Secure Network Authentication with Password Identification

Philip MacKenzie* Ram Swaminathan†

July 30, 1999

Submission to IEEE P1363a

Abstract

A password authentication protocol called SNAPI is proposed for inclusion in the P1363a document. SNAPI provides mutual authentication between a client and server based solely on a password, and does not require the client to store any other information (except the code that runs the protocol). SNAPI is the first protocol of this type that is *provably secure* against active adversaries (i.e., adversaries that can not only eavesdrop on communication, but also impersonate parties and replay messages), and in particular, does not reveal any information to active adversaries that would allow an off-line dictionary attack on the password. Security is proven in the random-oracle model and is based on the security of RSA. SNAPI also provides for key exchange (as secure as Diffie-Hellman), allowing a secure session to be initiated. A variant, SNAPI-X, is also proposed, in which the server stores a one-way function of the password, and does not allow an adversary who compromises the server to impersonate a client (without actually running a dictionary attack on the password file).

The protocols described in this contribution are from the paper, *Secure Network Authentication with Password Identification* [MS].

*Information Sciences Center, Bell Laboratories, Lucent Technologies, 600 Mountain Avenue, Murray Hill, NJ 07974–0636, philmac@research.bell-labs.com.

†Information Sciences Center, Bell Laboratories, Lucent Technologies, 600 Mountain Avenue, Murray Hill, NJ 07974–0636, swaram@research.bell-labs.com.

1 Introduction

A user authentication protocol is a protocol in which a system may verify the identity of a user in order to allow access to some resource. Due to its simplicity, the most common form of user authentication is password-based, that is, a user must be able to provide a memorized password in order to be authenticated. Password-based user authentication has the problem that users often choose weak passwords in order to be able to memorize them. These weak passwords (such as a birthday, or common English word) are potentially susceptible to *dictionary attacks*. Dictionary attacks are brute force attacks on passwords that are performed by testing a large number of likely passwords (say, all the words in an English dictionary and all possible dates from this century) against some publicly available information about the desired password. The concern over dictionary attacks is why most UNIX password files are now *shadowed*, and thus unviewable by normal users. See Wu [Wu99] for some recent demonstrations of the effectiveness of dictionary attacks.

When user authentication must be performed over an untrusted network, additional problems arise due to the fact that the communication may be overheard, or even altered, by an attacker. In general, one can not assume that there is a secure channel between the user's system (the client) and the authenticating system (the server). Thus, one must deal with eavesdropping attacks, replay attacks, and also spoofing (i.e., impersonation or man-in-the-middle attacks).

In light of these problems, the idea of providing strong security for network authentication using passwords that could potentially be weakly chosen seems to be contradictory. However, there have been several password-only user authentication (and key exchange) protocols proposed in the past few years that are designed to be secure, in particular with respect to off-line dictionary attacks. Jablon [Jab, Jab96] gives good descriptions of these, including an "ancestor tree" for the different protocols. Notable protocols are Encrypted Key Exchange (EKE) by Bellovin and Merritt [BM92], which is the first such protocol in the area, Augmented-EKE (A-EKE) [BM93], the secret public-key methods in [GLNS93, Gon95], Modified-EKE (M-EKE) [STW95], Simple Password EKE (SPEKE) and Diffie-Hellman EKE (DH-EKE) [Jab96], B-SPEKE [Jab97], Secure Remote Password Protocol (SRP) [Wu98], and Open Key Exchange (OKE) [Luc97]. No formal security proofs were provided for any of these protocols. In fact, certain attacks on some of these protocols were shown in [Pat97].

In this proposal, we present a protocol called SNAPI which provides mutual authentication and key exchange over a network between a user/client and a server. (SNAPI is derived from, and is very similar to, the OKE protocol. See Section 4 for details.) The client does not store any information specific to the server, and simply requires the user to input a password. The protocol is proven secure against active attackers, meaning that except for some negligible probability, the best attack on the system is simply to guess passwords and attempt authentication with a valid client or server. Obviously, this is the best security possible for a password-based protocol. We also present an extension of SNAPI called SNAPI-X which provides added (proven) security in the case of server compromise.

2 Description of the cryptographic technique

SNAPI and SNAPI-X are protocols that provide mutual authentication based on a password, and additionally provide key exchange. It is assumed that the server stores an RSA pair [RSA78] where the public key is (N, e) and the private key is (N, d), N is the product of two large primes P and Q, $\gcd(e, \phi(N)) = 1$, and $ed \equiv 1 \bmod \phi(N)$, where $\phi(N) = (P-1)(Q-1)$.

2.1 Security parameters

The system assumes two security parameters k and ℓ have been set, where k is the security parameter for the basic hash functions, and $\ell > k$ is the security parameter for RSA.

2.2 Hash functions

The SNAPI and SNAPI-X protocols use two short hash functions h and h', as well as two long hash functions H and H' (H' is used only in SNAPI-X.) For our security proofs, these hash functions are assumed to behave like random oracles [BR93b], and below we give constructions that seem to allow this using known hash functions. The constructions are based on those given in [BR93b].

The hash functions should take as input an arbitrary number of bits and output a certain fixed number of bits. We assume that the short hash functions output k bits, and currently we recommend SHA-1 or RIPEMD-160, with $k = 160$ for both. Let $hash$ denote the underlying hash function. To differentiate h and h', we recommend $h(x) = hash(00|x|x|00)$ and $h'(x) = hash(01|x|x|01)$, where "|" denotes concatenation, and "00" and "01" denote length 2 bit strings.

We assume the long hash functions output $\eta \geq \ell + k$ bits. For these hash functions we use different length 2 bit strings "10" and "11", and also an index corresponding to the output positions of the bits generated by each $hash$ application. We recommend

$$H(x) = hash(10|0|x|x|0|10) \,|\, hash(10|1|x|x|1|10) \,|\, hash(10|2|x|x|2|10) \,|\, \cdots \,|\, hash(10|i|x|x|i|10) \text{ and}$$

$$H'(x) = hash(11|0|x|x|0|11) \,|\, hash(11|1|x|x|1|11) \,|\, hash(11|2|x|x|2|11) \,|\, \cdots \,|\, hash(11|i|x|x|i|11),$$

where $i = \eta/k$ and η is a multiple of k, and the index is represented as an 8 bit number.

2.3 RSA restrictions

Let key generator GE define a family of RSA functions to be $(e, d, N) \leftarrow GE(1^\ell)$ such that $N = PQ$, where P and Q are prime numbers. Then, the public key is the pair (e, N) where $\gcd(e, \phi(N)) = 1$ and the order of the group $\phi(N) = (P-1) \cdot (Q-1)$. The encryption function $E : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ is defined by $E(x) \equiv x^e \bmod N$ and the decryption function $D : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ is $D(x) \equiv x^d \bmod N$, where the secret exponent d is chosen such that $ed \equiv 1 \bmod \phi(N)$.

The choice of P, Q, and e is generally left to the implementation, although it is recommended that P and Q be random large primes with about the same bit length (about $\ell/2$ for security

parameter ℓ) [IEE98]. There is no recommendation on the value e except that it obeys the constraint $\gcd(e, \phi(N)) = 1$. For efficiency e is often chosen to be a small prime and with a small number of ones in its binary representation, such as 3, 17, or 65537. However, this will not be the case in SNAPI and SNAPI-X.

SNAPI is parameterized by restrictions on the possible values of the public key. These restrictions must be efficiently testable by someone who only knows the public key. The restrctions are designed to force N to be large and $\gcd(e, \phi(N)) = 1$. The following four are possible candidates for such restrictions:

1. $N \in [2^{\ell-2}, 2^{\ell}]$, $e \in [2^{\ell}, 2^{\ell+1}]$, e prime.
2. $N \in [2^{\ell-2}, 2^{\ell}]$, $e \in [2^{\ell/2}, 2^{\ell/2+1}]$, e prime, $(N \bmod e) \not| N$.
3. $N \in [2^{\ell-2}, 2^{\ell}]$, e the smallest prime greater than 2^{ℓ}.
4. $N \in [2^{\ell-2}, 2^{\ell}]$, e the smallest prime greater than $2^{\ell/2}$, $(N \bmod e) \not| N$.

We require these choices of RSA parameters for our security proof, and these choices are generally accepted as providing strong security for RSA [IEE98]. Below we demonstrate SNAPI and SNAPI-X protocols for Case 1 only.

2.4 SNAPI protocol

In SNAPI, Alice (the server) and Bob (the client) agree on a common password $\pi \in P$ and Alice chooses an RSA public-key/private-key by generating $(e, d, N) \leftarrow \mathrm{GE}(1^{\ell})$, where $N = PQ$, $P, Q \in [2^{\ell/2-1}, 2^{\ell/2}]$ are randomly chosen primes, $e \in (2^{\ell}, 2^{\ell+1}]$ and e is prime.

Let g be a *generator* of a cyclic group Ω of size ω superpolynomial in k in which the Diffie-Hellman problem is hard; we use this group for obscuring the password (later in SNAPI-X), for Diffie-Hellman key exchange, and as a source of nonces.

Let $\mathcal{A}$ denote the identity of Alice, and let $\mathcal{B}$ denote the identity of Bob.

The protocol is shown in Figure 1.

Comments:

- The public key (N, e) used by Alice may be used for multiple sessions, i.e., Alice does not need to generate a new public key for each session.
- After Bob computes p, Bob checks if $\gcd(p, N) \neq 1$ or $p \geq 2^{\eta} - (2^{\eta} \bmod N)$. The second part of the test was used for the proof in order to remove bias in the value $p \bmod N$, but removing it does not seem to affect security, since the condition should only occur with negligible probability. (A similar comment holds when Bob does a similar test later.) Note that the first part of the test is *very* important, in that removing it may cause Bob to reveal information about the password to an adversary impersonating Alice.

- After Alice computes p', she checks if $\gcd(p', N) \neq 1$ or $p' \geq 2^\eta - (2^\eta \bmod N)$. Given that Alice, herself, is not corrupted, removing this test does not seem to affect security, since the condition should only occur with negligible probability.

2.5 SNAPI-X protocol

Password-based authentication protocols such as SRP are also designed to withstand server compromise, i.e., an attacker who steals the password file and any other information (such as cryptographic keys) from the server can not use that information directly to gain access to the system. Here we present an extended SNAPI which is also secure against server compromise. We call this extended protocol SNAPI-X (SNAPI-eXtended).

Again, let $\mathcal{A}$ denote the identity of Alice, and let $\mathcal{B}$ denote the identity of Bob.

In the SNAPI-X protocol, we assume there is an initialization in which the client, Bob, chooses a password $\pi \in P$ and a salt value, computes $x = H'(\mathcal{A}|\mathcal{B}|\pi|\text{salt})$ and sends the server, Alice, $X = g^x$ and the salt value, both of which are stored by Alice. As in SNAPI, Alice chooses RSA parameters (N, e, d). After the initialization Bob only needs to remember π.

The protocol is shown in Figure 2.

Comments:

- The comments for SNAPI apply to SNAPI-X also.
- Assuming ω is prime and Ω is the multiplicative subgroup of Z_ρ^* generated by g, where $\rho = d\omega + 1$ for some d where $\gcd(d, \omega) = 1$, then to test if $y \in \Omega$, one should test if $y^\omega \equiv 1$. In this case it may be possible for Bob to combine the computation of y^x and y^ω for efficiency in his last step [Yao76].

3 Claimed attributes and advantages

SNAPI and SNAPI-X have the following advantages compared to the other network user authentication protocols.

- SNAPI and SNAPI-X are password-only protocols, i.e., the client end does not have to store any server specific information except the password (which will not actually be stored on the client machine, but simply remembered by the user). In this aspect, SNAPI and SNAPI-X are easier to manage than protocols like `ssh` [SSH99] and IPSEC [HC98] which do require the client to hold server certificates for secure mutual authentication.
- The efficiency of the SNAPI and SNAPI-X protocols is close to the efficiency of SPEKE, SRP and `ssh` in terms of the number of steps and the number of public-key cryptographic operations.

Bob:
1. Send username to Alice.

Alice:
1. Choose $\alpha \in_R \mathbb{Z}_\omega$.
2. Set $m = g^\alpha$.
3. Send (m, N, e) to Bob.

Bob:
1. If $N \notin [2^{\ell-2}, 2^\ell]$, $e \notin (2^\ell, 2^{\ell+1}]$, or e is not prime, then reject,
2. Else,
   (a) Choose $\beta \in_R \mathbb{Z}_\omega$ and $a \in_R \mathbb{Z}_N^*$.
   (b) Set $\mu = g^\beta$.
   (c) Compute $p = H(N|e|m|\mu|\mathcal{A}|\mathcal{B}|\pi)$.
   (d) If $\gcd(p, N) \neq 1$ or $p \geq 2^\eta - (2^\eta \bmod N)$, then set $q = a$,
   (e) Else, set $q \equiv pa^e \bmod N$.
   (f) Send (μ, q) to Alice.

Alice:
1. If $\gcd(q, N) \neq 1$, then reject,
2. Else,
   (a) Compute $p' = H(N|e|m|\mu|\mathcal{A}|\mathcal{B}|\pi)$.
   (b) If $\gcd(p', N) \neq 1$ or $p' \geq 2^\eta - (2^\eta \bmod N)$, then reject,
   (c) Else,
      i. Set $a' \equiv (q/p')^d \bmod N$.
      ii. Set $r = h(a')$.
   (d) Send r to Bob.

Bob:
1. If $\gcd(p, N) = 1$, $p < 2^\eta - (2^\eta \bmod N)$ and $r = h(a)$, then
   (a) Accept,
   (b) Compute session key m^β.
   (c) Send $t = h'(a)$ to Alice.
2. Else, reject.

Alice:
1. If $t = h'(a')$, then
   (a) Accept.
   (b) Compute session key μ^α.
2. Else, reject.

Figure 1: SNAPI Protocol

Bob:
1. Send username to Alice.

Alice:
1. Choose $\alpha \in_R \mathbb{Z}_\omega$.
2. Retrieve X and salt from password file for username.
3. Set $m = g^\alpha$.
4. Send (m, N, e, salt) to Bob.

Bob:
1. If $N \notin [2^{\ell-2}, 2^\ell]$, $e \notin (2^\ell, 2^{\ell+1}]$, or e is not prime, then reject,
2. Else,
   (a) Set $x = H'(\mathcal{A}|\mathcal{B}|\pi|\text{salt})$.
   (b) Choose $a \in_R \mathbb{Z}_N^*$ and $\beta \in_R \mathbb{Z}_\omega$.
   (c) Set $\mu = g^\beta$.
   (d) Compute $p = H(E|m|\mu|g^x)$.
   (e) If $\gcd(p, N) \neq 1$ or $p \geq 2^\eta - (2^\eta \bmod N)$, then set $q = a$,
   (f) Else, set $q \equiv pa^e \bmod N$.
   (g) Send (μ, q) to Alice.

Alice:
1. If $\gcd(q, N) \neq 1$, then reject,
2. Else,
   (a) Compute $p' = H(E|m|\mu|X)$.
   (b) If $\gcd(p', N) \neq 1$ or $p' \geq 2^\eta - (2^\eta \bmod N)$, then reject,
   (c) Else,
      i. Set $a' \equiv (q/p')^d \bmod N$.
      ii. Set $r = h(a')$.
   (d) Choose $\gamma \in_R \mathbb{Z}_\omega$.
   (e) Set $y = g^\gamma$.
   (f) Send (r, y) to Bob.

Bob:
1. If $\gcd(p, N) = 1$, $p < 2^\eta - (2^\eta \bmod N)$, $y \in \Omega$ and $r = h(a)$, then
   (a) Accept,
   (b) Compute session key m^β.
   (c) Send $t = h'(a|y^x)$ to Alice.
2. Else, reject.

Alice:
1. If $t = h'(a'|X^\gamma)$, then
   (a) Accept.
   (b) Compute session key μ^α.
2. Else, reject.

Figure 2: SNAPI-X Protocol

- In contrast to the other known password-only authentication and key exchange protocols like SPEKE and SRP, SNAPI and SNAPI-X are provably secure (in the random oracle model).
- Like most of the other known password-only protocols, SNAPI and SNAPI-X are straightforward to implement.
- SNAPI-X is secure against server compromise while SNAPI is not. However, SNAPI is more efficient than SNAPI-X.

4 Security assessment and considerations

SNAPI and SNAPI-X have been shown to be secure mutual authentication and key exchange protocols in the Bellare-Rogaway communication model, which assumes that an adversary completely controls the communication channel. For a precise definition and proof of security, we refer to [MS].

As discussed above, SNAPI is derived from, and is very similar to, the OKE protocol of Lucks [Luc97]. The OKE protocol is designed for an "ideal encryption scheme" which does not correspond to any known "real-world" encryption scheme. In fact, to work with RSA, Lucks designed a much more complex "protected-OKE" protocol. However, protected-OKE seems to be insecure [Pat99]. The SNAPI protocol is based on the original OKE protocol, but is enhanced to provide partial testability of RSA keys by the client, and then mechanisms to provide security for the client even for incorrectly formed RSA keys that pass the partial tests.

One might ask, "What does it mean to be a secure mutual authentication protocol based on passwords?" Formally, if we assume that a password is chosen from a dictionary of size d, and the adversary performs v impersonation attacks (such as attempting to login as the user), then the adversary should have only a $(v/d) + \epsilon$ chance of succeeding, where ϵ is negligible. SNAPI and SNAPI-X are both secure mutual authentication protocols, meaning that the adversary has $(v/d) + \epsilon$ chance of succeeding with v impersonations.

SNAPI-X is also secure against server compromise. This means that even if an adversary compromises the server and obtains the password file and any cryptographic keys, the adversary still cannot impersonate a client to that server without performing a dictionary attack on the password file. (Of course, there is no way to prevent this attack once the adversary has obtained the password file.) More formally, we show that the probability that the adversary successfully impersonates a client is $(v/d) + \epsilon$, where v is the number of passwords tested in the dictionary attack, and ϵ is negligible.

To be precise about security against server compromise, in our model we assume that once an adversary compromises the server, it does not impersonate the server to the client anymore, and simply tries to impersonate the client. This would model a situation in which the server somehow accessed the files on the server, but could not corrupt any routers between the client and server in order to impersonate the server to the client. We do not know how an adversary that impersonates the server could obtain the plaintext password, but, unfortunately, we also do not know how to prove an adversary could not obtain the plaintext password through impersonating the server.

The security proofs for SNAPI and SNAPI-X hold in the random-oracle model [BR93a], i.e., it is assumed that all the hash functions behave like random hash functions. While a protocol having a security proof in the random-oracle model is certainly less desirable than a protocol having a proof in the standard model (using standard cryptographic assumptions) [CGH98], it is certainly preferable over a protocol which lacks any proof. Other protocols proven secure in the random-oracle model include Optimal Asymmetric Encryption Padding [BR94] (used in PKCS #1 v. 2 [Not99]) and Provably Secure Signatures [BR96].

5 Known limitations and disadvantages

SNAPI and SNAPI-X are slightly slower than some other protocols, such as `ssh` and SRP, because in addition to the Diffie-Hellman computations, there is an RSA encryption by the client and RSA decryption by the server. On current processors this may add about 600 milliseconds to the protocol. For user authentication, this does seem reasonable, and the combined cryptographic computations on the server and client still require less than 2 seconds.

SNAPI is not designed to be secure against server compromise. SNAPI-X is designed to be secure against server compromise, but is less efficient than SNAPI.

The security proofs for SNAPI and SNAPI-X hold only in the random-oracle model, and thus there is the possibility of an attacker using some currently unknown feature of the hash functions to break the protocol. However, to date no one has found such an (effective) attack on SHA-1 or RIPEMD-160.

There is a password authentication protocol that seems to be proven secure under a complexity-theoretic assumption about the intractability of noisy polynomial reconstruction, but this assumption is non-standard, and the protocol is inefficient [NP99].

Export restrictions and other patent issues may be involved since the protocol uses RSA encryptions and decryptions.

6 Intellectual property issues

Lucent Technologies has filed a patent application on the protocols described in this contribution. Lucent will license any resulting patent in a reasonable and non-discriminatory fashion in compliance with its intellectual property rules and regulations. A letter to this effect will be provided.

Acknowledgements

We thank Daniel Bleichenbacher for helpful and stimulating discussions, and for showing us how our protocol can remain secure with a shorter RSA public exponent e (Restriction cases 2 and 4 in Section 2.3). We also thank Victor Boyko for his thorough reading of the paper and helpful

comments.

References

[BM92] S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 72–84, 1992.

[BM93] S. M. Bellovin and M. Merritt. Augumented encrypted key exchange: A password-based protocol secure against dictionary attacks and password file compromise. In CCS'93 [CCS93], pages 244–250.

[BR93a] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In CCS'93 [CCS93].

[BR93b] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In *Advances in Cryptology—CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer-Verlag, 22–26 August 1993.

[BR94] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *Advances in Cryptology—EUROCRYPT 94*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer-Verlag, 1995, 9–12 May 1994.

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures—how to sign with RSA and Rabin. In *Advances in Cryptology—EUROCRYPT 96*, volume 1070 of *Lecture Notes in Computer Science*, pages 399–416. Springer-Verlag, 12–16 May 1996.

[CCS93] *Proceedings of the First Annual Conference on Computer and Communications Security*, 1993.

[CGH98] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, pages 209–218, Dallas, Texas, 23–26 May 1998.

[GLNS93] L. Gong, T. M. A. Lomas, R. M. Needham, and J. H. Saltzer. Protecting poorly chosen secrets from guessing attacks. *IEEE Journal on Selected Areas in Communications*, 11(5):648–656, June 1993.

[Gon95] L. Gong. Optimal authentication protocols resistant to password guessing attacks. In *Proceedings of the 8th IEEE Computer Security Foundations Workshop*, pages 24–29, 1995.

[HC98] D. Harkins and D. Carrel. The internet key exchange. *RFC 2409*, 1998.

[IEE98] IEEE P1363 Annex D/Editorial Contribution 1c: Standard specifications for public-key cryptography, June 1998.

[Jab] D. Jablon. Integrity sciences web site. http://www.IntegritySciences.com.

[Jab96] D. Jablon. Strong password-only authenticated key exchange. *ACM Computer Communication Review, ACM SIGCOMM*, 26(5):5–20, 1996.

[Jab97] D. Jablon. Extended password key exchange protocols immune to dictionary attack. In *WETICE'97 Workshop on Enterprise Security*, 1997.

[Luc97] Stephan Lucks. Open key exchange: How to defeat dictionary attacks without encrypting public keys. In *Proceedings of the Workshop on Security Protocols*, 1997.

[MS] P. MacKenzie and R. Swaminathan. Secure network authentication with password information. manuscript.

[Not99] RSA Laboratories Technical Note. PKCS #1, version 2, RSA encryption standard. `http://www.rsa.com/rsalabs/pubs/PKCS/`, 1999.

[NP99] M. Naor and B. Pinkas. Oblivious transfer and polynomial evaluation. In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, pages 245–254, Atlanta, Georgia, 1–4 May 1999.

[Pat97] S. Patel. Number theoretic attacks on secure password schemes. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 236–247, 1997.

[Pat99] S. Patel, 1999. Personal communication.

[RSA78] R. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signature and public key cryptosystems. *Communications of the ACM*, 21:120–126, 1978.

[SSH99] SSH. Web site, 1999. www.ssh.fi.

[STW95] M. Steiner, G. Tsudik, and M. Waidner. Refinement and extension of encrypted key exchange. *ACM Operating System Review*, 29:22–30, 1995.

[Wu98] T. Wu. The secure remote password protocol. In *Proceedings of the 1998 Internet Society Network and Distributed System Security Symposium*, pages 97–111, 1998.

[Wu99] T. Wu. A real world analysis of kerberos password security. In *Proceedings of the 1999 Internet Society Network and Distributed System Security Symposium*, 1999.

[Yao76] Andrew Chi-Chih Yao. On the evaluation of powers. *SIAM Journal on Computing*, 5(1):100–103, March 1976.