $(C, S, Y_2, V, Y_3^\star)$ were generated by oracle instances. Note that we can safely do so due to the absence of collisions in the transcript. Like in $\mathbf{G}_4$, any value $\mathsf{PreMasterSecret}$ computed during such passive sessions becomes completely independent of Hash, Z and Z'.

As in previous games, we can upper-bound the difference in the success probabilities of $\mathcal{A}$ in games $\mathbf{G}_5$ and $\mathbf{G}_4$ by upper-bounding the probability that $\mathcal{A}$ queries the hash function Hash on $(C, S, v, Y_2\|V\|Y_3^\star\|Z\|Z')$, for such a passive transcript; we call this (bad) event $\mathsf{AskH\text{-}Passive\text{-}Send}$. Toward this goal, we consider an auxiliary game $\mathbf{G}_5$', in which the simulation of the players changes slightly without affecting the view of the adversary. In this simulation, we are given a $\mathsf{CDH}_{g,\mathbb{G}}$-instance (U_1, U_2), and choose at random one of the $\mathsf{Send}(S, \text{“start”})$-queries being asked to S and we reply as above, with $x^\star = 1$ since there is not need of random self-reducibility. If the event $\mathsf{AskH\text{-}Passive\text{-}Send}$ occurs and our guess for the passive session is correct (the adversary simply forwarded the messages), then we can extract $Z = U_2^{y^\star}/\mathsf{CDH}_{g,\mathbb{G}}(U_1, U_2)^{pw}$ from Λ_{Hash}. Similarly to above, we get: $\Pr[\mathsf{AskH\text{-}Passive\text{-}Send}_5] \leq \textit{negl()}$.

Game $\mathbf{G}_6$: In this game, we extend the replacement of the oracle Hash by the private oracle Hash' in any simulation, but before any corruption only: instead of computing $\mathsf{PreMasterSecret} = \mathsf{Hash}(C, S, v, Y_2\|V\|Y_3^\star\|Z\|Z')$, even when (Y_2, V) or Y_3 has been generated by the adversary, we set $\mathsf{PreMasterSecret} = \mathsf{Hash}'(C, S, Y_2\|V\|Y_3^\star)$, as long as no $\mathsf{Corrupt}$-query has occurred. Clearly, the games $\mathbf{G}_6$ and $\mathbf{G}_5$ are indistinguishable as long as $\mathcal{A}$ does not query the hash function Hash on an input $(C, S, v, Y_2\|V\|Y_3^\star\|Z\|Z')$, for some execution transcript $(C, S, Y_2\|V\|Y_3^\star)$. We denote this (bad) event by $\mathsf{AskHbC\text{-}Active}$.

Game $\mathbf{G}_7$: In this game, we replace the pseudo-random functions by truly random functions for all the sessions in which the value of $\mathsf{PreMasterSecret}$ has been derived with the private oracle Hash'. Since the value $\mathsf{PreMasterSecret}$ that is being used as the secret key for the pseudo-random function is independently and uniformly distributed, the distance can be proven by a classical sequence of hybrid games, where the counter is on the pre-master secrets. That is, each time a new pre-master secret is set, we increment the counter. Then, $\Pr[\mathsf{S}_{w/t}\mathsf{A}_7] = \frac{1}{2}$. But the difference involves the pseudo-randomness of the PRFs: $\Pr[\mathsf{Dist\text{-}PRF}_7] \leq \textit{negl()}$.

Game $\mathbf{G}_8$: In this game, we exclude collisions on MAC keys for all the sessions in which the pre-master secret $\mathsf{PreMasterSecret}$ has been derived with the private oracle Hash'(which event is denoted $\mathsf{CollPRF}$). For these sessions, the MAC keys of length ℓ_{M} are independently and uniformly distributed (because the PRF were replaced by random functions): $\Pr[\mathsf{CollPRF}_8] \leq \textit{negl()}$.

Game $\mathbf{G}_9$: In this game, we exclude games wherein for some transcript $(C, S, Y_2\|V\|Y_3^\star)$, there are two verifiers v_0 and v_1 such that the corresponding pre-master secrets lead to a collision of the MAC-values (which event is denoted CollM).

Since we know that MAC-keys are truly random and different from each other at this point, the event CollM means that a MAC with a random key (one of the q_{hash} possible values) may be a valid forgery for another random key. Thus, by randomly choosing the two indices for the hash queries, we get the following upper-bound: $\Pr[\mathsf{CollM}_9] \leq \textit{negl()}$.

Before proceeding with the rest of the analysis, we split the event $\mathsf{AskHbC\text{-}Active}$ into two disjoint sub-cases depending on whether the adversary impersonates the client (and thus interacts with the server) or the server (and thus interacts with the client). We denote these events $\mathsf{AskHbC}_w\mathsf{S}$ and $\mathsf{AskHbC}_w\mathsf{C}$, respectively. Also we denote by $q_{\mathsf{fake-server}}$ (respectively, $q_{\mathsf{fake-client}}$) the number of sessions in which the adversary impersonates the server (resp., the client). Obviously, one has $q_{\mathsf{fake-server}} + q_{\mathsf{fake-client}} \leq q_{\mathsf{active}}$.

Game $\mathbf{G}_{10}$: In this game, we focus on $\mathsf{AskHbC}_w\mathsf{C}$ only. We now reject all the authenticators sent by the adversary, impersonating the server: $\Pr[\mathsf{A}_{10}] = 0$. In order to evaluate the distance between the games $\mathbf{G}_{10}$ and $\mathbf{G}_9$, we consider the probability of the event $\mathsf{AskHbC}_w\mathsf{C}$, in which the adversary succeeds in faking the server by sending a valid authenticator to the client before a $\mathsf{Corrupt}$-query.

To evaluate the probability of event $\mathsf{AskHbC}_w\mathsf{C}$, we note that, up to the moment in which a $\mathsf{Corrupt}$-query occurs, no information on the verifier v of a user is revealed to the adversary, despite the fact that it is still used in the computation of $Y_3^\star$. To see that, note that, for any given transcript $(C, S, Y_2\|V\|Y_3^\star)$ in which $Y_3^\star$ was created by an oracle instance, and for each verifier $v = U^{pw} \in \mathbb{G}$, there exists a value $x \in \mathbb{Z}_q$, such that $Y_3^\star = g^x \times v$, which is never revealed to the adversary: this is the first active attack, and the passive attacks are simulated without using v. Moreover, since we have removed collisions on the pre-master secrets, on the MAC keys, and on the MAC values, there is at most one verifier that can lead to a valid authenticator. As a result, the probability that the adversary succeeds in sending a valid authenticator in each of theses sessions is at most $1/N$. Thus, we get $\Pr[\mathsf{AskHbC}_w\mathsf{C}_{10}] \leq q_{\mathsf{fake-server}}/N$.

Game $\mathbf{G}_{11}$: We finally concentrate on the success probability of the adversary in faking the client. What we show in this game is that the adversary cannot eliminate more than one password/verifier in the dictionary by impersonating a client. To do so, we first upper-bound the probability that, for some transcript $(C, S, Y_2\|V\|Y_3^\star)$ in which $(Y_2 = (Y_1, Y_2^{(1)}, Y_2^{(2)}), V)$ was created by server instance, there are two hash queries with (v_0, Z_0, Z_0') and (v_1, Z_1, Z_1') in Λ_{Hash}, and v_0, v_1 possible verifiers (that is that correspond to $U^{\mathcal{G}(C\|S\|U\|pw)}$ for pw that is in the dictionary of size N) such that one has, for $i = 0, 1$,

$$Z_i = \mathsf{CDH}_{Y_1,\mathbb{G}}(Y_3^\star/v_i, Y_2^{(2)}), Z_i' = \mathsf{CDH}_{U,\mathbb{G}}(V, v_i).$$

We denote this event CollH.

In order to upper-bound the probability of event CollH, we consider an auxiliary game in which the simulation of the players changes slightly without affecting the view of the adversary. The goal is to use the adversary to help us compute the computational Diffie-Hellman value of U_1 and U_2, as above. In this simulation, we do as in the Game $\mathbf{G}_5$: we set $U = U_1$. We choose at random one of the $\mathsf{Send}(S, \text{“start”})$-queries being asked to S and we reply with $Y_2 = \{g^{x_1}, U_2, U_2^{x_1}\}$ and $V = U^{x_2'}$, in the hope that this is the session which leads to a collision in the transcript. For all other sessions, Y_2 and V are simulated as usual. Now, let us assume that the event CollH happens, with $v_0 = U^{pw_0}$ and $v_1 = U^{pw_1}$. If our guess for the $\mathsf{Send}(S, \text{“start”})$-query was correct, then we can extract the value $\mathsf{CDH}_{g,\mathbb{G}}(U_1, U_2)$ as $(Z_1/Z_0)^u$, where u is the inverse of $(pw_0 - pw_1)x_1$, by simply guessing the indices of the two hash queries involved in the collision. We note that u is guaranteed to exist since $v_0 \neq v_1$, and thus $pw_0 \neq pw_1$. It follows that $\Pr[\mathsf{CollH}] \leq \mathit{negl}()$.

When the event CollH does not happen, for each transcript $(C, S, Y_2\|V\|Y_3^\star)$ in which Y_2 was created by server instance, there is at most one verifier value v such that the tuple $(Y_2, Y_3^\star, Z)$ is in Λ_{Hash}. Thus, we get $\Pr[\mathsf{AskHbC}_w\mathsf{S}_{11}] \leq q_{\mathsf{fake-client}}/N + \mathit{negl}()$.

Since $\Pr[\mathsf{A}_{11}] = 0$, this concludes the proof. □

5 Conclusion

This paper provides the open source community with strong password-based ciphersuites for TLS that are believed to be free from patent infringements. We have added the verifier-based feature to the 3-party group Diffie-Hellman ciphersuites of Abdalla et al. [ABC+07] to make these ciphersuites better suited to today's practical use in open source libraries.

Acknowledgments

The first author is supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, Mathematical Information and Computing Sciences Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098. Also Office of Science Contract No. DE-AC02-05CH11231.

References

[ABC+06] Michel Abdalla, Emmanuel Bresson, Olivier Chevassut, Bodo Möller, and David Pointcheval. Provably secure password-based authentication in TLS. In *ASIACCS 06*, pages 35–45. ACM Press, 2006.

[ABC+07] Michel Abdalla, Emmanuel Bresson, Olivier Chevassut, Bodo Möller, and David Pointcheval. Strong Password-Based Authentication in TLS Using the Three-Party Group Diffie-Hellman Protocol. In *International Journal of Security and Networks* (IJSN), 2007 (to appear).

[ACP05] Michel Abdalla, Olivier Chevassut, and David Pointcheval. One-time verifier-based encrypted key exchange. In Serge Vaudenay, editor, *PKC 2005*, volume 3386 of *LNCS*, pages 47–64. Springer-Verlag, Berlin, Germany, January 2005.

[BCP02] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Dynamic group Diffie-Hellman key exchange under standard assumptions. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 321–336. Springer-Verlag, Berlin, Germany, April / May 2002.

[BCP03] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. Security proofs for an efficient password-based key exchange. In Sushil Jajodia, Vijayalakshmi Atluri, and Trent Jaeger, editors, *ACM CCS 03*, pages 241–250. ACM Press, October 2003.

[BCP04] Emmanuel Bresson, Olivier Chevassut, and David Pointcheval. New security results on encrypted key exchange. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *PKC 2004*, volume 2947 of *LNCS*, pages 145–158. Springer-Verlag, Berlin, Germany, March 2004.

[BM92] Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *1992 IEEE Symposium on Security and Privacy*, pages 72–84. IEEE Computer Society Press, May 1992.

[BM93] Steven M. Bellovin and Michael Merritt. Augmented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In V. Ashby, editor, *ACM CCS 93*, pages 244–250. ACM Press, November 1993.

[BMP00] Victor Boyko, Philip D. MacKenzie, and Sarvar Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 156–171. Springer-Verlag, Berlin, Germany, May 2000.

[BPR00] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 139–155. Springer-Verlag, Berlin, Germany, May 2000.

[GMR06] Craig Gentry, Philip MacKenzie, and Zulfikar Ramzan. A method for making password-based key exchange resilient to server compromise. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 142–159. Springer-Verlag, Berlin, Germany, August 2006.