CLAMBAKE: Certified, Long-term Anonymous, Multi-user Building Access with zero-Knowledge Encryption

Michael Kielstra and Beatrice Nash

January 12, 2022

Abstract

We present a new scheme for establishing both secure and anonymous building access. By combining cryptographic techniques including broadcast encryption and garbled circuits, we develop a new model for anonymous communication in which the knowledge-restriction protocol replaces the multi-receiver, honest majority requirement of other secure multi-party schemes. The resulting advantages include a scheme that is robust and secure to unauthorized and dishonest parties, anonymous to the authorized user, adaptable to online changes, and polynomial time in the number of users. We provide justification for our chosen approach, as well as other possible future directions for further understanding and developing such schemes.

1 Introduction

In this study, we discuss a protocol for use in constructing secure, anonymous building access. The model of anonymous communication is broadly defined as one in which the keycard, one of a number of allowed users, wishes to prove access to a building, which knows the secrets of all keycards, while retaining its anonymity. We use "building-keycard" communication instead of more standard user names, e.g. Alice and Bob, in reference to a specific use case of a more general communication setup in order to help with intuition.

In this model, the only information the building needs to complete the protocol is proof that the keycard's access is as claimed (or lack thereof, which results in a denial of access). For the scheme to be both secure and user-anonymous, the building must be able to positively or negatively certify whether the keycard is one of the legitimate users with near certainty, but the building cannot differentiate it from any other user with its same credentials. We operate under the "honest-but-curious" assumption: following successful completion of the protocol, the building may attempt to access identifying information about the

card to which it does not have access, yet it will follow the protocol as an honest party. On the other hand, the keycard may act dishonestly in that it could try to prove credentials it does not actually possess to gain unauthorized access.[1]

To construct a secure scheme based on these requirements, we use broadcast encryption, which allows for secure, certificateless communication between two parties, combined with a proof of zero-knowledge on the part of the building constructed using Yao's garbled circuit method. The result is a scheme that is secure, efficient, and anonymous. Our paper is organized as follows: first, we provide background information necessary for our key techniques – broadcast encryption and garbled circuits – then, we present the model of encryption, followed by the protocol itself and a high-level overview of how to prove its security. Details of implementation are not the focus but are addressed in the concluding discussion.

2 Background and related work

2.1 Broadcast Encryption

The first technique we need is a way to communicate securely over multi-receiver channels and revoke access to those channels without interacting with the receivers. In 1991, Berkovits [Ber91] introduced broadcast encryption, billed as a way to securely transmit keys for symmetric encryption to only a targeted subset of users. In 1994, Fiat and Naor developed the methods and theory further, including introducing the first broadcast encryption schemes to be robust with repeated use [FN94]. Broadcast encryption is used familiarly in cable television: every set-top box gets all the channels, but to decrypt the relevant ones requires a key provided only to subscribers.

A broadcast encryption scheme comprises three randomized algorithms [BGW05]:

1. **SETUP**(n), which takes as input the number of potential receivers n and generates a public key and n private keys,
2. **ENC**(V, PK), which takes V, the subset of receivers that are valid, as well as PK, the public key, and generates a header Hdr and a message-encryption key K, which can then be used to to encrypt messages using a symmetric algorithm.
3. **DEC**(V, i, d_i, Hdr, PK), which takes V, a user ID i, the private key d_i, the header, and the public key, and outputs the message-encryption key K which was generated along with that header.

[1] There are also other uses for this sort of scheme. A library might buy access to a newspaper, but the library's subscribers might not want to reveal which articles they, as opposed to other subscribers, are reading. Politicians weary of government espionage, or social-justice advocates worried about racist security policies, might demand that border guards allow citizens to enter the country without checking those citizens' exact names or other identifying information.

The scheme is then completed with a standard symmetric CCA-secure scheme to encrypt the actual messages. Note: since we only wish to exchange random messages, we could ignore this and just treat K as our message. However, for the purpose of increasing the generality of our scheme, we will assume that the building and the card use K to pass messages to each other with a CCA-secure encryption method.

There exist a number of different methods for broadcast encryption that satisfy the needs of our scheme. The one we choose, introduced by Boneh, Gentry, and Waters [BGW05], is useful for our purposes due to the short sizes of its public keys and messages. Many broadcast encryption schemes, such as [LPQ12], have either message sizes or public key sizes, or both, growing linearly with the number n of authorized users. [BGW05] gives a broadcast encryption scheme for which the length of public keys and messages grows only with the square root of the number of authorized users and where the length of the private key is constant. The exact details of this scheme, while interesting, are better left to a tangential discussion – see appendix B for implementation details, along with a proof of CPA security.

We might naively assume that broadcast encryption solves all our problems. The building simply chooses a random number, encrypts it, and sends it to the card. The card sends back the decrypted number, and, if it matches the number originally chosen, the card is accepted. Anonymous authentication is achieved. However, this only works if the building is honest. A dishonest building might follow the access protocol honestly but, upon termination, compute on the transcript of the interaction to gain information about the identity of the keycard (beyond that which is essential for the protocol to complete). Such a building might send not an encrypted message, but instead just garbage, and emulate each keycard in turn to see how each responds when trying to “decrypt" the garbage. This attack would allow the building to distinguish between keycards in time linear in the total number of potential keycards.

To avoid this, the keycard first demands that the building prove that the message it sent would decrypt to the same thing by the other private keys belonging to valid keycards. Of course, we do not wish to surrender those private keys to the keycard, which would then make the protocol vulnerable to later attacks. Another idea would be to have the building provide encrypted versions of the keys under a secret key unbeknownst to the keycard. The building would then use fully homomorphic encryption to perform key-switching, allowing the keycard to compute (i.e. verify decryption to the same message) using the encrypted keys without revealing them. The former option is insecure and the latter is highly inefficient, so we take a different route.

2.2 Yao’s Garbled Circuits

We would like the building to be able to demonstrate that it is in possession of many private keys, all of which decrypt the given message to the same thing,

without actually revealing the keys. This is a classic use case for zero-knowledge proofs, or ZKPs, which were first conceived by Goldwasser, Micali and Rackoff in their 1985 paper introducing the topic [GMR85]. The groundbreaking work also introduced interactive proof systems more generally, an innovation for which they – along with Babai and Moran for their 1988 paper introducing Arthur-Merlin prover systems [BM88] – won the first-awarded Gödel Prize in 1993 [AT]. Interactive proof systems are today used in a wide variety of contexts, from proving fundamental results in quantum complexity to essential applications including maintaining the security of blockchains [GMR89].

Some of the most useful methods of constructing knowledge-restricted proof systems are based on Yao's garbled circuits, which were not formalized by Yao but whose origin can be traced to his work [Yao82; Yao86]. We use garbled circuits as a practical zero-knowledge proof system to construct a framework for secure computation between two honest-but-curious parties, which we can then extend to a framework for secure communication in the malicious setting. A simple example is the scenario in which Bob and Alice, in respective possession of bits $m_B, m_A \in \{0, 1\}$, want to compute the function $\texttt{argmax}(m_A, m_B)$, and they want to do so securely, that is, without compromising information about their bitstring. Parties are referred to as honest-but-curious (sometimes referred to in other settings as semi-honest) because, while they play the game according to the rules, they may later refer to the transcript of their interactions to glean information about the other party's message.

In order for such a scheme to be secure, one wants it to be *oblivious*, *private*, and *authentic* [BHR12]. Obliviousness is the condition that an adversary, from a garbled circuit H and garbled input X, cannot efficiently compute anything about the actual input x. Privacy is the condition that an adversary, in possession of private key d to decrypt the output of H on garbled input X, cannot efficiently learn anything additional about input x beyond that revealed by $f(x, \cdot)$. Finally, authenticity is the guarantee that, without private key d, the adversary cannot efficiently decrypt outputs of the garbled circuit.

The formal definitions follow:

Definition 1 (Obliviousness)**.** A garbled circuit protocol $\mathcal{G}(f, x) = (H, X)$ is **oblivious** if for every efficient adversary $\mathcal{A}$, there exists an efficiently computable function Sim such that,

$$\textsc{Sim}(f) \overset{d}{\approx} \texttt{View}_{\mathcal{A}}(H, X),$$

where $\overset{d}{\approx}$ indicates approximate equivalence of probability distributions to within negligible difference ϵ.

Definition 2 (Privacy)**.** A garbled circuit protocol $\mathcal{G}(f, x) = (H, X)$ is **private** if for every efficient adversary $\mathcal{A}$ with access to private key d, there exists an

Garbled Circuit

1. Alice, in possession of bit m_A, and Bob, in possession of bit m_B, want to compute $f(m_A, m_B)$ without betraying non-essential information to the other about their respective message.
2. Alice randomly generates two sets of keys, (k_A^0, k_A^1) and (k_B^0, k_B^1), and a permutation $\pi : \{0,1\}^2 \to \{0,1\}^2$.
3. Alice computes $f(i,j)$ for all pairs $(i,j) \in \{0,1\}^2$ and outputs a circuit H_f such that:

$$H_f(i,j) = E_{k_A^\alpha}(E_{k_B^\beta}(f(\alpha,\beta)))$$

   for $(\alpha,\beta) = \pi(i,j)$. Alice sends to Bob H_f and $k_A^{m_A}$.
4. Alice and Bob perform OT such that Bob learns $k_B^{m_B}$ without Alice learning m_B or Bob learning $k_B^{1-m_B}$.
5. Bob decrypts the output table of H_f. His pair of keys, $(k_A^{m_A}, k_B^{m_B})$, can decrypt only the correct output message. Bob outputs $f(m_A, m_B)$.

Protocol 1: Garbled circuit construction.

efficiently computable function SIM such that,

$$\text{SIM}(f,d) \overset{d}{\approx} \texttt{View}_{\mathcal{A}}(H,X,d).$$

Definition 3 (Authenticity)**.** Consider the game $\mathcal{G}^{\mathcal{A}}_{H,X}(d, 1^\lambda)$ which simulates efficient adversary $\mathcal{A}$ on H, X and outputs 1 if $\mathcal{A}$ successfully decrypts $E_e(m_b)$ for $m_b \neq f(x)$, $b \in \{0,1\}$. A garbled circuit protocol $\mathcal{G}(f,x) = (H,X)$ is **authentic** if, for any $\mathcal{A}$,

$$\Pr[\mathcal{G}^{\mathcal{A}}_{H,X}(1^\lambda) \text{ outputs } 1] \leq 1/2 + \epsilon(\lambda).$$

The definitions provided for privacy and obliviousness are equivalent to those presented in [BHR12]. Our definition for authenticity is slightly modified in that we require only that efficient adversaries cannot correctly decrypt the circuit outputs given H, X, not that an error will be necessarily be produced when they attempt to do so. As discussed in subsequent sections, this requirement is sufficient for security as defined in our model.

Before discussing how these goals can be achieved using garbled circuits, one must first define *oblivious transfer* (OT). Consider the specific scenario in which one honest-but-curious party, Alice, has two messages, (m_0, m_1), and wants to communicate m_b with another honest-but-curious party, Bob. However, only Bob knows the value of b. In order for Alice to communicate m_b to Bob without

learning b or sharing m_{1-b}, she and Bob will perform OT. OT is essential to make garbled circuits work, however, the details of its implementation are not particularly relevant to our scheme. For further details, we refer the reader to appendix A.

Protocol 1 only demonstrates a garbled circuit on a function of two bits that outputs a single bit. To garble a more complicated circuit H_g that computes a function $g : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^*$, one need not construct the entire 2^{2n}-size input/output mapping for g. Instead, the individual gates of H_g are garbled according to the protocol above (for a truth table representation of a garbled gate, see Table 1).

The resulting garbled circuit has size $4 \cdot |H_g|$, where $|H_g|$ is the number of gates in the un-garbled circuit computing g. Alice garbles the circuit, then sends Bob the resulting circuit and the input wires corresponding to her input. After receiving through OT the keys, Bob decrypts the output table and sends to Alice the final result: $g(m_A, m_B)$. This garbling procedure fulfills the three requirements of security above: privacy, obliviousness, and authenticity.

Garbled circuits are useful in a wide range of scenarios. In our scheme, they are used as part of the zero-knowledge protocol between building and keycard to prove the anonymity of the keycard. In order to do so, however, we must introduce an additional degree of security beyond that of the standard garbled circuit.

2.2.1 Cut-and-choose

The garbled circuit protocol described in the prior section is designed for security against honest-but-curious parties only. If player Alice were to output a garbled circuit that did not compute the agreed upon function, Bob, in the basic protocol, would not find cause to abort. As will become clear in Section 3, this attack is particularly problematic to our model of communication. Hence, we require a strengthened version of garbled circuits for use in our proposed scheme.

A simple and effective solution, as developed in [LP15] and explained in [Sny14],

inputs		output
0	0	$f(0,0)$
0	1	$f(0,1)$
1	0	$f(1,0)$
1	1	$f(1,1)$

$\Longrightarrow$

inputs $\pi(i,j) = (\alpha_{i,j}, \beta_{i,j})$		output
$\alpha_{0,0}$	$\beta_{0,0}$	$E_{k_A^{\alpha_{0,0}}}(E_{k_B^{\beta_{0,0}}}(f(\alpha_{0,0}, \beta_{0,0})))$
$\alpha_{0,1}$	$\beta_{0,1}$	$E_{k_A^{\alpha_{0,1}}}(E_{k_B^{\beta_{0,1}}}(f(\alpha_{0,1}, \beta_{0,1})))$
$\alpha_{1,0}$	$\beta_{1,0}$	$E_{k_A^{\alpha_{1,0}}}(E_{k_B^{\beta_{1,0}}}(f(\alpha_{1,0}, \beta_{1,0})))$
$\alpha_{1,1}$	$\beta_{1,1}$	$E_{k_A^{\alpha_{1,1}}}(E_{k_B^{\beta_{1,1}}}(f(\alpha_{1,1}, \beta_{1,1})))$

Table 1: Garbled gate computing $f : \{0,1\}^2 \to \{0,1\}$.

is the *cut and choose* method. Cut and choose involves the garbler (say, B), i.e. the party charged with constructing the garbled circuit, providing the verifying party (C) with not one garbled circuit but with m different ones. After receiving the garbled circuits, C will then select $m/2$ of them and demand that they be un-garbled, at which point C will check that the majority of these circuits are correct. If not, C will abort the protocol. Once this check is completed successfully, C provides B with the wires corresponding to its inputs in the $m/2$ remaining garbled circuits. B and C then compute the remaining circuits and take a majority vote on the result.

In this case, B would only succeed convincing C to accept a malicious circuit if:

1. It corrupts more than $m/4$ and less than $3m/4$ of the circuits, and
2. $< m/4$ of those circuits are in the $m/2$ circuits that C chose to check, and
3. $\geq m/4$ of those circuits are in the $m/2$ circuits chosen for evaluation.

The probability that this occurs is negligible in m. In fact, if we make the slight change of ungarbling 3/5 of the circuits instead of 1/2 of them, then the adversary succeeds with probability at most $(1/2)^{0.32m}$, which is optimal [SS11].

3 Methods

We set up the problem as follows. First, we assume the building is a slightly different version of honest-but-curious, which we call semi-dishonest:

Definition 4 (Semi-dishonest building)**.** A semi-dishonest building B acts honest-but-curiously in all steps of Protocol 2 except steps 2, 3, and 4 (i.e. those which are concerned with verifying C's anonymity, not with C's access credentials), in which B may act dishonestly in order to glean more information about C than it is privy to.

Based on the assumption that the building is a semi-dishonest party, and the card is potentially dishonest, we want to ensure the following:

Closed-door Security The building never accepts a card without valid, explicitly-granted access to the building;

Open-door Security The building always accepts an honest card with valid access to the building;

Revocation The building can, at any time between authentication operations, make arbitrary changes to the list of cards with access;

Impersonation Security The card is not able to determine the private key of any card with valid access to the building with non-negligible advantage using any efficient strategy;

Card Anonymity The building is not able to distinguish an honest card from any other with its same access with non-negligible advantage using any

CLAMBAKE

1. B chooses a random message m and encrypts it using Boneh-Getry-Waters broadcast encryption on $(e_0, ..., e_j)$ for all $(e_i, d_i) \in V$, i.e., the public keys of all the cards with access to B. It sends the encrypted message c to C.
2. B computes n garbled circuits $\{H_f\}$, each of which compute the function $f_c(m, (d_0, ..., d_j))$. $f_c(m, (d_0, ..., d_j))$ computes $D_{d_i}(c)$ for all $i \in [0, j]$ and outputs 1 if every key decrypts the ciphertext to m and 0 otherwise.
3. C randomly selects $p = 3n/5$ garbled circuits $\{H_f^0, ..., H_f^p\}$ to test and sends the set to B.
4. B ungarbles the circuits $\in \{H_f^0, ..., H_f^p\}$ and sends the result to C.
5. C examines $H_f^i \forall i \in [0, p]$ and verifies that the circuits compute the agreed-upon function. If any circuits are incorrect, C aborts with an error. If all circuits are correct, C computes $m' = D_d(c)$, chooses $H_f' \notin \{H_f^0, ..., H_f^p\}$, and computes the result of the garbled circuit on B's provided input. If $f(m', (d_0, ..., d_j)) = 1$, C sends m' to B. Else, C aborts with an error.
6. B accepts C if $m' = m$.

Protocol 2: CLAMBAKE encryption scheme.

efficient strategy.

To achieve the goals outlined above, we also require that our building has access to a secure system (e.g. a trusted third-party database) of the private and public keys of all keycards and whether they are granted access. Through this system, our protocol allows for keycard access to change dynamically and on an individual basis.

Protocol 2 requires a public-key broadcast encryption scheme, the encryption and decryption functions of which are given by (E, D). We denote the building as B, the keycard as C, their public and private key pairings as (e, d), and the set of cards with valid access as V.

4 Anonymity and security

We define two security games in Protocol 3: the building's, in which the card wins if it convinces the building it has access different from its own, and the card's, in which the building wins if it gains identifying information about the card. In both games, aborting the protocol is considered not allowing entry. While security against these games provide only an intuition of a security proof,

Building's security game

1. $C \notin V$ selects messages (m_0, m_1) and sends them to B.
2. B randomly selects $b \in \{0, 1\}$ sends to C: $E_{e'}(m_b)$ and $H_f(m, (d'))$ for some card $C' \neq C, C' \in V$ with keys (e', d').
3. B and C authenticate. C may act honestly or it may not; B acts honestly. C may repeat this step a polynomial number of times.
4. C chooses $b' \in \{0, 1\}$ and wins if $b = b'$.

Card's security game

1. B chooses two sets $c_0, c_1 \in V$ of public/private keys. Assign profile c_b to C, where $b = 0$ with probability $1/2$ and 1 otherwise.
2. B and C authenticate. B is semi-dishonest; C acts honestly. B may repeat this step a polynomial number of times.
3. B outputs $b' \in \{0, 1\}$ and wins if $b = b'$.

Protocol 3: Security games.

we hope that they will provide the reader with a grasp of the types of scenarios this scheme is designed to protect against.

We provide an intuition for proving that the building and keycard win their respective games with negligible advantage over $1/2$.

4.1 Intuition for security and zero-knowledge

Note that these security games do not preclude the card or building from becoming corrupted and purposefully causing the scheme to abort, or from the building going rouge and verifying whichever cards it wants, regardless of access. We are concerned only with robustness and security for the case in which the building acts in an semi-dishonest fashion. (We imagine that any user of this system would find proper access control more important than any particular cryptographic integrity or lack thereof.) The card, on the other hand, can act as maliciously as it desires. Under these assumptions, we demonstrate security under the definition given in Section 3 through use of the security games shown above.

4.1.1 Building's security game

By CPA security of Boneh-Getry-Waters encryption, without knowing the private key, the card is unable to distinguish between two encrypted messages. Therefore,

any card strategy which can win the building's security game with some advantage over $1/2$ must be able to extract some information on d'. By the oblivious property of our garbled circuit construction, C cannot extract any information about the function inputs, d' or m, from H, X.

To enter the building illegitimately, the card would have to be able to win this game: if it can decrypt a random message sent by the building, then it can decrypt one of two chosen messages. Therefore, the building's access remains secure to cards without legitimate access.

4.1.2 Card's security game

Now, consider the card's security game. By the privacy of our chosen garbled circuit construction, B cannot learn anything about C beyond that revealed by $f(x)$. Given the cut-and-choose method ensures that $f(x)$ has been calculated in the agreed-upon fashion, if $f(x)$ returns 1, there must indeed exist $j-1$ other keys that decrypt the ciphertext to the same message as C. Since C_0, C_1 both decrypt the ciphertext to the same message, B cannot distinguish between the two based on this communication with any improvement over random. B only learns that C is *one of* the cards with valid access to B, which is the only information it needs to provide C with access.

5 Discussion and Implementation Details

This is, of course, not a protocol that can be implemented by a card incapable of making decisions, or even a card incapable of transmitting complicated data. Luckily, protocols exist which could be adapted for this. Standard Chip and PIN technology, as described in [EMV], gives cards the ability to send arbitrary data to card readers in response to various verbs which the readers can use. This would of course have to be extended with further verbs and possibly more flexibility to work with our system, but in theory it could be done.

The more challenging issue is that of processing power. While we have tried to ensure that all the most computationally-intensive parts of the protocol – generating garbled circuits, choosing and encrypting random messages – are on the building's side, the card still has to do significant work. In particular, if we use the cut-and-choose method of securing our circuits, the card has to process and verify a large number of circuits in order to satisfy itself that it is not being cheated.

One possible solution to this would be allowing the cardholder to set security parameters to their own liking. Those who just wanted to get into the building could even disable the garbled circuit portion of the protocol, and reveal some information about themselves which would allow buildings to cut down on public key size, while the ultra-paranoid could demand full public keys and dozens or even hundreds of cut-and-choose circuits, with the wait time that that would entail. This kind of philosophy is not so strange: at least one of the authors

of this paper uses a password manager, which causes him pain and hassle in configuration but makes up for it in improved password security. We might even imagine increased anonymity being allowed, via cards configured to ask for it, to only those for whom it is truly necessary or meaningful. Perhaps a celebrity, perfectly happy to broadcast his own location at all times, would prefer for the media not to know which of his family members were visiting him when.

We could argue, in conclusion, that we have failed in our implied ultimate goal: to remove identity from authentication in a way that could be adopted by identity card manufacturers and users everywhere. The increased time that this protocol would take is simply too much for, say, a university with thousands of students tapping in and out of any given door on any given day. However, as a proof of existence of such privacy-respecting card protocols, this work is still important. Privacy helplessness, in which users decide that the game is so rigged against them that they might as well just give up all privacy from the start [Cho21], is a serious problem on social media, which might suggest the prevalence of a similar attitude in the world of authentication. We show that a better world is possible. Building it with these protocols alone might be difficult, but it is not inconceivable.

References

[Yao82] Andrew C. Yao. "Protocols for secure computations". In: *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*. 1982, pp. 160–164. DOI: 10.1109/SFCS.1982.38.

[GMR85] S Goldwasser, S Micali, and C Rackoff. "The knowledge complexity of interactive proof-systems". In: *Proceedings of the seventeenth annual ACM symposium on Theory of computing - STOC '85*. ACM Press, 1985, pp. 291–304. ISBN: 9780897911511. DOI: 10.1145/22145.22178. URL: http://portal.acm.org/citation.cfm?doid=22145.22178.

[Yao86] Andrew Chi-Chih Yao. "How to generate and exchange secrets". In: *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*. 1986, pp. 162–167. DOI: 10.1109/SFCS.1986.25.

[BM88] László Babai and Shlomo Moran. "Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes". In: *Journal of Computer and System Sciences* 36.2 (Apr. 1988), pp. 254–276. ISSN: 00220000. DOI: 10.1016/0022-0000(88)90028-1. URL: https://linkinghub.elsevier.com/retrieve/pii/0022000088900281.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. "The Knowledge Complexity of Interactive Proof Systems". In: *SIAM Journal on Computing* 18.1 (1989), pp. 186–208. DOI: 10.1137/0218012. eprint: https://doi.org/10.1137/0218012. URL: https://doi.org/10.1137/0218012.

[Ber91] Shimshon Berkovits. "How To Broadcast A Secret". In: *Advances in Cryptology — EUROCRYPT '91*. Ed. by Donald W. Davies. Vol. 547. Springer Berlin Heidelberg, 1991, pp. 535–541. ISBN: 9783540546207.

DOI: 10.1007/3-540-46416-6_50. URL: http://link.springer.com/10.1007/3-540-46416-6_50.

[FN94] Amos Fiat and Moni Naor. "Broadcast Encryption". In: *Advances in Cryptology — CRYPTO' 93*. Ed. by Douglas R. Stinson. Vol. 773. Springer Berlin Heidelberg, 1994, pp. 480–491. ISBN: 9783540577669. DOI: 10.1007/3-540-48329-2_40. URL: http://link.springer.com/10.1007/3-540-48329-2_40.

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys". In: *Advances in Cryptology – CRYPTO 2005*. Ed. by Victor Shoup. Vol. 3621. Springer Berlin Heidelberg, 2005, pp. 258–275. ISBN: 9783540281146. DOI: 10.1007/11535218_16. URL: http://link.springer.com/10.1007/11535218_16.

[SS11] Abhi Shelat and Chih-Hao Shen. "Two-Output Secure Computation with Malicious Adversaries". In: *Proceedings of the 30th Annual International Conference on Theory and Applications of Cryptographic Techniques: Advances in Cryptology*. EUROCRYPT'11. Tallinn, Estonia: Springer-Verlag, 2011, pp. 386–405. ISBN: 9783642204647.

[BHR12] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. "Foundations of Garbled Circuits". In: *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. CCS '12. Raleigh, North Carolina, USA: Association for Computing Machinery, 2012, pp. 784–796. ISBN: 9781450316514. DOI: 10.1145/2382196.2382279. URL: https://doi.org/10.1145/2382196.2382279.

[LPQ12] Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. "Anonymous Broadcast Encryption: Adaptive Security and Efficient Constructions in the Standard Model". In: *Public Key Cryptography – PKC 2012*. Ed. by Marc Fischlin, Johannes Buchmann, and Mark Manulis. Vol. 7293. Springer Berlin Heidelberg, 2012, pp. 206–224. ISBN: 9783642300561. DOI: 10.1007/978-3-642-30057-8_13. URL: http://link.springer.com/10.1007/978-3-642-30057-8_13.

[Sny14] Peter Snyder. "Yao ' s Garbled Circuits : Recent Directions and Implementations". In: 2014.

[LP15] Yehuda Lindell and Benny Pinkas. "An Efficient Protocol for Secure Two-Party Computation in the Presence of Malicious Adversaries". In: *Journal of Cryptology* 28.2 (Apr. 2015), pp. 312–350. ISSN: 0933-2790, 1432-1378. DOI: 10.1007/s00145-014-9177-x. URL: http://link.springer.com/10.1007/s00145-014-9177-x.

[Cho21] Hichang Cho. "Privacy helplessness on social media: its constituents, antecedents and consequences". eng. In: *Internet research* ahead-of-print.ahead-of-print (2021). ISSN: 1066-2243.

[AT] ACM Special Interest Group on Algorithms and Computation Theory. *ACM SIGACT - Gödel Prize*. URL: https://www.sigact.org/prizes/g%C3%B6del.html.

[EMV] EMVCo. *Application Independent ICC to Terminal Interface Requirements*. 4.3. Vol. 1. Integrated Circuit Card Specifications for

Oblivious Transfer (Honest-but-Curious)

1. Let $P1$ have two messages m_0, m_1.
2. $P2$ chooses some $i \in \{0, 1\}$ and generates k_1, k_2, where k_i is the public key corresponding to a private key e and k_{1-i} is random. $P2$ transmits k_1, k_2 to $P1$ and keeps i secret.
3. $P1$ encrypts m_j with k_j and transmits the result.
4. $P2$ decrypts $E_{k_i}(m_i)$ with e. It cannot decrypt $E_{k_{1-i}}(m_{1-i})$, since k_{1-i} has no associated private key.

Protocol 4: Oblivious transfer (honest-but-curious).

Oblivious Transfer (Malicious)

1. As before, $P1$ has m_0, m_1 and $P2$ chooses i.
2. $P1$ and $P2$ agree on some q and some g which generates $\mathbb{Z}_q^*$.
3. $P1$ chooses $C \in \mathbb{Z}_q^*$ such that $P2$ cannot find the discrete logarithm of C with respect to g efficiently.
4. $P2$ generates $0 \leq x_i \leq q - 2$ and sets $k_i = g^{x_i}$, $k_{1-i} = Ck_i^{-1}$ and transmits k_0, k_1 to $P1$.
5. $P1$ verifies that $k_0 k_1 = C$.

Protocol 5: Oblivious transfer (malicious).

Payment Systems. URL: https://www.emvco.com/wp-content/uploads/documents/EMV_v4.3_Book_1_ICC_to_Terminal_Interface_2012060705394541.pdf.

A Details of Oblivious Transfer

The aim of oblivious transfer, or OT, is for one party to be able to transmit two values so that another party may learn at most one of them, without revealing which one. As discussed in [Sny14], protocol 4 works in the honest-but-curious setting.

The attack in the malicious setting is obvious: $P2$ can create and transmit two private keys. To defend against this, $P1$ can put constraints on the keys that $P2$ can choose, such as in protocol 5.

At this point, $P1$ and $P2$ have agreed on two numbers, only one of which can be a public key in a cryptosystem based on the hardness of the discrete logarithm problem. They may then choose their favorite such cryptosystem and use it to

encrypt m_0 and m_1 and decrypt only m_i. (If $P2$ were somehow able to decrypt both m_0 and m_1, then it would be able to find the discrete logarithm of C.)

B Details of Boneh-Gentry-Waters Broadcast Encryption

B.1 Bilinear Diffie-Hellman Exponentiation

Let $\mathbb{G}$ and $\mathbb{G}_1$ be groups of prime order p, with g a generator of $\mathbb{G}$. Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$ be a bilinear map. The ℓ-BDHE (bilinear Diffie-Hellman Exponent) problem is to start with a vector

$$(h, g, g^{\alpha}, g^{(\alpha^2)}, \ldots, g^{(\alpha^{2\ell})}) \in \mathbb{G}^{2\ell+1}$$

and output $e(g,h)^{(\alpha^{\ell+1})}$.

This can also be written as a decision problem. For a given g and α, define g_i as shorthand for $g^{(\alpha^i)}$. Then define $y_{g,\alpha,\ell} = (g_1, \ldots, g_\ell, g_{\ell+1}\hat{}, g_{\ell+2}, \ldots, g_{2\ell})$. The ℓ-BDHE assumption is then that the distribution $(g, h, y_{g,\alpha,\ell}, e(h, g_{\ell+1}))$, where g and h are randomly-chosen elements of $\mathbb{G}$ and α is randomly chosen from $\mathbb{Z}_{|\mathbb{G}|}$, is computationally indistinguishable from $(g, h, y_{g,\alpha,\ell}, T)$, where $T \in \mathbb{G}_1$ is also chosen randomly. We will base our cryptographic system on this assumption.

B.2 Construction

Assume we have n users and pick A and B such that $AB = n$. Our encryption will essentially split into A parallel schemes, each of which can broadcast to B users.

1. **SETUP**: Pick a random generator $g \in \mathbb{G}$ and a random $\alpha \in \mathbb{Z}_{|\mathbb{G}|}$. Pick also random $\gamma_i \in \mathbb{Z}_{|\mathbb{G}|}$ for $i \in \{1, \ldots, A\}$ and set $g_b = g^{\alpha^b}$, $v_i = g^{\gamma_i}$. Then the public key is

   $$PK = (g, g_1, \ldots, g_B, g_{B+2}, \ldots, g_{2B}, v_1, \ldots, v_A).$$

   To find the private key for the j-th user, let $i = (a-1)B + b$ for $1 \leq b \leq B$. Then $d_i = g_b^{\gamma_a} = v_a^{(\alpha^b)}$.

2. **ENC**(S, PK): Define $\hat{S}_\ell = S \cap \{(\ell-1)B+1, \ldots, \ell B\}$ and $S_\ell = \{1, \ldots, |\hat{S}_\ell|\}$. $\hat{S}_\ell$ contains all the users which should be communicated with using the ℓ-th scheme, and S_ℓ indexes them from 1 to however many there are. Pick a random $t \in \mathbb{Z}_{|\mathbb{G}|}$ and set $K = e(g_{B+1}, g)^t$. Then

   $$Hdr = \left(g^t, v_1 \cdot \prod_{j \in S_1} g_{B+1-j}, \ldots, v_A \cdot \prod_{j \in S_A} g_{B+1-j} \right).$$

3. **DEC**(S, i, d_i, Hdr, PK): Write $Hdr = (C_0, \ldots, C_A)$ and $i = (a-1)B + b$ for $1 \leq b \leq B$. Then

$$K = e(g_b, C_a) \cdot e\left(d_i \cdot \prod_{j \in S_a \setminus \{b\}} g_{B+1-j+b}, C_0\right)^{-1}.$$

We may verify that this works correctly by simply expanding all the definitions. This is tedious, so we refer the reader to the original paper.

This works for any pair A, B that multiply to n. We choose $A = B = \sqrt{n}$, as this gives the shortest public keys and messages.

B.3 Proof of CPA security

Suppose there exists an adversary $\mathcal{A}$ that wins the CPA game against this encryption scheme, for some pair A, B, with advantage ϵ over $1/2$. We will construct an adversary that can distinguish between the two distributions given in the B-BDHE problem with some non-negligible advantage over $1/2$.

There are two twists to the standard CPA game when working with multi-user encryption. The first is that adversaries begin by listing the users they wish to attack. They then receive the private keys for all other users. All encryption requests are then fulfilled as being encrypted for those users alone to read.

The second twist is that the job of the adversary is to distinguish the defender's chosen key from randomness, given a header, not to distinguish between two chosen messages. Any algorithm that could solve the CPA game for actual encrypted messages using those keys could also be used to distinguish the keys from randomness, simply by running it and seeing if it were successful.

We sample a tuple $(g, h, y_{g,\alpha,B}, Z)$ from our distribution, where Z could be either $e_(g_{B+1}, h)$ or just a random group element. Then, we run $\mathcal{A}$, starting by receiving the set S of users it wishes to attack. Define $\hat{S}_\ell$ and S_ℓ as before, and choose random $u_i \in \mathbb{Z}_{|\mathbb{G}|}$. Set

$$v_i = g^{u_i} \cdot \prod_{j \in S_i} g^{-1}_{B+1-j}$$

and give $\mathcal{A}$ the public key

$$PK = (g, g_1, \ldots, g_B, g_{B+2}, \ldots, g_{2B}, v_1, \ldots, v_A.$$

We calculate public keys

$$d_i = g_b^{u_i} \cdot \prod_{j \in S_a} g^{-1}_{B+1-j+b},$$

which we can verify are equal to $v_a^{(\alpha^b)}$, and hand over the relevant ones to $\mathcal{A}$.

In the standard CPA game, this would be the time for us to begin encrypting messages at $\mathcal{A}$'s behest. In this game, we instead generate a key for $\mathcal{A}$ to distinguish from randomness. In fact, we set

$$Hdr = (h, h^{u_1}, \ldots, h^{u_A})$$

and give $\mathcal{A}$ (Hdr, Z).

If Z is in fact $e(g_{B+1}, h)$, then we can verify by setting $h = g^t$ for some t and expanding definitions that Hdr is in fact a valid header for the key Z. $\mathcal{A}$ will therefore be able to distinguish between the case where Z is random and that where $Z = e(g_{B+1}, h)$.

This scheme is also CCA secure under slightly stronger assumptions, but, since we do not need this property for our authentication scheme to be secure as we have defined it, we do not reproduce the proof here.