An Attack on A Traitor Tracing Scheme

Jeff Jianxin Yan[1] and Yongdong Wu[2]

[1] Computer Laboratory, University of Cambridge
Jeff.Yan@cl.cam.ac.uk

[2] Kent Ridge Digital Labs (KRDL), Singapore
wydong@krdl.org.sg

Abstract. In Crypto'99, Boneh and Franklin proposed a public key traitor tracing scheme [1], which was believed to be able to catch all traitors while not accusing any innocent users (i.e., full-tracing and error-free). Assuming that Decision Diffie-Hellman problem is unsolvable in G_q, Boneh and Franklin proved that a decoder cannot distinguish valid ciphertexts from invalid ones that are used for tracing. However, our novel pirate decoder P_3 manages to make some invalid ciphertexts distinguishable without violating their assumption, and it can also frame innocent user coalitions to fool the tracer. Neither the single-key nor arbitrary pirate tracing algorithm presented in [1] can identify all keys used by P_3 as claimed. Instead, it is possible for both algorithms to catch none of the traitors. We believe that the construction of our novel pirate also demonstrates a simple way to defeat some other black-box traitor tracing schemes in general.

Keyword: Security, black-box traitor tracing, copyright protection

1 Introduction

A traitor tracing scheme traces the source of keys used in pirate decoders for sensitive or proprietary data, such as pay-TV programmes. Those subscribed users who leak their legitimate keys are *traitors*. The pirate may reuse original traitors' keys, or use new keys generated out of them.

If a pirate decoder is tamper-resistant, a tracer might be unable to extract a key or keys used by the pirate. Without the key(s), it is impossible for the tracer to identify any traitor. Therefore, a very practical and important property of

a traitor tracing scheme is to support black-box tracing, where a decoder is treated as a black box, and its embedded key(s) can be deduced by testing how it decrypts some chosen ciphertexts.

Since Chor et al [2] introduced the concept of traitor tracing, lots of research has been done on this topic. However, many proposed schemes like [2] [6] [3] were probabilistic tracing. They tried to maximize the chance of catching just *one* of the traitors while minimizing the chance of accusing any innocent user. The traitor tracing scheme [1] proposed by Boneh and Franklin provided a deterministic tracing approach, and it was believed to able to catch *all* traitors, while not accusing any innocent users, as long as the number of traitors is at or below a collusion threshold k. The Boneh-Franklin scheme supports black-box tracing.

In this paper, we show a novel pirate decoder P_3 that defeats the Boneh-Franklin black-box traitor tracing algorithms presented in [1]. We also demonstrate a simple way to defeat some black-box traitor tracing schemes in general by following the same philosophy of P_3. Unless stated explicitly, we only discuss black-box traitor tracing in this paper.

2 The Boneh-Franklin Public Key Traitor Tracing Scheme

In the Boneh-Franklin traitor tracing scheme, there is one public encryption key, but many private decryption keys. Digital contents are encrypted with the public key and distributed through a broadcast channel, and each legitimate user can decrypt them by using his own private key. If the ciphertext is "valid" then all the decoders will produce the same plaintext output, and this is how the equipment would normally be operated in broadcast mode. However, if a pirate device appears then a tracer feeds into it "invalid" ciphertexts, that decrypt differently under different pirate decryption keys. By observing the results, he hopes to be able to identify the key or keys used by the pirate device.

Assuming that the Decision Diffie-Hellman (DDH) problem is unsolvable in G_q, Boneh and Franklin proved that a decoder cannot distinguish invalid ciphertexts from valid ones (and thus cheat), and the only way that an opponent can construct a valid new key is by taking a convex combination of known keys

supplied by traitors. Based on these results, they proposed in [1] two black-box traitor tracing algorithms for single-key pirates (where only one single key is embedded in a pirate decoder) and arbitrary pirates (i.e., all non-single-key pirates) respectively.

For the mathematical detail of both the Boneh-Franklin scheme and their tracing algorithms please refer to the Appendix of this paper or [1].

3 P_3: An Intelligent Pirate Decoder

We design an intelligent decoder P_3 ("Pirate 3") to defeat the Boneh-Franklin scheme. As shown in Fig. 1, P_3 has three decoding circuits with distinct keys. It is unnecessary to construct new keys out of traitors' keys for P_3; these three circuits may simply reuse any three original keys, say $\bar{d}_1, \bar{d}_2$ and $\bar{d}_3$. Any input ciphertext is multiplexed to all three decoding logic that independently solve S_1, S_2 and S_3 like stand-alone decoders, and then output these values to a comparator. When any two of S_1, S_2 and S_3 are equal, the comparator will output their value, e.g. S_1 when $S_1 = S_2 \neq S_3$; when $S_1 \neq S_2 \neq S_3$, it will output random bits.

On input valid ciphertexts, P_3 always has $S_1 = S_2 = S_3$, so it will work as a legitimate decoder in normal operation (e.g. content decryption).

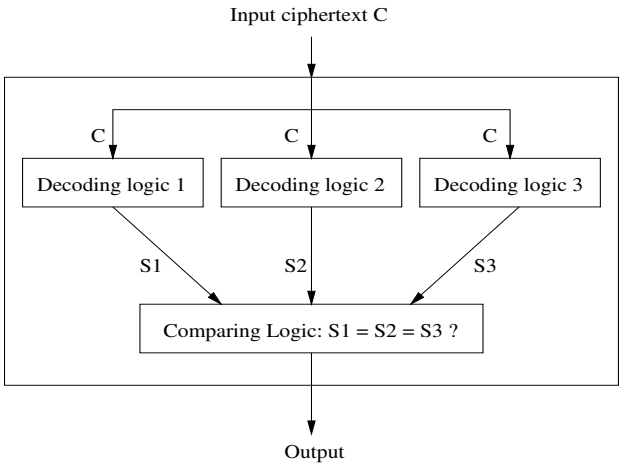

Fig. 1. The structure of a pirate decoder P_3

3.1 Single-key pirate tracing

The Boneh-Franklin single-key pirate tracing algorithm works as follows. After $2k$ rounds of invalid ciphertext tests, a tracer gets a $2k$-dimensional linear system with a unique solution. After solving this linear system, he can further work out a unique vector $\bar{d} = (\delta_1, \ldots, \delta_{2k})$, which is the embedded key. By error correction methods the tracer can identify all traitors who contribute to $\bar{d}$. There is a one-to-one corresponding relationship between the $2k$-dimensional linear system and the single embedded key. It is impossible for the single-key pirate tracing to identify all three keys used by P_3 by solving a linear system with a unique solution. Please refer to the Appendix of this paper for the mathematical detail.

3.2 Arbitrary pirate tracing

The Boneh-Franklin arbitrary pirate tracing algorithm relies on a black-box confirmation algorithm to confirm guilty user coalitions. For example, when the tracer suspects a coalition, say $\{\bar{d_1}, \bar{d_2}, \bar{d_3}\}$, each time he will choose an invalid ciphertext that decrypts the same under $\bar{d_1}, \bar{d_2}$ or $\bar{d_3}$, but differently under any other keys. By observing the decryption result, he can confirm or reject the suspected set. The tracer does black-box tracing by running the confirmation algorithm on all $\binom{n}{k}$ candidate coalitions, where n is the number of users in the system.

Firstly, we use a simple example ($n = 8, k = 3$) to show possible black-box confirmation tests that can be done for P_3 as follows. K denotes the set of all keys used by P_3.

Test A. Confirmations on all $\binom{8}{3}$ candidate coalitions

All $\binom{8}{3}$ coalition sets and their corresponding confirmation results are shown in the following table, where '*' denotes any single element of set $\Delta = \{\text{all users}\} \backslash \{\bar{d_1}, \bar{d_2}, \bar{d_3}\}$ ($A \backslash B = \{x | x \in A \text{ and } x \notin B\}$). Confirmation tests on coalitions that include any two of $\bar{d_1}, \bar{d_2}$ and $\bar{d_3}$ as elements confirm that they include K as a subset.

Coalition set	**Comparator state**	P_3**'s Output**	**Confirmation**
$\{\bar{d_1}, \bar{d_2}, \bar{d_3}\}$	$S_1 = S_2 = S_3$	S_1	yes
$\{\bar{d_1}, \bar{d_2}, *\}$	$S_1 = S_2 \neq S_3$	S_1 or S_2	yes
$\{\bar{d_1}, \bar{d_3}, *\}$	$S_1 = S_3 \neq S_2$	S_1 or S_3	yes
$\{\bar{d_2}, \bar{d_3}, *\}$	$S_1 \neq S_2 = S_3$	S_2 or S_3	yes
other sets	$S_1 \neq S_2 \neq S_3$	random bits	no

The naive application of the Boneh-Franklin tracing algorithm now simply frames innocent users. For example, user $\bar{d_1}$ builds P_3 pirates with user $\bar{d_2}$ and $\bar{d_3}$, and murders these two guys in the process. He then sells P_3 pirates, and sends the tracer an anonymous tip-off that the traitors are $\bar{d_2}$, $\bar{d_3}$ and $\bar{d_5}$. The tracer buys an instance of the pirate decoder and finds that it does indeed confirm the use of these three keys when the Boneh-Franklin arbitrary tracing algorithm is applied. The police now arrest the innocent victim $\bar{d_5}$ and charge him now just with copyright offences, but quite possibly with the murder of $\bar{d_2}$ and $\bar{d_3}$ as well.

The only reasonable way to work out the set K is to intersect all confirmed coalition sets. However, the intersection of those sets only leads to $K = \Phi$. That is to say, the tracer will catch no traitors unless he accuses innocent users.

Test B. Confirmations on all $\binom{8}{2}$ *candidate coalitions*

Only $\{\bar{d_1}, \bar{d_2}\}, \{\bar{d_1}, \bar{d_3}\}$ and $\{\bar{d_2}, \bar{d_3}\}$ can confirm that they are supersets of K, and the intersection of these three supersets still only gives out $K = \Phi$. No more information about the traitors can be obtained.

Test C. Confirmations on all $\binom{8}{1}$ *candidate coalitions*

The confirmation algorithm does not make sense when it is run upon single-element coalitions. The tracer obtains no meaningful result from confirmation tests on all $\binom{8}{1}$ candidate coalitions.

This simple example shows that the arbitrary tracing algorithm fails to identify all traitors' keys embedded in P_3 without accusing innocent users.

We analyze the applicability of P_3 for any k and n ($n \geq 2k+2$ as constrained by the Boneh-Franklin scheme). It is meaningless to discuss P_3 in the case of $k = 1$ or $k = 2$. When $k = 3, n > 8$, Test A for $\binom{n}{3}$ and Test B for $\binom{n}{2}$ coalition sets still yield the same result as above. When $k > 3, n \geq 2k + 2$ and the black-box confirmation test is run against $\binom{n}{k}$ coalition sets, only sets that have any

two of $\bar{d_1}, \bar{d_2}$ and $\bar{d_3}$ as elements confirm that they are supersets of K. However, the intersection of all those supersets is still empty, i.e., $K = \Phi$. That is to say, a tracer catches no traitor at all, when he tests P_3 by running the arbitrary pirate black-box tracing algorithm, for any n and $k \geq 3$.

In any case, neither the single-key nor the arbitrary pirate tracing algorithm presented in [1] can, as claimed, catch all traitors who contribute to the construction of P_3. P_3 manages to make some invalid ciphertext distinguishable without violating the assumption of difficulty of DDH problem in G_q, and it can maliciously respond to test queries to frame innocent coalitions (e.g., $\{\bar{d_1}, \bar{d_2}, *\}$ in the above example). That is why the tracer is fooled.

3.3 Key leakage or "Tracer, are you going to be lucky today?"

P_3 is designed to defeat Boneh-Franklin's black-box tracing algorithms by interfering its own output channel; the logic used by P_3's comparator is the interfering rules. By "framing" innocent coalitions, this interference always prevents their black-box confirmation algorithm from identifying any traitor. However, meaningful key information may sometimes leak from the interfered output channel in the process of single-key pirate tracing. That is where a tracer's jackpot is.

For the comparator in the P_3, there are five possible input combinations, i.e.,

$$S_1 = S_2 = S_3,\ S_1 = S_2 \neq S_3,\ S_1 = S_3 \neq S_2$$
$$S_2 = S_3 \neq S_1,\ S_1 \neq S_2 \neq S_3$$

It is easy to prove that neither $S_1 = S_2$, $S_1 = S_3$ or $S_2 = S_3$ always holds in P_3 during $2k$ rounds of tests in the process of single-key pirate tracing. Otherwise, the single-key pirate tracing will deduce a same vector $\bar{d}$ for two distinct keys. In the process of single-key pirate tracing against P_3, only when $S_1 = S_2$ occurs i times, and $S_1 = S_3 \neq S_2$ occurs j times and $i + j = 2k$, the tracer will have a chance to get correct outputs from P_3 to build a meaningful linear system, which will lead to successful identification of an only key $\bar{d_1}$. In any other case, there is no such key leakage, since the interference (e.g. $S_1 \neq S_2 \neq S_3$ or any others) works at least **once** during $2k$ rounds of tests, and consequently, the tracer cannot get a meaningful linear system and will catch none of the traitors. Furthermore, the criteria for choosing invalid ciphertexts for tracing do not gu-

rantee that this key leakage always happens. Actually, the leakage happens in a very small probability.

What can a tracer do when he faces a confiscated pirate, say, our P_3? Note: we only consider black-box tracing. The tracer knows his system parameters (n, k), but does not know how many traitors' keys are used in the box. Since the Boneh-Franklin scheme is k-resilient, the tracer knows that there are k possibilities for the number of traitors. Assume he starts the single-key pirate tracing algorithm first. If he is not lucky enough so that the interference works once, he obtains nothing meaningful. He will doubt that this is not a single-key pirate, and then run the black-box confirmation algorithm. However, Section 3.2 shows that the black-box confirmation of all $\binom{n}{1}, \binom{n}{2}, \ldots, \binom{n}{k}$ coalitions does not help. So the unlucky tracer cannot catch any traitor.

If the tracer is lucky, he will obtain a traitor's key $\bar{d}_1$, by single-key pirate tracing. On the other hand, as *Test C* shows in Section 3.2, the black-box confirmation algorithm cannot get any meaningful result when it is run against a single-key pirate. If the tracer doesn't know the existence of pirates like P_3, he will naively believe that he has caught all traitors, and thus stop after identifying $\bar{d}_1$ only. Even if he doesn't stop but follows the the black-box confirmation algorithm, he will still get nothing more.

Therefore, though there is possible key leakage, **it is impossible for a tracer to identify all keys in P_3 by following the Boneh-Franklin algorithms**, no matter whether he is lucky or not. Instead, **it is highly possible for the tracer to identify none of keys in P_3**.

3.4 P_3-like pirates

When only two decoding circuits are put in a pirate decoder and an appropriate comparing logic is used, a P_2 box is constructed. If the comparator in P_2 outputs S_1 when $S_1 = S_2$, but random bits when $S_1 \neq S_2$, the Boneh-Franklin's single-key pirate tracing will fail to identity any embedded keys. However, their arbitrary pirate tracing is able to catch both $\bar{d}_1$ and $\bar{d}_2$, since only a unique set $\{\bar{d}_1, \bar{d}_2\}$ can confirm that it is the superset of K, when the confirmation test is run against all $\binom{n}{2}$ coalition sets. If the comparator in P_2 outputs S_1 when $S_1 = S_2$, and outputs S_2 when $S_1 \neq S_2$, the arbitrary pirate tracing is able to

catch $\bar{d_2}$, since only sets that have $\bar{d_2}$ as an element can confirm that they are the superset of K, and the intersection of all those supersets is $\{\bar{d_2}\}$.

P_2 is not a design as good as P_3. However, it shows that P_3 is a minimal best design that defeats the attacked tracing algorithms. On the other hand, the construction of P_2 and P_3 shows that different structures or different comparator logic may lead to pirate decoders that have different properties, and it is easy to design P_3-like pirates such as $P_4, \ldots, P_k$.

4 "Traffic Analysis": A Generic Method to Defeat Black-box Traitor Tracing

Provided a tracer knows the structure of P_3 and its interference rules, it is not difficult for him to modify the Boneh-Franklin arbitrary pirate tracing algorithm to catch all traitors. For example, if a lucky tracer catches $\bar{d_1}$ as discussed in Section 3.3, he can do the black-box confirmation on $\binom{7}{2}$ candidate coalitions by excluding $\bar{d_1}$ out of the full set. Or the tracer can simply run confirmation tests on $\binom{8}{2}$ coalition sets, and the **union** of all sets, which confirm that they include traitors' keys, is the set of all traitors's keys. However, neither is strictly black-box tracing, since the correctness of tracing results is based on the fact that the structure of P_3 and its interference rules are known to the tracer, otherwise he even cannot be sure that this result is full-tracing and error-free, and consequently acceptable or not at all. Moreover, it is easy to show that neither modified algorithm provides a generic full-tracing and error-free method that is applicable to all P_3-like pirates with different comparators (i.e. interfering logic) or different numbers of decoding circuits.

The key step to catch all traitors in P_3-like pirates appears to identify interfering rules used by those pirates. Theoretically, it can be done by brute force guessing, since the tracer knows keys of all legitimate users in order to do black-box confirmations, but it is extremely inefficient.

We are not keen in looking for generic black-box full-tracing algorithms, regardless of efficient or inefficient ones, for the Boneh-Franklin scheme against arbitrary pirates, since we highly suspect that they are vulnerable to "traffic analysis" attacks that will be highlighted as follows.

Because digital content distribution systems typically use broadcast encryption schemes in order to disseminate contents such as pay-TV programmes in a secure and efficient way, the construction of P_3 shows a simple way for a pirate box to automatically distinguish tracing traffic from normal ciphertext traffic: theoretically, if the current input is a valid ciphertext, $S_1 = S_2 = S_3$ must hold; if $S_1 = S_2 = S_3$ is not true, the current input must be an invalid ciphertext. With the help of a built-in comparator (like in Fig 1) or a majority voter, a pirate is capable to analyse the cipertext traffic, and then distinguish tracing mode from the normal decoding mode.

Following a common assumption of current traitor tracing research, a pirate box does not need to support a suicide strategy, but it can simply output random bits to disturb a tracing algorithm, when its auto-sensing component detects that the tracing algorithm is running against itself. Chor et al provided a traitor tracing method for their one level scheme [2], which can be used as a black-box tracing method. Pfitzmann [6] also introduced a probabilistic black-box tracing method for the same one level scheme. Both methods are vulnerable when a P_3-like pirate is used. This is one kind of simple "traffic analysis".

Suppose there is an efficient black-box "full-tracing" algorithm for the Boneh-Franklin scheme against arbitrary pirates. Unfortunately, when any efficient tracing algorithm is running against a pirate decoder, it must present some distinguishable traffic patterns that are totally different from normal traffic. It is not difficult for a pirate box to detect and exploit the same patterns. We may add into P_3 a counter that counts the occurrence of each possible comparator state and does some statistical analysis of ciphertext traffic. It is easy for the pirate to output random bits at an appropriate time to disturb and then defeat the black-box full-tracing. This is another kind of more complicated "traffic analysis". Although Boneh and Franklin didn't state explicitly, they always assumed a pirate box stateless, and did not take account of the fact that the box can be stateful. However, it is easy to turn a stateless box into stateful by equipping it with a majority logic voting device, a counter and some memory. A stateful pirate decoder can easily protect itself from tracing algorithms based on stateless models.

Traitor tracing researchers appear to have used unrealistic threat models, which unavoidably lead to failures when an intelligent pirate box is used by a hacker.

5 Conclusion

We presented a novel pirate decoder P_3, which contains three keys and reacts appropriately when a ciphertext decrypt differently under some of them. Although [1] proved that a decoder cannot distinguish valid ciphertexts from invalid ones that are used for tracing, P_3, with the help of a built-in comparator, manages to make some of them distinguishable without breaking their DDH assumption. Furthermore, it adopts a sophisticated strategy designed to frame innocent users. The tracing algorithms presented in [1] cannot catch all traitors who contribute keys to P_3 as claimed. Instead, it is possible for them to catch none of the traitors. Using different comparator logic and/or different number of decoding circuits, an attacker can easily design other P_3-like pirates.

"Traffic analysis" following the philosophy of our novel pirate may provide a simple way to defeat some black-box traitor tracing schemes in general.

Acknowledgement

The work was done by 10 Dec 2000, and presented as [8] at the Oakland rump session on 15 May 2001, as well as a seminar at HP labs Bristol on 26 June 2001. The first author thanks Ross Anderson for his valuable comments.

References

1. D. Boneh and M. Franklin, "An Efficient Public Key Traitor Tracing Scheme", in *Advances in Cryptology - Crypto'99*, M. Wiener (Ed.), *Lecture Notes in Computer Science* 1666, 1999, pp 338-353
2. B. Chor, A. Fiat and M. Naor, "Tracing Traitors", in *Advances in Cryptology - Crypto'94*, Y. G. Desmedt (Ed.), *Lecture Notes in Computer Science* 839, 1994, pp257-270
3. M. Naor and B. Pinkas, "Threshold Traitor Tracing", in *Advances in Cryptology - Crypto'98*, Springr-Verlag LNCS 1462, 502-517,1998

4. T. Okamoto, S. Uchiyama, “A new public key cryptosystem as secure as factoring”, in Proc. of Eurocrypt ’98, pp. 308-318
5. P. Paillier, “Public-Key Cryptosystems Based on Discrete Logarithm Residues”, in proc. Eurocrypt ’99, pp. 223-238
6. B. Pfitzmann, “Trails of traced traitors”, Information Hiding Workshop, Cambridge, UK, LNCS 1174, 49-64,1996
7. D. R. Stinson and R. Wei, “Combinatorial properties and constructions of traceability schemes and frameproof codes”, SIAM J. on Discrete Math, Vol.11, 1, 41-53,1998
8. Jeff Jianxin Yan and Yongdong Wu. “An Attack on Black-box Traitor Tracing Schemes”. Rump session, IEEE Symposium on Security and Privacy, Oakland, USA, May 2001. at http://www.cl.cam.ac.uk/~jy212/oakland01.pdf.

Appendix

The Boneh-Franklin Traitor Tracing Scheme

Key Generation: Let G_q be a group of prime order q, and $g \in G_q$ be a generator of G_q. For $i = 1, ..., 2k$ choose a random $r_i \in Z_q$ and compute $h_i = g^{r_i}$. The public key is $(y, h_1, ..., h_{2k})$, where $y = \prod_{i=1}^{2k} h_i^{\alpha_i}$ for random $\alpha_1, ..., \alpha_{2k} \in Z_q$.

Assume $l \geq 2k + 2$, $q > max(l, 2k)$. Define a $(l - 2k) \times l$ matrix A as follows.

$$\mathbf{A} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & 3 & \dots & l \\ 1^2 & 2^2 & 3^2 & \dots & l^2 \\ 1^3 & 2^3 & 3^3 & \dots & l^3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1^{l-2k-1} & 2^{l-2k-1} & 3^{l-2k-1} & \dots & l^{l-2k-1} \end{pmatrix} (mod\, q)$$

Let b_1, ..., b_{2k} be a basis of the linear space of vectors satisfying $A\bar{x} = 0\,(mod\, q)$. By regarding these $2k$ vectors as the columns of a matrix we obtain a $l \times 2k$ matrix $B = (b_1\, b_2\, \dots\, b_{2k})$. Denote the ith row of B by $\gamma^{(i)} = (\gamma_1, \dots, \gamma_{2k})$, and $\theta_i = (\sum_{j=1}^{2k} r_j \alpha_j)/(\sum_{j=1}^{2k} r_j \gamma_j) \quad (mod\, q)$. Multiplying the ith $(i = 1, 2, ..., l)$ row of B by θ_i, we get a new matrix B',

$$\mathbf{B'} = \begin{pmatrix} \delta_{1,1} & \delta_{1,2} & \dots & \delta_{1,2k} \\ \delta_{2,1} & \delta_{2,2} & \dots & \delta_{2,2k} \\ \vdots & \vdots & \ddots & \vdots \\ \delta_{l,1} & \delta_{l,2} & \dots & \delta_{l,2k} \end{pmatrix}$$

where any row i satisfies $\sum_{j=1}^{2k} r_j \delta_{i,j} = \sum_{j=1}^{2k} r_j \alpha j$. And $\bar{d_i} = (\delta_{i,1}, ..., \delta_{i,2k})$ is the private key of the ith subscriber.

Encryption: Pick a random element $a \in Z_q$, then encrypt a message M in G_q to ciphertext $C = (M \cdot y^a, h_1^a, \dots, h_{2k}^a)$.

Decryption: To decrypt a ciphertext $C = (S, H_1, ..., H_{2k})$ using user i'th private key $\bar{d_i} = (\delta_{i,1}, ..., \delta_{i,2k})$, compute $M = \frac{S}{\prod_{j=1}^{2k} H_j^{\delta_{i,j}}}$. Since,

$$\prod_{j=1}^{2k} H_j^{\delta_{i,j}} = (g^{\sum_{j=1}^{2k} r_j \delta_{i,j}})^a = (g^{\sum_{j=1}^{2k} r_j \alpha_j})^a = (\prod_{j=1}^{2k} g^{r_j \alpha_j})^a = (\prod_{j=1}^{2k} h_j^{\alpha_j})^a = y^a$$

Tracing Algorithms

Black-box tracing of single-key pirates: For a single-key pirate, the basic idea of the black-box tracing is to observe the pirate decoder's behavior on an invalid ciphertext $\tilde{C} = (S, h_1^{z_1}, ..., h_{2k}^{z_{2k}})$, where the non-constant vector $\bar{z}$ is chosen by the tracer. The tracing algorithm deduces from a single-key pirate its embedded key $\bar{d} = (\delta_1, \ldots, \delta_{2k})$ as follows. On input $\tilde{C}$, which is invalid since the h_i's are raised to different powers, the pirate decoder must respond with A, where $A = \frac{S}{\prod_{j=1}^{2k} h_j^{\delta_j z_j}}$, since it cannot distinguish $\tilde{C}$ from a valid ciphertext. By querying with invalid ciphertexts the tracer learns the value $\prod_{j=1}^{2k} h_j^{\delta_j z_j} = S/A$ for vectors $\bar{z}$ of its choice. After $2k$ rounds of invalid ciphertext tests, a tracer gets a following $2k$ simultaneous equations,

$$\begin{cases} \prod_{j=1}^{2k} h_j^{\delta_j z_{1,j}} = c_1 \\ \prod_{j=1}^{2k} h_j^{\delta_j z_{2,j}} = c_2 \\ \cdots \\ \prod_{j=1}^{2k} h_j^{\delta_j z_{2k,j}} = c_{2k} \end{cases} \tag{1}$$

where $\bar{z}_1, \ldots, \bar{z}_{2k}$ are linearly independent vectors, which are **randomly** chosen by the tracer; $c_1, \ldots, c_{2k}$ are values of S/A for each round. (1) is equivalent to a following linear system that has $\ln h_1^{\delta_1}, \ln h_2^{\delta_2}, \ldots, \ln h_1^{\delta_1}$ as variables.

$$\begin{pmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,2k} \\ z_{2,1} & z_{2,2} & \cdots & z_{2,2k} \\ \vdots & \vdots & \ddots & \vdots \\ z_{2k,1} & z_{2k,2} & \cdots & z_{2k,2k} \end{pmatrix} \begin{pmatrix} \ln h_1^{\delta_1} \\ \ln h_2^{\delta_2} \\ \vdots \\ \ln h_{2k}^{\delta_{2k}} \end{pmatrix} = \begin{pmatrix} \ln c_1 \\ \ln c_2 \\ \vdots \\ \ln c_{2k} \end{pmatrix} \tag{2}$$

If $\ln c_1 = \ln c_2 = \cdots = \ln c_{2k} = 0$, (2) is a homogeneous linear system; otherwise, it is a non-homogeneous linear system. Since $\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_{2k}$ are linearly independent, the $2k \times 2k$ matrix $H = (\bar{z}_1\, \bar{z}_2\, \ldots\, \bar{z}_{2k})^T$ has a rank of $2k$. When (2) is a homogeneous system, it has only a trivial solution, i.e., $\ln h_1^{\delta_1} = \ln h_2^{\delta_2} = \cdots = \ln h_1^{\delta_1} = 0$, so $(\delta_1, \delta_2, \ldots, \delta_{2k}) = (0, 0, \ldots, 0)$ is the embedded key. When (2) is a non-homogeneous system, H guarantees that it has a unique (non-trivial) solution,

$$(\ln h_1^{\delta_1}\ \ln h_2^{\delta_2}\ \ldots \ln h_1^{\delta_1})^T = H^{-1}(\ln c_1\ \ln c_2\ \cdots \ln c_{2k})^T.$$

The tracer can solve a unique solution for $h_1^{\delta_1}$, ..., $h_{2k}^{\delta_{2k}}$. Since he also knows the discrete log of the h_i's to the base g, he can compute g^{δ_1}, ..., $g^{\delta_{2k}}$. Afterwards,

he can recover a unique vector $\bar{d} = (\delta_1, \ldots, \delta_{2k})$, which is the embedded key, from (g^{δ_1}, ..., $g^{\delta_{2k}}$) by using recent number theory results on trapdoors of the discrete log modulo p^2q and modulo N^2 [4] [5].

After holding $\bar{d}$, the tracer can identify all traitors (without accusing any innocent users) by error correction methods, since the set of rows of matrix B is actually l codewords over Z_q^{2k} of a Reed-Solomon code.

Note: Running the above algorithm on P_3, a tracer gets a linear system like (1), which is impossible to lead to identify all the three distinct keys in P_3.

Black-box tracing of arbitrary pirates: For an arbitrary pirate, the traitor tracing is based on a black-box confirmation algorithm. The tracer suspects a particular set T of at most k traitors. Let $\bar{d_1}, ..., \bar{d_k}$ be the keys of those k traitors. To confirm his suspicion of T, the tracer queries the decoder with an invalid ciphertext $\tilde{C} = (S, g^{z_1}, \ldots, g^{z_{2k}})$, where the vector $\bar{z} = (z_1, \ldots, z_{2k})$ satisfies $\bar{z} \cdot \bar{d_i} = w$ for all $i \in T$. The decoder cannot distinguish this invalid ciphertext from a valid one, and it will respond with $A = S/\prod g^{z_i \delta_i}$ where $(\delta_1, \ldots, \delta_{2k})$ is some representation of y. If, for a suspect coalition T, the pirate box always responds with $A = S/g^w$, then the pirate must possess a subset of the keys belonging to T. Suppose n is the number of subscribed users in the system. The tracer does black-box tracing to catch all traitors contributing to an arbitrary pirate by running the confirmation algorithm on all $\binom{n}{k}$ candidate coalitions.

Though it is public-key based, the Boneh-Franklin scheme doesn't guarantee the non-repudiation property defined by [6] in case the tracer needs black-box confirmation.