Prompted User Retrieval of Secret Entropy: The Passmaze Protocol

Daniel R. L. Brown*

December 15, 2005

Abstract

A prompting protocol permits users to securely retrieve secrets with greater entropy than passwords. The retrieved user secrets can have enough entropy to be used to derive cryptographic keys.

1 Introduction

Users are generally treated as the weakest link in the information security chain. One of users' main contribution to security is low-entropy passwords. Users' long-term keys, if any, are generally stored in a device, encrypted with passwords. The entropy of a user's password, say 20 bits, represents the relative cost for an adversary to extract the key from its password-encrypted form, which would be approximately 2^{20} password-decryptions in this example. Access to the stored password-encrypted user key should therefore be limited.

Passwords may provide adequate security in many applications. However, the low-entropy nature of passwords is not intrinsic to human nature, but rather to the computer user interface. Human memory, including memory of individual secrets, has a capacity far greater than what is needed for a secure cryptographic key. Unfortunately, keyboard entry of passwords (or passphrases) has relative low entropy input rate per character stroke, and as the number of character strokes increases, so does user inconvenience and chance of user error. With proper prompting however, the entropy rate input per character stroke (or mouse click) can be considerably increased. This may be useful in certain applications.

A prompting protocol described in this paper provides a secure and reliable method for users to input high-entropy secrets. Its name, *passmaze*, is derived from *passphrase* and indicates the fact that the user must navigate a maze of challenges and responses. Many systems have been proposed to mitigate the various deficiencies of using passwords (see references section). The passmaze protocol aims to do so by replacing or supplementing passwords with something potentially more secure.

2 The Passmaze Protocol

This section describes the protocol. First, the user interface is described. Second, the underlying cryptographic operations are described.

*Certicom Research

2.1 User Interface

An example of a user interface to the passmaze protocol is shown in Figure 1. A *render* function, computed by the client device, converts cryptographic value, computed by the client and server, into a prompt r_{j-1}. The example interface renders the prompt r_{j-1} as a list of English words from a fixed dictionary. The user input is a choice c_j of one word from the prompt r_{j-1}. The client device converts the user input into a cryptographic value, which is then used to compute the next prompt r_j.

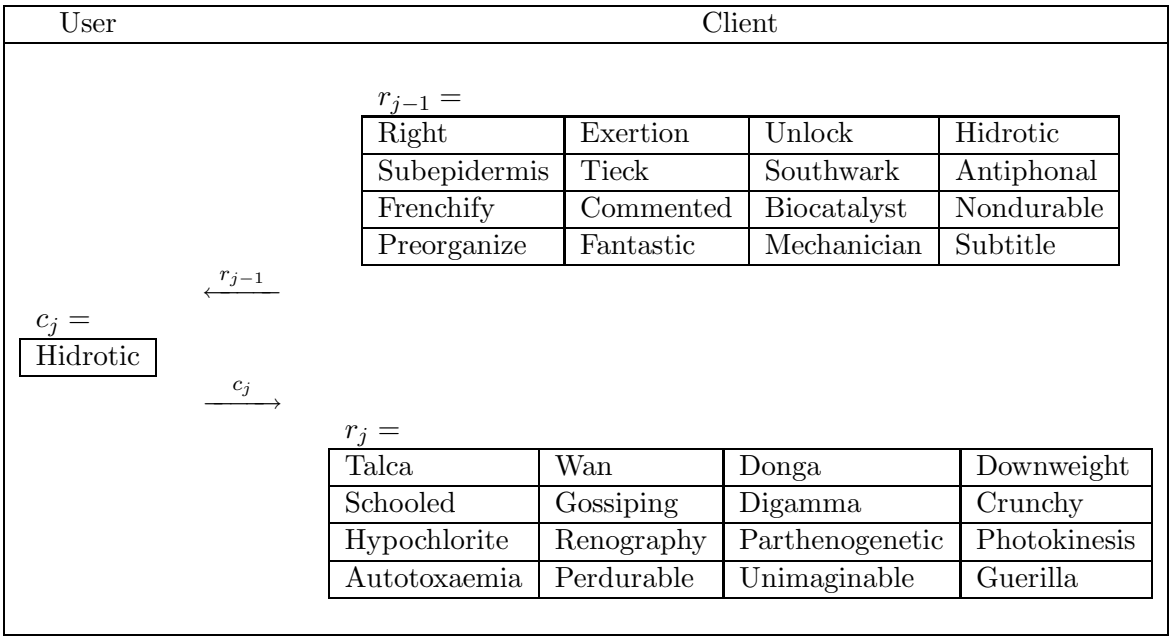

Figure 1: Prompt, Choice and Next Prompt in a User Interface

Other user interfaces are possible in the passmaze protocol:

- Other languages than English.
- Proper names, instead of dictionary words.
- Nonsense words, instead of dictionary words.
- Photographic images from a database, instead of words in the list.
- Client-generated abstract images, instead of words in the list.
- Larger (generated or from a database) images, instead of lists of images, where the user chooses a portion of the image.
- Larger (generated or from a database) images, instead of lists of images, where the user types some short string associated with the string.
- Animated images, such as a maze, that the user navigates in a game-like fashion.

Users can be given the option to customize the interface, without altering the underlying cryptographic protocol. Customized interfaces may enhance resistance to over-the-shoulder attacks,

where the adversary is a person who gets to see parts of the prompt, because the customized interface would be less familiar to the adversary and so less likely to be remembered. User interfaces may optionally provide a back feature that undoes the previous choice so that authorized users can correct occasional accidental mistakes.

For security, the user interface should have certain characteristics. User inputs must be convertible into precise cryptographic values that determine the user secret. Values of user's secret should be assigned with enough entropy. (It is preferable to customize the interface rather than the secret values.) The user interface should not be accessible to adversaries. Each choice of the user contributes some entropy to the total secret. The entropy of the total secret is the sum of the entropy of each choice. The example user interface in Figure 1 has 4 bits of entropy per user input, so would provide 80 bits of entropy if 20 rounds of user input were used.

2.2 Cryptographic Operations

The underlying cryptographic operations of the passmaze protocol can be instantiated with a variety of different cryptographic algorithms. A single round of the protocol, using a client-server network model and an algorithm instantiation with hash functions and elliptic curve cryptography, is illustrated in Figure 2.

User		Client		Server
Select c_j	$\xrightarrow{c_j}$	$C_j = \text{Parse}(c_j)$		
		$H_j = \text{Hash}(C_1 \Vert C_2 \Vert \ldots \Vert C_j)$		
		$G_j = \text{Point}(H_j)$		
		Random k_j		
		$P_j = k_j G_j$	$\xrightarrow{P_j}$	$Q_j = d_j P_j$
		$R_j = k_j^{-1} Q_j = d_j G_j$	$\xleftarrow{Q_j}$	
Recognize r_j	$\xleftarrow{r_j}$	$r_j = \text{Render}(R_j)$		

Figure 2: Round j of the protocol

Note that r_j is a deterministic function of $c_1, \ldots, c_j$. The random value k_j serves mainly to blind communciation between the client and server, so that neither server learns the user selection nor any adversaries intervening between the client and server. Secrecy of d_j mainly helps protects the user from adversaries using an unauthorized client to capture the user selections.

The user must effectively navigate a (deterministic) tree, starting from the root, and ending at a leaf, as illustrated in Figure 3. There are many such paths, and suitably chosen cryptographic functions applied to the path help protect from partial exposures.

After the final round, the client derives the retrieved key from all the user selections. Neither the client nor the server stores any permanent records of any of the individual user choices c_j and client responses r_j. The only permanent record of the choices c_j must be derived in some way from the final retrieved key.

The client does not know whether any user selection c_j is correct or not, so does not provide any explicit feedback on the correctness of each c_j to the user. The client responds to every choice as if correct, so computes the next prompt r_j as a function of c_j, for whatever value of c_j the user entered, whether correct or not.

The authorized user may be able to detect if her choice c_j was correct by observing r_j. The prompt r_j, if correct, should contain her next choice c_{j+1}. If she does not see her next choice c_{j+1} if r_j, then she will conclude that either she accidentally misentered the wrong choice for c_j or that something is amiss in that the client and server are failing to operate correctly. If the user notices

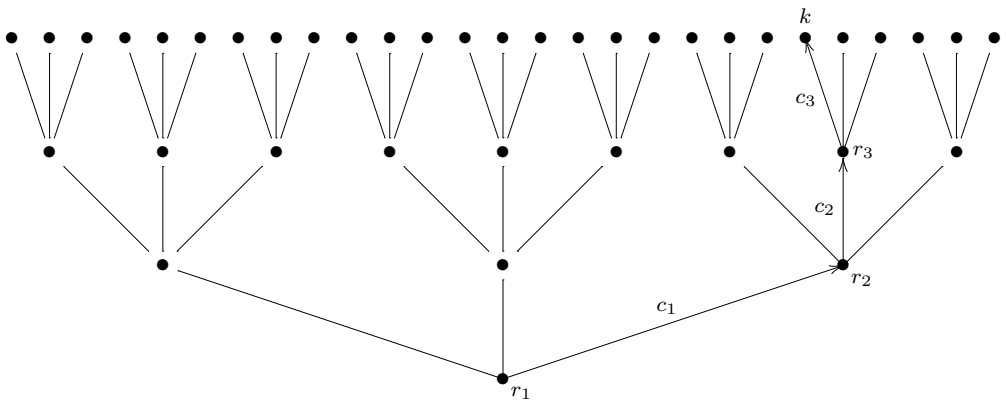

Figure 3: Navigation of the tree

the problem frequently, then she should temporarily cease using the protocol and seek help from a system administrator because of the possibility of an attack.

The adversary as an unauthorized user, however, will not be able to tell whether response r_j is valid, because he does not know c_{j+1}. He has no way to confirm his guess at c_j. This explains why it is crucial for the client not to reveal the correctness of individual values c_j to the user, because if it did, then the adversary could guess values of c_j until the correct one is confirmed. The client should force the user to make all choices $c_1, c_2, \ldots, c_n$ before getting any feedback about correctness. A result is correct only if all the c_j are correct.

Because the client blinds its communication with the server, the user can remain anonymous during the key retrieval process. Anonymity has certain advantages and disadvantages. If anonymity is not desired, then the client can reveal the identity of the user to the server. Furthermore, the server may be able to customize its responses to individual users, which may enhance the security somewhat. Another blinding mechanism and server secret function pair is based on the RSA and its variants. See [19] for details.

The server acts as a raw ECDH oracle, which has some potential security risks [9]. This risk can be avoided by using specially chosen groups, or by the client being implemented as a trusted module that the server can authenticate.

Although the retrieved key can be use for any purpose, probably the most versatile use of the retrieved key is to protect other user keys. The retrieved key can then protect an arbitrary amount of user keying material.

The server and the client can be embedded on the same device. In this approach, the server could be specific to the user, so that the server private key is unique to the user. Then, other devices cannot impersonate the user's device in order to steal the user's secret. This mechanism has considerable security benefit.

The client can include decoys in the prompt. The decoys are random options that are not deterministic functions of the user's previous selections. The decoys serve mainly to prevent over-the-shoulder attacks in which an adversary sees just one option per round. Even if these options are not the user selections, the adversary can test all the choices for c_j to see if r_j has an observed option. The number of log-on attempts adversary would need in this case is about the number of rounds times half the number of options per round. Each observation that is a decoy, however, costs such an adversary somewhat. If the j^{th} seen option is a decoy, then the adversary has to

guess both c_j and c_{j+1} to see if the $(j+1)^{\text{st}}$ option appears in r_{j+1}.

3 Analysis

This section presents some preliminary usability and security analysis of the protocol.

3.1 Usability

Preliminary experiments have shown that sixteen randomly assigned challenges rounds of sixteen choices of random words can be successfully recognized, even with delays of a few months between attempts. Longer periods tend to result in an inability to recognize to identify the correct challenges from the previous responses.

In a Java prototype of the system, the client and server were merged and the responses calculate by inclusion of a secret in the SHA-256 evaluation. The user's first challenge consisted of a conventional user identity dbrown and password (not displayed) certicom, which was not randomly assigned. The dictionary of words used was 2^{16} words extracted from a commonly available freeware spell-checker.

The prototype included, for testing purposes, back and forward buttons to give the test-user a chance to try to challenges until they got a response they recognized. Responses were generally recognized mainly by the presence of absence of the next challenge word, but also somewhat by the presence of other words. Familiarity with the other words in the response could gradually erode the user's ability to select the next challenge, as all words sparked recognition to similar degrees. A mechanism for emphasizing recognition of the user's challenge values was not incorporated in the prototype but might be helpful in an actual system, provided it does not compromise security. The prototype did not incorporate decoys or randomized presentations, so the usability of these features is not yet established.

Figure 4 provides a screenshot of the test prototype implemented in Java. For testing purposes various information is displayed that would not necessarily be secure for display on the client.

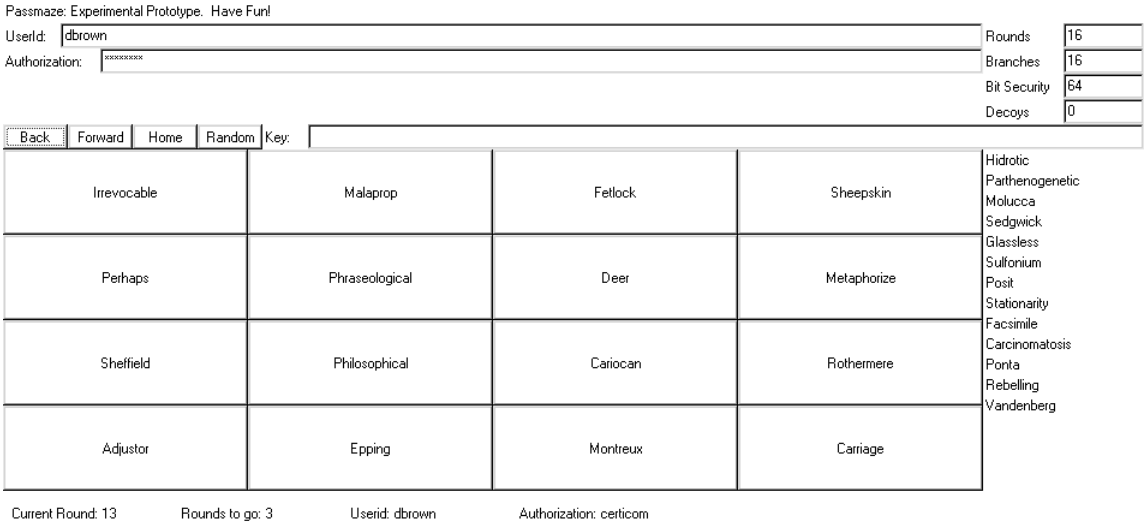

Figure 4: Look and feel of the passmaze protocol

3.2 Security

This sections gives very preliminary security analysis.

3.2.1 Security Against Off-Line Attacks

If strong entropy is enforced, (that is, the user challenges are assigned not user-selected), the server key is not useful for launching an off-line dictionary attack against the user. Once a user's secret is assigned with sufficient entropy, only on-line interactions with the user can feasibly capture the user secret. Conventional passwords generally do not enjoy such a property, unless they are combined with deliberately slow cryptographic functions. When deliberately slow functions are used, they must run in some amount of time convenient for the user, say a second, on some particular device. The entropy of the user's secret, whether a conventional password or a passmaze secret, represents the number of seconds, on a device of the similar power to the user's that an adversary would need to exhaustively extract the user's secret, given a value derived from the user's secret and any other secrets associate with the derived value (such as the server key). Presumably the adversary may use more powerful device than the user, and the may also use several devices in parallel. If a conventional password has about 30 bits of entropy and the adversary has access to one thousand times the computing power of the user, then the adversary may be able to recover the user's secret in about 10 days, thus an off-line attack may be feasible in this case, even when deliberately slow functions are used. A passmze secret with 64 bits of entropy combined with a secure slow function to derive a key, may be infeasible for the same adversary to exhaustively search. Under an accelerated version of Moore's law where computing power doubling every year, each extra bit of entropy in the user secret, adds a year to its lifetime against off-line attacks. A very crude estimate is that, when combined with secure deliberate slow key derivation function, a 64 bit passmaze secret is secure against off-line attacks for about 30 years.

3.2.2 Security Against On-Line Attacks

A rogue client that has live access to a legitimate client can obviously obtain a user's secret path, just be emulating the true client's prompts. Of course, conventional passwords are vulnerable to this kind of attack. To guard against this, countermeasures can be taken to ensure user presence, so that true clients cannot be abused in this manner. If access is not live, so that the rogue client can obtain correct prompts at some later stage, which is more difficult to prevent, then the adversary can obtain the user secret in n interactions with the user, where n is the number of rounds of the protocol. The adversary obtains the first prompt from the true client, which the false client present to the user to capture the first user selection. Then the false client garbles or terminates the session, meanwhile obtaining the next prompt from the true client for the next sesssion. At the next session, the false client displays the second correct prompt and obtains the user's next selection. This continues with a session for each round. Users subjected to such an attack may notice the failed sessions, and should be advised to report failed sessions to an administrator because of the possibility of an attack.

Note that the passmaze and conventional password protocols fit into a general user interface framework for prompted user retrieval of secret entropy. For passwords, the prompt is essentially trivial: a keyboard, some fixed displayed text requesting a password to be entered, and possibly some asterisks displayed for each character typed. The prompt does not change signficantly as the user enters each key stroke, and therefore does not substantially assist the user to retrieve as much entropy as the passmaze prompts do. Indeed, the passmaze protocol could be rendered in a such way that the prompts at each round are completely static, without varying depending on the

previous user selections. This degenerate form of the passmaze protocol may potentially provide the user access to as much secret entropy, but would have less resistance to the attack above in which the adversary presents the user with a false client, because the adversary needs just one interaction with the true client to obtain the static prompts for all rounds.

If each round has b choices (each node of the tree has b branches) and the protocol has h rounds (the tree has height h), then the number of user secrets is b^h. Setting $b \approx h$ roughly minimizes the number of rounds and choices for a given amount of secret entropy. Usability diminishes as b and h get larger, although probably tolerance to large b appears higher. Security against false clients improves with larger h.

Denial-of-service attacks on the server are a serious concern, because the server applies its secret function to any message it receives. An attacker could bombard the server with numerous messages, forcing the server to calculate its secret function repeatedly. One method to partly mitigate these attacks is to require the client to perform a slow calculation that the server can verify quickly, such as solving the discrete logarithm problem in a small group, or factoring a small integer.

Another risk of the protocol is that user makes the challenge selection by rote, without examining the responses. In this case, the user will not authenticate the responses, which could be risky. The user will not be sure the correct key will be retrieved. In addition, if an over-the-shoulder attack or keyboard-bug attack is being launched the pattern of the user's selections may be learnt. To mitigate these two risks, the client can randomized the presentation of the responses and challenges. Thus, although the user's challenges are identical, they must be entered in a different way each time.

3.3 Formal Security Analyses

A formal security analysis for the passmaze protocol may be provided later. The first step to a formal security analysis is a formal security definition. The passmaze protocol does not appear to belong to a class of protocols that has an accepted formal security definition (such as signature schemes), therefore a reasonable security definition must be established. Although it is possible to postulate here a formal definition, it may be prudent to first subject the scheme to creative cryptanalysis, for otherwise a formal definition risks being too narrow.

A second difficulty anticipated with providing a formal security analysis of the protocol is the human element. Therefore, it is likely that the underlying cryptographic operations of the passmaze protocol, rather than the user interface that will be best suited to the formal security analysis. The user interface can also be analyzed, but probably not as formally.

4 Conclusion

The main advantages of the passmaze protocol over conventional password-only protocols are that users are:

1. Able to remember cryptographically strong secrets.
2. Immune to off-line dictionary searches, even from trusted authorities.
3. Able to retrieve a stable signature or decryption key.
4. Not prone to accidentally revealing secrets into existing insecure password systems.
5. Able to directly authenticate server, not just through a certificate installed in the client.

6. Able to be anonymous.

7. Able to customize representation of challenges and responses.

8. Less likely to forget secrets after long periods without use.

This set of security advantages may fit some security application niche in which passwords do not provide adequate security.

As with passwords, the passmaze protocol can be combined with other user authentication technologies, such as cryptographic tokens and biometrics. Systems that require high security user authentication and authorization may benefit by combining the passmaze protocol with tokens and biometrics.

Acknowledgments

Rene Struik, Yongge Wang, Alfred Menezes, Rob Gallant and Scott Vanstone made several valuable suggestions to the protocol and its analysis. Various other Certicom employees helped as test users. David Naccache's comments led to some clarifications in the exposition.

References

[1] A. Adams and M. A. Sasse, *Users are not the enemy: Why users compromise security mechanism and how to take remedial measures*, Communications of the ACM **42** (1999), no. 12, 40–46.

[2] A. Adams, M. A. Sasse, and P. Lunt, *Making passwords secure and usable*, Proceedings of the HCI'97 Conference on People and Computers XII, 1997, pp. 1–19.

[3] M. Bellare and P. Rogaway, *The AuthA protocol for password-based authenticated key exchange*, Tech. report, Contribution to the IEEE P1363 study group, March 2000.

[4] S. M. Bellovin and M. Merritt, *Encrypted key exchange: Password-based protocols secure against dictionary attacks*, Proceedings of the I.E.E.E. Symposium on Research in Security and Privacy (Oakland), May 1992.

[5] ———, *Augmented encrypted key exchange: a password-based protocol secure against dictionary attacks and password file compromise*, Tech. report, AT&T Bell Laboratories, 1994.

[6] M. K. Boyarsky, *Public-key cryptography and password protocols: The multi-user case*, Proceedings of the 6th ACM Conference on Computer and Communications Security (Singapore), November 1999.

[7] V. Boyko, P. MacKenzie, and S. Patel, *Provably secure password authenticated key exchange using Diffie-Hellman*, Advances in Cryptology - EUROCRYPT 2000 (B. Preneel, ed.), May 2000.

[8] S. Brostoff and M. A. Sasse, *Are passfaces more usable than passwords? a field trial investigation*, HCI, September 2000.

[9] D. R. L. Brown and R. P. Gallant, *The static Diffie-Hellman problem*, ePrint 2004/306, IACR, 2004, `http://eprint.iacr.org/`.

[10] P. Buhler, T. Eirich, M. Steiner, and M. Waidner, *Secure password-based cipher suite for TLS*, Proceedings of the Year 2000 Network and Distributed System Security Symposium, February 2000.

[11] J. Callas, L. Donnerhacke, H. Finney, and R. Thayer, *Openpgp message format*, IETF RFC 2440, November 1998.

[12] R. Dhamija, *Usability of computer security: A bibliography*, `http://www.sims.berkeley.edu/~rachna/security_usability.html`, August 2000.

[13] R. Dhamija and A. Perrig, *Deja vu: A user study. using images for authentication*, Proceedings of the 9th USENIX Security Symposium (Denver, Colorado), August 2000.

[14] W. Diffie and M. Hellman, *New directions in cryptography*, IEEE Transactions on Information Theory **IT-22** (1976), 644–654.

[15] W. Diffie, P. van Oorschot, and M. Wiener, *Authentication and authenticated key exchanges*, Designs, Codes and Cryptography **2** (1992), 107–125.

[16] C. Ellison, *Establishing identity without certificate authorities*, Proceedings of the Sixth Annual USENIX Security Symposium (San Jose), July 1996.

[17] ———, *SPKI requirements*, IETF RFC 2692, September 1999.

[18] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen, *SPKI certificate theory*, IETF RFC 2693, September 1999.

[19] W. Ford and B. Kaliski, *Server-assisted generation of a strong secret from a password*, Proceedings of the IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (NIST, Gaithersburg MD), June 2000.

[20] O. Goldreich and Y. Lindell, *Session-key generation using human passwords only*, Tech. Report 057, Cryptology ePrint Archive, 2000.

[21] L. Gong, *Optimal authentication protocols resistant to password guessing attacks*, Proceedings of the 8th IEEE Computer Security Foundations Workshop (County Kerry, Ireland), June 1995, pp. 24–29.

[22] L. Gong, M. Lomas, R. Needham, and J. Saltzer, *Protecting poorly chosen secrets from guessing attacks*, I.E.E.E. Journal on Selected Areas in Communications **11** (1993), no. 5, 648–656.

[23] S. Halevi and H. Krawczyk, *Public-key cryptography and password protocols*, Fifth ACM Conference on Computer and Communication security, ACM, November 1998, pp. 122–131.

[24] ———, *Public-key cryptography and password protocols*, ACM Transactions on Information and System Security (TISSEC) **2** (1999), no. 3, 230–268.

[25] N. Haller, *The S/KEY one-time password system*, Proceedings of the ISOC Symposium on Network and Distributed System Security (San Diego, CA), February 1994.

[26] N. Haller, *The S/KEY one-time password system*, IETF RFC 1760, February 1995.

[27] R. Housley, W. Ford, W. Polk, and D. Solo, *Internet X.509 public key infrastructure certificate and CRL profile*, IETF RFC 2459, January 1999.

[28] IEEE P1363.2, *Standard specifications for public key cryptography: Password-based techniques*, Institute of Electrical and Electronics Engineers, March 2002, Draft.

[29] ITU-T Recommdenation X.509(2000) — ISO/IEC 9594-8:2001, *Information technology — open systems interconnect — the directory: Public-key and attribute certificate frameworks*, ITU-T and ISO/IEC, 2000.

[30] D. Jablon, *Strong password-only authenticated key exchange*, Computer Communication Review, ACM SIGCOMM **26** (1996), no. 5, 5–26.

[31] ———, *Extended password key exchange protocols immune to dictionary attacks*, Proceedings of the Sixth Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET-ICE '97) (Cambridge, MA), IEEE Computer Society, June 1997.

[32] ———, *Password authentication using multiple servers*, Topics in Cryptology – CT-RSA 2001, Lecture Notes in Computer Science, Springer-Verlag, April 2001, pp. 344–360.

[33] I. Jermyn, A. Mayer, F. Monrose, M. K. Reiter, and A. D. Rubin, *The design and analysis of graphical passwords*, Proceedings of the 8th USENIX Security Symposium (Washington, DC), December 1999.

[34] Jonathan Katz, Rafail Ostrovsky, and Moti Yung, *Efficient password-authenticated key exchange using human-memorable passwords*, Advances in Cryptology - EUROCRYPT 2001 (Innsbruck, Austria) (Birgit Pfitzmann, ed.), Lecture Notes in Computer Science, vol. 2045, Springer-Verlag, May 2001, pp. 475–494.

[35] Seungjoo Kim, Byungchun Kim, Sungjun Park, and Sung-Ming Yen, *Comments on password-based private key download protocol of ndss'99*, Electronics Letters **35** (1999), no. 22, 1937–1938.

[36] D. Kirovski, *Remarks*, Microsoft Press Release, November 2002, Silicon Valley Speaker Series.

[37] T. Kwon, *Authentication and key agreement via memorable password*, NDSS 2001 Symposium Conference Proceedings, February 2001.

[38] T. Kwon, M. Kang, S. Jung, and J. Song, *An improvement of the password-based authentication protocol (k1p) on security against replay attacks*, IEICE Transactions on Communications **E82-B** (1999), no. 7, 991–997.

[39] T. Kwon and J. Song, *Efficient and secure password-based authentication protocols against guessing attacks*, Computer Communications **21** (1998), no. 9, 853–861.

[40] ———, *Secure agreement scheme for g^{xy} via password authentication*, Electronics Letters **35** (1999), no. 11, 892–893.

[41] ———, *Authentication and key agreement via memorable password*, Tech. Report 026, Cryptology ePrint Archive, August 2000.

[42] ———, *A study on the generalized key agreement and password authentication protocol*, IEICE Transactions on Communications **E83-B** (2000), no. 9, 2044–2050.

[43] L. Lamport, *Password authentication with insecure communication*, Communications of the ACM (1981).

[44] Y. K. Lee and J. K. Lee, *EC-SRP protocol: Elliptic curve secure remote password protocol*, Tech. report, Korea Information Security Agency, December 1998.

[45] ———, *EC-SRP protocol: Elliptic curve secure remote password protocol*, Korea Institute of Information Security & Cryptology **9** (1999), no. 1, 85–102.

[46] S. Lucks, *Open key exchange: How to defeat dictionary attacks without encrypting public keys*, The Security Protocol Workshop '97 (Ecole Normale Superieure), April 1997.

[47] P. MacKenzie, *More efficient password-authenticated key exchange*, Topics in Cryptology — CT-RSA 2001, Lecture Notes in Computer Science, vol. 2020, Springer-Verlag, April 2001, pp. 361–377.

[48] ———, *On the security of the SPEKE password-authenticated key exchange protocol*, Tech. Report 057, Cryptology ePrint Archive, 2001.

[49] T. Matsumoto, *Human-computer cryptography: an attempt*, Proceedings of the 3rd ACM conference on Computer and communications security, ACM Press, 1996, pp. 68–75.

[50] T. Matsumoto and H. Imai, *Human identification through insecure channel*, Advances in Cryptology — EUROCRYPT '91 (D. W. Davies, ed.), Lecture Notes in Computer Science, vol. 547, Springer-Verlag, 1991, pp. 409–421.

[51] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *The handbook of applied cryptography*, CRC Press, 1997.

[52] National Institute of Standards and Technology, *FIPS 181: Automated password generator*, October 1993.

[53] ———, *FIPS 197: Advanced encryption standard (AES)*, November 2001.

[54] S. Patel, *Information leakage in encrypted key exchange*, Proceedings of DIMACS Workshop on Network Threats, vol. 38, December 1996, pp. 33–40.

[55] ———, *Number theoretic attacks on secure password schemes*, Proceedings 1997 IEEE Symposium on Security and Privacy (Oakland, California), May 1997, pp. 236–247.

[56] R. Perlman and C. Kaufman, *Secure password-based protocol for downloading a private key*, Proceedings of the 1999 Network and Distributed System Security, February 1999.

[57] A. Perrig and D. Song, *Hash visualization: A new technique to improve real-world security*, Proceedings of the 1999 International Workshop on Cryptographic Techniques and E-Commerce, 1999.

[58] S. Porter, *A password extension for improved human factors*, Advances in Cryptology (Santa Barbara, California) (A. Gersho, ed.), 1981, p. 81.

[59] N. Provos and D. Mazières, *A future-adaptable password scheme*, Proceedings of the FREENIX Track: 1999 USENIX Annual Technical Conference (Monterey, California), June 1999.

[60] A. G. Reinhold, *The diceware passphrase home page*, `http://world.std.com/~reinhold/diceware.html`, August 1995.

[61] M. Roe, B. Christianson, and D. Wheeler, *Secure sessions from weak secrets*, Tech. report, University of Cambridge and University of Hertfordshire, 1998.

[62] S.A.B.R.O. Net, *Vulnerabilities in the S/KEY one time password system*, `http://home.indy.net/~sabronet/secure/skeyflaws.html`.

[63] S. L. Smith, *Authenticating users by word association random access*, Proceedings of the Human Factors Society 31st Annual Meeting, 1987, pp. 135–138.

[64] M. Steiner, G. Tsudik, and M. Waidner, *Refinement and extension of encrypted key exchange*, Operating Systems Review **29** (1995), no. 3, 22–30.

[65] O. Tirosh, *Mnemonic*, `http://www.tothink.com/mnenomic/`.

[66] C.H. Wang, T. Hwang, and J.-J. Tsai, *On the Matsumoto and Imai's human identification scheme*, Advances in Cryptology — EUROCRYPT '95 (L. C. Guillou and J.-J. Quisquater, eds.), Lecture Notes in Computer Science, vol. 921, Springer-Verlag, 1995, pp. 382—392.

[67] Y. Wang, *IEEE P1363.2 submission D2001-06-21*, `http://grouper.ieee.org/groups/1363/passwdPK/index.html`, June 2001, Describes elliptic curve version of SRP.

[68] C. J. C. H. Watkins, *A method of automatic verification of personal identity*, WO 91/09383, June 1991.

[69] T. Wu, *The secure remote password protocol*, Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium (San Diego), March 1998, pp. 97–111.

[70] ———, *The SRP authentication and key exchange system*, IETF RFC 2945, 2000.

[71] ———, *Telnet authentication: SRP*, IETF RFC 2944, 2000.

[72] P. R. Zimmermann, *PGPfone: Pretty good privacy phone owner's manual*, `http://web.mit.edu/network/pgpfone/manual/index.html#PGP000062`, January 1996, Appendix E: Biometric Word Lists.