An Efficient Traitor Tracing Scheme

and

Pirates 2.0

Duong Hieu Phan, Paris 8

(*joint work with Olivier Billet, Orange Labs*)

ENS Crypto Seminar — Jan. 15, 2009

Outline

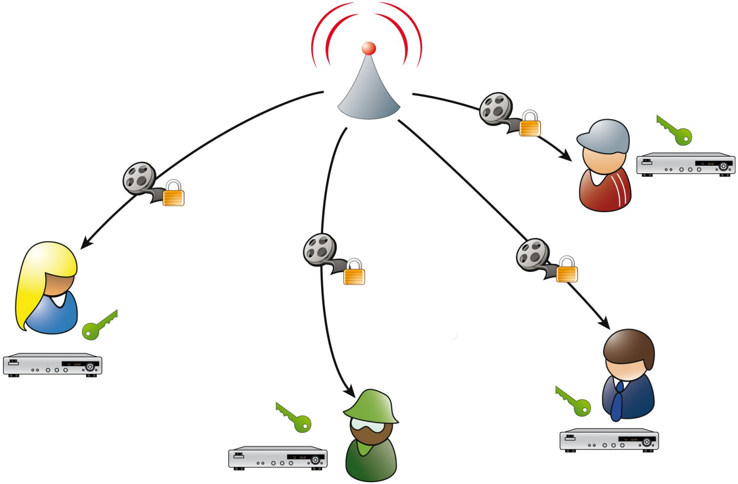
1. Code-based Traitor Tracing
   - Collusion Secure Codes
   - Tardos Code supporting Erasure
   - Constant Size Ciphertext

2. Pirates 2.0
   - Pirate 2.0 vs. NNL Schemes
   - Pirates 2.0 against Code Based Schemes

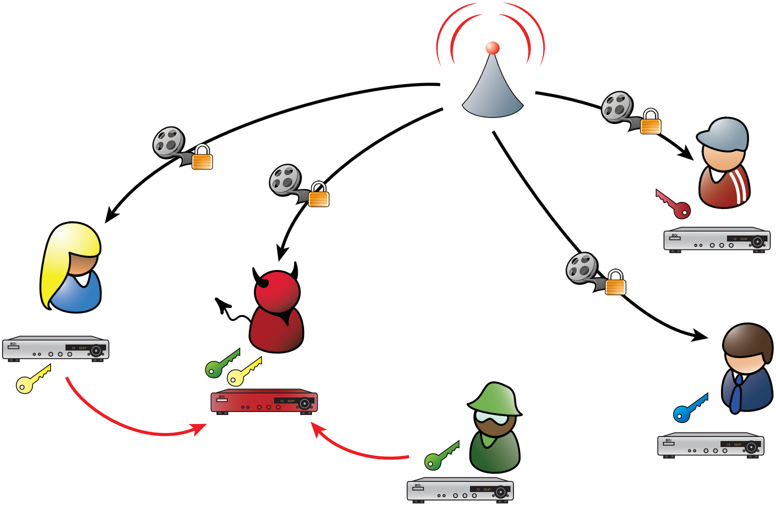
Outline

1. Code-based Traitor Tracing
   - Collusion Secure Codes
   - Tardos Code supporting Erasure
   - Constant Size Ciphertext
2. Pirates 2.0
   - Pirate 2.0 vs. NNL Schemes
   - Pirates 2.0 against Code Based Schemes

Traitor Tracing



Traitor Tracing



Main Approaches for Constructing Traitor Tracing

Tree based Approach

One of the most famous schemes: Naor–Naor–Lotspiech (2001)

Algebraic Approach

Some schemes: Boneh–Franklin (1999), Boneh–Sahai–Waters (2006), ...

Code-based Approach

Some schemes: Boneh–Shaw 99, Kiayias–Yung 01, Chabanne–Phan–Pointcheval 05, Sirvent 07, ...

Main Approaches for Constructing Traitor Tracing

Tree based Approach

One of the most famous schemes: Naor–Naor–Lotspiech (2001)

Algebraic Approach

Some schemes: Boneh–Franklin (1999), Boneh–Sahai–Waters (2006), ...

Code-based Approach

Some schemes: Boneh–Shaw 99, Kiayias–Yung 01, Chabanne–Phan–Pointcheval 05, Sirvent 07, ...

Collusion secure Codes

Traitor 1	1	0	1	0	1	1	0	1	0	1	1	1	0	0	1	···	0	0	1	0	0
Traitor 2	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	0	0	1	0	1
Traitor 3	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	1	0	1	0	0

Collusion secure Codes

Traitor 1	1	0	1	0	1	1	0	1	0	1	1	1	0	0	1	···	0	0	1	0	0
Traitor 2	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	0	0	1	0	1
Traitor 3	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	1	0	1	0	0
Pirate	1	0	1	0	1			1	0	1	1		0	0	1	···		0	1	0	

Marking Assumption

At positions where all the traitors get the same bit,
the pirate codeword must retain that bit

From Collusion Secure Codes to Traitor Tracing

KGen :

Table 0	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	...	$k_{0,\ell}$
Table 1	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	...	$k_{1,\ell}$

From Collusion Secure Codes to Traitor Tracing

KGen :

Table 0	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	...	$k_{0,\ell}$
Table 1	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	...	$k_{1,\ell}$
Codeword i	1	1	0	1	0	...	1

From Collusion Secure Codes to Traitor Tracing

KGen :

Table 0	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	...	$k_{0,\ell}$
Table 1	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	...	$k_{1,\ell}$

Codeword i	1	1	0	1	0	...	1
user i	$k_{1,1}$	$k_{1,2}$	$k_{0,3}$	$k_{1,4}$	$k_{0,5}$	...	$k_{1,\ell}$

From Collusion Secure Codes to Traitor Tracing

KGen :

Table 0	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	...	$k_{0,\ell}$
Table 1	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	...	$k_{1,\ell}$
Codeword i	1	1	0	1	0	...	1
user i	$k_{1,1}$	$k_{1,2}$	$k_{0,3}$	$k_{1,4}$	$k_{0,5}$	...	$k_{1,\ell}$

Enc :

Message	m_1	m_2	m_3	m_4	m_5	...	m_ℓ

From Collusion Secure Codes to Traitor Tracing

KGen :

Table 0	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	...	$k_{0,\ell}$
Table 1	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	...	$k_{1,\ell}$

Codeword *i*	1	1	0	1	0	...	1
user *i*	$k_{1,1}$	$k_{1,2}$	$k_{0,3}$	$k_{1,4}$	$k_{0,5}$	...	$k_{1,\ell}$

Enc :

Message	m_1	m_2	m_3	m_4	m_5	...	m_ℓ
Ciphertext	$c_{0,1}$	$c_{0,2}$	$c_{0,3}$	$c_{0,4}$	$c_{0,5}$	...	$c_{0,\ell}$
	$c_{1,1}$	$c_{1,2}$	$c_{1,3}$	$c_{1,4}$	$c_{1,5}$	...	$c_{1,\ell}$

From Collusion Secure Codes to Traitor Tracing

KGen :

Table 0	$k_{0,1}$	$k_{0,2}$	$k_{0,3}$	$k_{0,4}$	$k_{0,5}$	...	$k_{0,\ell}$
Table 1	$k_{1,1}$	$k_{1,2}$	$k_{1,3}$	$k_{1,4}$	$k_{1,5}$	...	$k_{1,\ell}$
Codeword i	1	1	0	1	0	...	1
user i	$k_{1,1}$	$k_{1,2}$	$k_{0,3}$	$k_{1,4}$	$k_{0,5}$	...	$k_{1,\ell}$

Enc :

Message	m_1	m_2	m_3	m_4	m_5	...	m_ℓ
Ciphertext	$c_{0,1}$	$c_{0,2}$	$c_{0,3}$	$c_{0,4}$	$c_{0,5}$	...	$c_{0,\ell}$
	$c_{1,1}$	$c_{1,2}$	$c_{1,3}$	$c_{1,4}$	$c_{1,5}$	...	$c_{1,\ell}$

Tracing Traitors

- At each position j, send $c_{0,j}$ and $c_{1,j}$ corresponding to two different messages m_j and $m'_j \rightarrow v_j \rightarrow$ a pirate codeword v
- From tracing algorithm of Secure Code, identify traitors

Pros and Cons

Pros

- Constant ciphertext rate
- Black-box Tracing

Pros and Cons

Pros

- Constant ciphertext rate
- Black-box Tracing

Cons 1

- The pirate may ignore some positions j
  in order to make the tracing procedure fail
- Solution (Kiayias–Yung): Use an All-or-Nothing Transform

$$M = M_1 || \cdots || M_\ell = AONT(m_1 || \cdots || m_\ell)$$

Pros and Cons

Pros

- Constant ciphertext rate
- Black-box Tracing

Cons 1

- The pirate may ignore some positions j
  in order to make the tracing procedure fail
- Solution (Kiayias–Yung): Use an All-or-Nothing Transform

$$M = M_1 || \cdots || M_\ell = AONT(m_1 || \cdots || m_\ell)$$

Cons 2

- Ciphertext size is very large, user key is also very large
- With AONT, users need to receive the whole ciphertext to be able to decrypt a single bit of the plaintext

Codes based Approach: Solutions

Sirvent

- Objective: Getting rid of AONT
- Advantage: Progressive Decryption
- Solution: Boneh–Shaw Code supporting erasure

Codes based Approach: Solutions

Sirvent

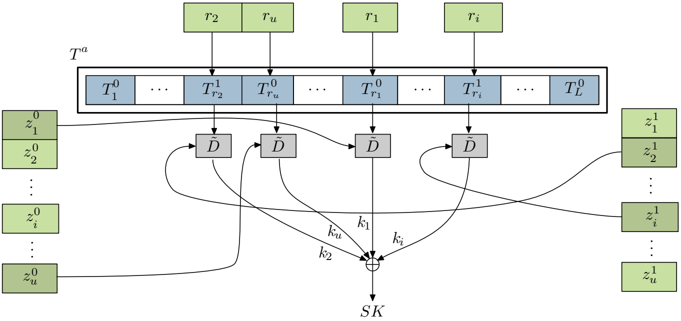
- Objective: Getting rid of AONT
- Advantage: Progressive Decryption
- Solution: Boneh–Shaw Code supporting erasure

Our Work: achieving constant size ciphertexts

- Encryption: use only some randomly chosen positions from a large code for each ciphertext
  (Boneh–Naor independently use single positions at CCS'08)
- Construction of Tardos' Code supporting erasure
  (Boneh–Naor rely on Boneh–Shaw codes supporting erasure)
- About the length of Tardos' Code vs. Boneh–Shaw Code

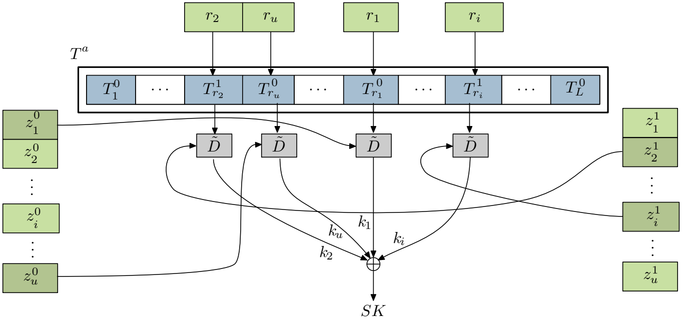
$$O(c^2 log(n/\epsilon)\text{ vs. }O(c^4 log(n/\epsilon)$$

Achieving Constant Size Ciphertexts



- Choose u random positions $r_1, \cdots, r_u$
- Decompose $SK = \bigoplus_1^u k_i$
  each k_i is encrypted using the key at position r_i

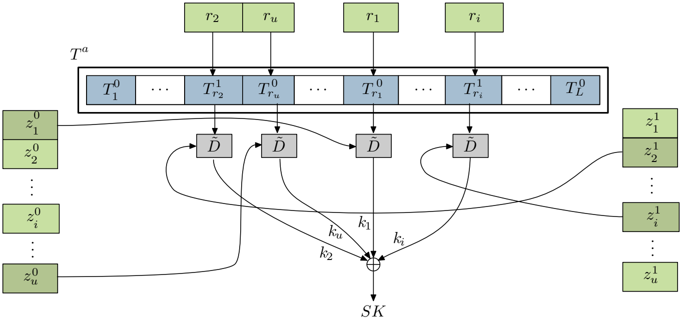
Constant Size Ciphertexts: Remarks



Perfect Pirate Decoder

The classical tracing procedure works well

Constant Size Ciphertexts: Remarks



Imperfect Pirate Decoder

If the pirate decoder decides to erase its keys at rate α:

- The pirate can decrypt with a probability of $(1 - \alpha)^u$
- The classical tracing procedure does not work anymore
- Solution: Collusion Secure Codes supporting Erasure

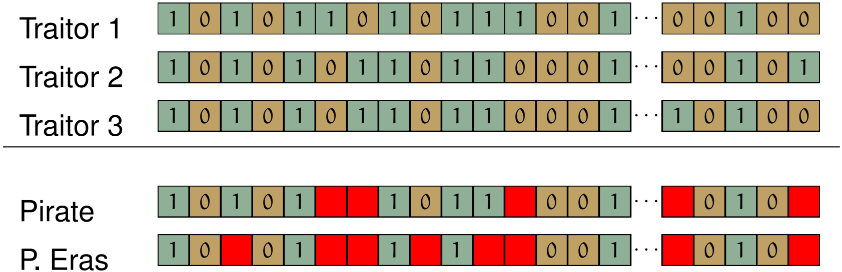
Codes Supporting Erasure

Traitor 1	1	0	1	0	1	1	0	1	0	1	1	1	0	0	1	···	0	0	1	0	0
Traitor 2	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	0	0	1	0	1
Traitor 3	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	1	0	1	0	0

Codes Supporting Erasure

Traitor 1	1	0	1	0	1	1	0	1	0	1	1	1	0	0	1	···	0	0	1	0	0
Traitor 2	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	0	0	1	0	1
Traitor 3	1	0	1	0	1	0	1	1	0	1	1	0	0	0	1	···	1	0	1	0	0
Pirate	1	0	1	0	1			1	0	1	1		0	0	1	···		0	1	0	

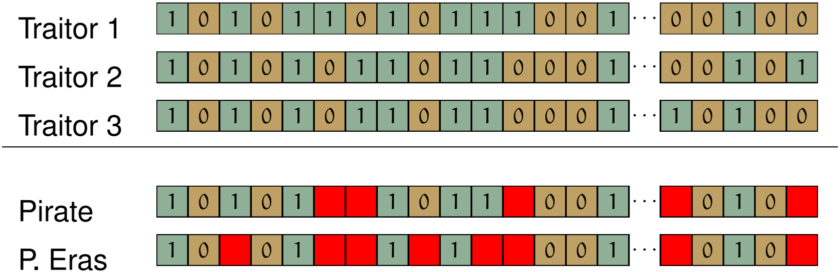
Codes Supporting Erasure



Constructions

- Sirvent, Boneh–Naor: Boneh–Shaw Code supporting erasure

Codes Supporting Erasure



Constructions

- Sirvent, Boneh–Naor: Boneh–Shaw Code supporting erasure
- No known Tardos Code supporting erasure

Tardos' Secure Code

user 1
user 2
user 3
user 4

Construction

Tardos' Secure Code

	p_1
user 1	1
user 2	0
user 3	1
user 4	0

Construction

- each p_i is randomly chosen relatively close to 0 or 1
- for each user j, randomly draw cell w_{ji}:

$$\Pr[w_{ji} = 1] = p_i, \qquad \Pr[w_{ji} = 0] = 1 - p_i$$

Tardos' Secure Code

	p_1	p_2
user 1	1	1
user 2	0	0
user 3	1	1
user 4	0	1

Construction

- each p_i is randomly chosen relatively close to 0 or 1
- for each user j, randomly draw cell w_{ji}:

$$\Pr[w_{ji} = 1] = p_i, \qquad \Pr[w_{ji} = 0] = 1 - p_i$$

Tardos' Secure Code

	p_1	p_2	p_3
user 1	1	1	0
user 2	0	0	0
user 3	1	1	0
user 4	0	1	1

Construction

- each p_i is randomly chosen relatively close to 0 or 1
- for each user j, randomly draw cell w_{ji}:

$$\Pr[w_{ji} = 1] = p_i, \qquad \Pr[w_{ji} = 0] = 1 - p_i$$

Tardos' Secure Code

	p_1	p_2	p_3	p_4
user 1	1	1	0	1
user 2	0	0	0	1
user 3	1	1	0	0
user 4	0	1	1	1

Construction

- each p_i is randomly chosen relatively close to 0 or 1
- for each user j, randomly draw cell w_{ji}:

$$\Pr[w_{ji} = 1] = p_i, \qquad \Pr[w_{ji} = 0] = 1 - p_i$$

Tardos' Secure Code

	p_1	p_2	p_3	p_4	p_5
user 1	1	1	0	1	1
user 2	0	0	0	1	0
user 3	1	1	0	0	0
user 4	0	1	1	1	0

Construction

- each p_i is randomly chosen relatively close to 0 or 1
- for each user j, randomly draw cell w_{ji}:

$$\Pr[w_{ji} = 1] = p_i, \qquad \Pr[w_{ji} = 0] = 1 - p_i$$

Tardos' Secure Code

	p_1	p_2	p_3	p_4	p_5	...	p_ℓ
user 1	1	1	0	1	1	...	1
user 2	0	0	0	1	0	...	1
user 3	1	1	0	0	0	...	0
user 4	0	1	1	1	0	...	1

Construction

- each p_i is randomly chosen relatively close to 0 or 1
- for each user j, randomly draw cell w_{ji}:

$$\Pr[w_{ji} = 1] = p_i, \qquad \Pr[w_{ji} = 0] = 1 - p_i$$

Tardos' Secure Code: Tracing

Tracing: Given a codeword v

A user u is declared guilty if:

$$f(u,v) = \sum_{i=1}^{\ell} v_i U_i \geq Z(= 20c \log 1/\epsilon)$$

where:

$$U_i = \begin{cases} \sqrt{\frac{1-p_i}{p_i}} & \text{if } u_i = 1 \\ -\sqrt{\frac{p_i}{1-p_i}} & \text{if } u_i = 0 \end{cases}$$

Remark

When $v_i = 1$, the user u is more suspicious if $u_i = 1$ and less suspicious otherwise.

Coalition $\mathcal{C}$ of c traitors

Strategy for coalitions of c traitors

Produce a codeword v such that

$$S = \sum_{u_j \in \mathcal{C}} f(u_j, v) = \sum_{i=1}^{\ell} v_i \Big(\sum_{u_j \in \mathcal{C}} U_{ji}\Big) \leq c \times Z$$

Remark

- If $v = 0^\ell$ then $f(\mathcal{C}, v) = 0$
- However, the pirate cannot produce this codeword
  At a position, if all traitors receive bit 1, it should retain bit 1

Coalition $\mathcal{C}$ of c traitors

$$S = \sum_{u_j \in \mathcal{C}} f(u_j, v) = \sum_{i=1}^{\ell} v_i (\sum_{u_j \in \mathcal{C}} U_{ji}) \leq c \times Z$$

Tardos shows that:

- For columns where $\mathcal{C}$ have both 0 and 1, the choice of v in any $\mathcal{C}$-strategy has a minor effect on the expectation of S *i.e.* the wins and loses almost cancel out
- The increase of S coming from all 1 columns is enough to make $S \leq c \times Z$ with negligible probability:

$$\Pr[S \leq c \times Z] \leq \epsilon^{c/4}$$

- Code length:

$$100c^2 log(n/\epsilon)$$

Tardos' Code supporting erasure: Innocent users

Double Tardos Code supporting one half erasure

- If in original Tardos' Code,
  an innocent user is accused with probability ϵ,
- Then in *Double Tardos supporting one half erasure*,
  an innocent user is accused with the same probability ϵ

Tardos' Code supporting erasure: Innocent users

Double Tardos Code supporting one half erasure

- If in original Tardos' Code,
  an innocent user is accused with probability ϵ,
- Then in *Double Tardos supporting one half erasure*,
  an innocent user is accused with the same probability ϵ

Key Fact in Tardos Code

Tardos' Code supporting erasure: Innocent users

Double Tardos Code supporting one half erasure

- If in original Tardos' Code,
  an innocent user is accused with probability ϵ,
- Then in *Double Tardos supporting one half erasure*,
  an innocent user is accused with the same probability ϵ

Key Fact in Tardos Code

- codewords of users are chosen totally independently from each others

Tardos' Code supporting erasure: Innocent users

Double Tardos Code supporting one half erasure

- If in original Tardos' Code,
  an innocent user is accused with probability ϵ,
- Then in *Double Tardos supporting one half erasure*,
  an innocent user is accused with the same probability ϵ

Key Fact in Tardos Code

- codewords of users are chosen totally independently from each others
- one can consider that the pirate codeword *v* is fixed before the codeword of an innocent user is selected

Tardos' Code supporting erasure: Innocent users

Double Tardos Code supporting one half erasure

- If in original Tardos' Code,
  an innocent user is accused with probability ϵ,
- Then in *Double Tardos supporting one half erasure*,
  an innocent user is accused with the same probability ϵ

Key Fact in Tardos Code

- codewords of users are chosen totally independently from each others
- one can consider that the pirate codeword v is fixed before the codeword of an innocent user is selected
- Tardos: " not only is the overall probability of the event $j \in \sigma(\rho(\mathcal{C}))$ bounded by ϵ, but conditioned on any set of values p_i and v, the probability of $j \in \sigma(y)$ is bounded by ϵ "

Tardos' Code supporting erasure: Tracing traitors

Strategy of Pirate

- If the pirate erases a position where he has both 0 and 1, he does not take advantage from the erasure. He can simply put 0 for that position in the pirate codeword
- The real problem comes from the fact that the pirate can erase positions at all 1 columns!

Tardos' Code supporting erasure: Tracing traitors

Strategy of Pirate

- If the pirate erases a position where he has both 0 and 1, he does not take advantage from the erasure. He can simply put 0 for that position in the pirate codeword
- The real problem comes from the fact that the pirate can erase positions at all 1 columns!

Solution to the erasure of all 1 columns

Tardos' Code supporting erasure: Tracing traitors

Strategy of Pirate

- If the pirate erases a position where he has both 0 and 1, he does not take advantage from the erasure. He can simply put 0 for that position in the pirate codeword
- The real problem comes from the fact that the pirate can erase positions at all 1 columns!

Solution to the erasure of all 1 columns

- Putting many *fake all* 1 *columns* in the code, at random positions k: $p_k = 1$

Tardos' Code supporting erasure: Tracing traitors

Strategy of Pirate

- If the pirate erases a position where he has both 0 and 1, he does not take advantage from the erasure. He can simply put 0 for that position in the pirate codeword
- The real problem comes from the fact that the pirate can erase positions at all 1 columns!

Solution to the erasure of all 1 columns

- Putting many *fake all* 1 *columns* in the code, at random positions k: $p_k = 1$
- The adversary cannot distinguish a real all 1 column from a fake all 1 column

Tardos' Code supporting erasure: Tracing traitors

Strategy of Pirate

- If the pirate erases a position where he has both 0 and 1, he does not take advantage from the erasure. He can simply put 0 for that position in the pirate codeword
- The real problem comes from the fact that the pirate can erase positions at all 1 columns!

Solution to the erasure of all 1 columns

- Putting many *fake all* 1 *columns* in the code, at random positions k: $p_k = 1$
- The adversary cannot distinguish a real all 1 column from a fake all 1 column
- Erasing half of all 1 columns, there still remain one half of real all 1 columns

Tardos' Code supporting erasure of rate $1/4$

$$\left|\begin{array}{cccc|cccc|cccc|cccc} 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right|$$

Code of four times the length of a normal Tardos' Code

- Two normal Tardos' Codes
- Two fake Tardos Codes of all 1 columns, randomly incorporated in the above two normal Tardos Codes

Tardos' Code supporting erasure of rate $1/4$

$$\left|\begin{array}{cccc|cccc|cccc|cccc} 1 & \color{red}{1} & 0 & \color{red}{1} & \color{red}{1} & 1 & 0 & \color{red}{1} & 0 & \color{red}{1} & \color{red}{1} & 1 & \color{red}{1} & 0 & 0 & \color{red}{1} \\ 0 & \color{red}{1} & 1 & \color{red}{1} & \color{red}{1} & 0 & 1 & \color{red}{1} & 1 & \color{red}{1} & \color{red}{1} & 1 & \color{red}{1} & 0 & 0 & \color{red}{1} \\ 0 & \color{red}{1} & 0 & \color{red}{1} & \color{red}{1} & 1 & 0 & \color{red}{1} & 1 & \color{red}{1} & \color{red}{1} & 1 & \color{red}{1} & 1 & 0 & \color{red}{1} \\ 0 & \color{red}{1} & 1 & \color{red}{1} & \color{red}{1} & 1 & 0 & \color{red}{1} & 0 & \color{red}{1} & \color{red}{1} & 1 & \color{red}{1} & 0 & 0 & \color{red}{1} \end{array}\right|$$

Code of four times the length of a normal Tardos' Code

- Two normal Tardos' Codes
- Two fake Tardos Codes of all 1 columns, randomly incorporated in the above two normal Tardos Codes

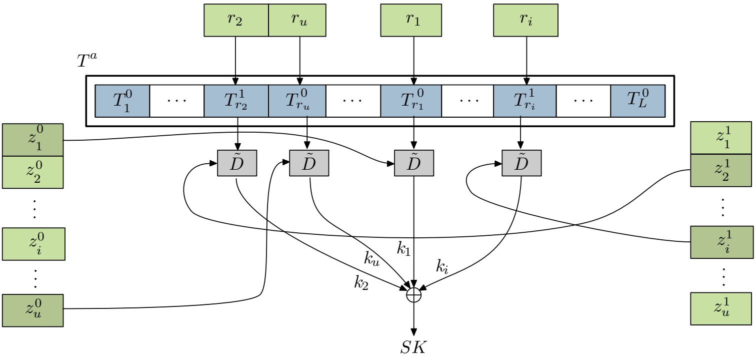
Tardos' Code supporting erasure of rate $1/4$

$$\left|\begin{array}{cccc|cccc|cccc|cccc|} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right.$$

Analysis

- Erasing $1/4$, at least one normal Tardos Code remains $\Rightarrow$ sufficient to prevent innocent people from being accused
- Erasing $1/4$ implies erasing less than one half of all 1 columns
- As pirate cannot distinguish between fake all 1 columns and normal all 1 columns, the remaining normal all 1 columns suffice to accuse traitors as in original Tardos' Code

Recall our Scheme



Remark

With an erasure rate of $1/4$, a pirate has only a probability of $(3/4)^u$ of successfully decrypting ciphertexts

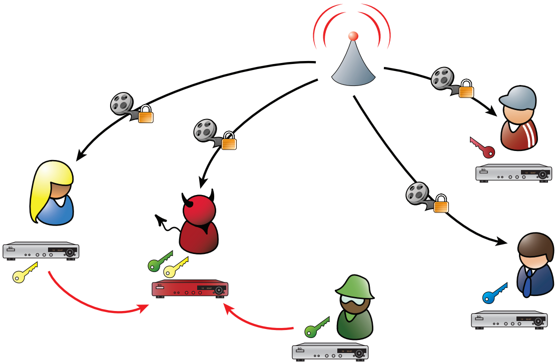
Comparison between schemes

Schemes	User key size	Ciphertext size	Enc time	Dec time
BF99	$O(1)$	$O(c)$	$O(c)$ exp	$O(c)$ exp
BSW06	$O(1)$	$\sqrt{N}$	$O(\sqrt{N})$ exp	$O(1)$ p/r
NNL01	$O(\log^2(N))$	$O(r)$	$O(\log(n))$	$O(1)$
BN08	$O(c^4 \log(N/\epsilon))$	$O(1)$	$O(1)$	$O(1)$
Ours	$O(c^2 \log(N/\epsilon))$	$O(1)$	$O(1)$	$O(1)$

Figure: Comparison between schemes

Outline

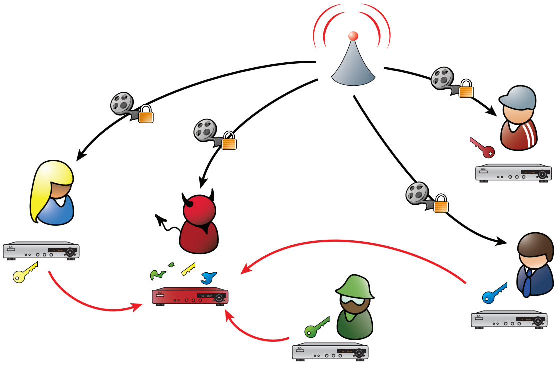
Collusion in Classical Model



Fact

- Each user contributes its whole key
- Traitors should trust each other

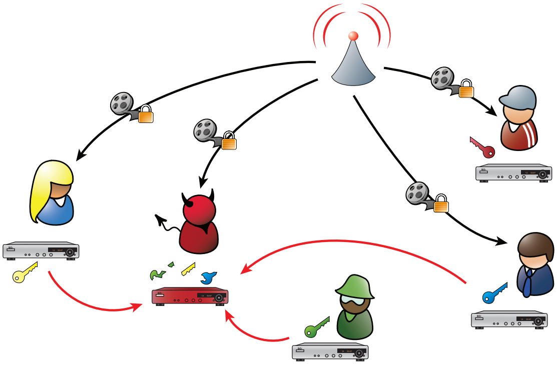
Pirates 2.0: Traitors Collaborating in Public



Principle

Each traitor contributes a partial or derived information

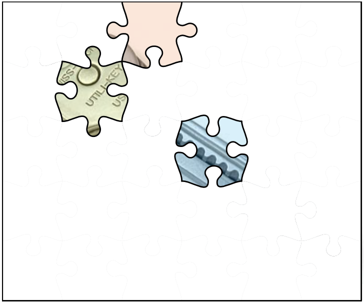
Pirates 2.0: Traitors Collaborating in Public



Anonymity level of a traitor

Number of users in system that share traitor's contributed material

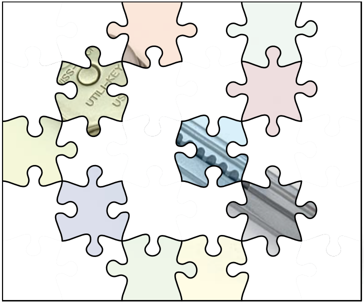
Practical Impact of Pirates 2.0



Collusion size

- Traitors do not need to trust someone
- Guaranteed anonymity is a big incentive to contribute secrets
- Even partial information extracted from tamper resistant or obfuscated decoders can be useful

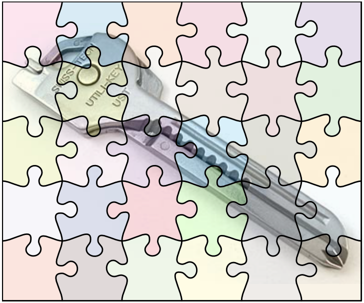
Practical Impact of Pirates 2.0



Static vs. Adaptative

- The classical model of pirate is static:
  coalitions consist of randomly drawn decoders
- In a Pirates 2.0 attacks,
  traitors can contribute information adaptatively

Practical Impact of Pirates 2.0



Application

- In the 2.0 internet, a server collects the traitors' contributions
- Any client of the server can produce a pirate decoder
- Dynamic coalitions: traitors only contribute missing pieces
  $\Rightarrow$ no need for centralized server, peer-to-peer is OK

Classical Tracing vs. Pirates 2.0

Classical Tracing vs. Pirates 2.0

Classical assumption for tracing

On input a valid ciphertext, pirate decoder "should" return the correct plaintext, otherwise it is useless

Classical Tracing vs. Pirates 2.0

Classical assumption for tracing

On input a valid ciphertext, pirate decoder “should” return the correct plaintext, otherwise it is useless

Reasonable in classical model

As soon as a pirate collects a key,
he is able decrypt all valid ciphertexts

Classical Tracing vs. Pirates 2.0

Classical assumption for tracing

On input a valid ciphertext, pirate decoder “should” return the correct plaintext, otherwise it is useless

Reasonable in classical model

As soon as a pirate collects a key,
he is able decrypt all valid ciphertexts

In Pirates 2.0

Do not assume perfect decoders and classical tracing may fail
Does it mean pirate decoders are useless? Not really, example:

- Pirate decoder can’t decrypt ciphertexts with headers $>$ 1 Go
- It can decrypt any ciphertext with headers of size $<$ 1 Go

NNL01: Subset Cover Framework

Idea

- To revoke a set R of users, partition the remaining users into subsets from some predetermined collection
- Encrypt for each subset separately

Framework

- Predetermined collection of subsets

$$S_1, S_2, \cdots, S_w \qquad (S_i \subseteq \mathrm{N})$$

- Each subset S_j is associated with a long-lived key L_j
- A user $u \in S_j$ must be able to derive L_j from its secret information I_u

NNL01: Subset Cover Framework

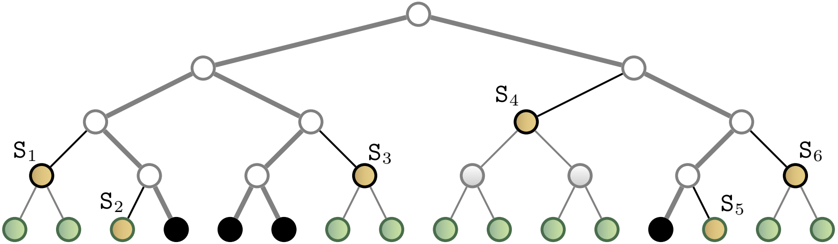
Encryption

- Given a revoked set R, the non-revoked users $N \setminus R$ are partitioned into m disjoint subsets $S_{i_1}, S_{i_2}, ..., S_{i_m}$

$$N \setminus R = \bigcup S_{i_j}$$

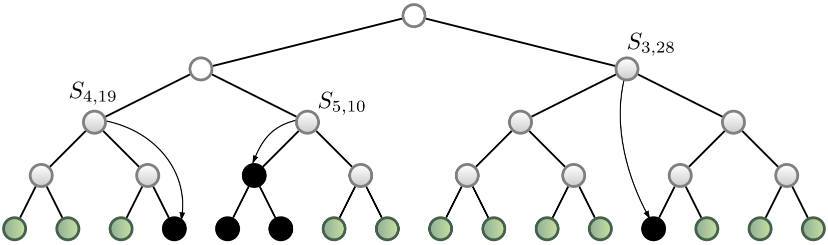
- a session key K is encrypted m times with $L_{i_1}, L_{i_2}, \cdots, L_{i_m}$.

Defining Subsets: Complete Subtree



Each subset at node *i* contains all leaves in the subtree of node *i*

Defining Subsets: Subset Difference



Each subset corresponds to a pair of nodes (i, j), where j is in the subtree rooted at i

$S_{i,j}$ contains all leaves in the subtree of node i but NOT in the subtree of node j

General Attack Strategy against Subset-Cover

Main Idea

Select a collection of subsets $S_{x_1}, \ldots, S_{x_t}$ such that:

- The number of users in each subset S_{x_k} is large
  $\Rightarrow$ the anonymity level of the traitors is guaranteed

General Attack Strategy against Subset-Cover

Main Idea

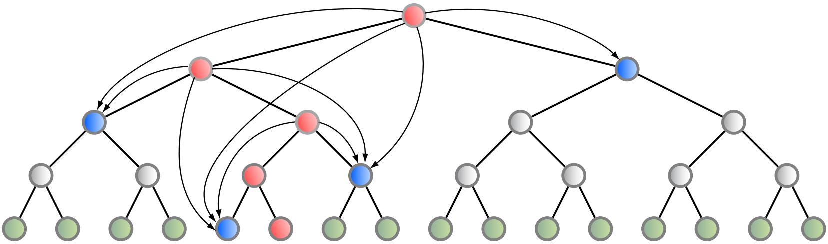
Select a collection of subsets $S_{x_1}, \ldots, S_{x_t}$ such that:

- The number of users in each subset S_{x_k} is large
  $\Rightarrow$ the anonymity level of the traitors is guaranteed
- For any set R of revoked users and any method used by the broadcaster to partition

$$N \setminus R = S_{i_1} \cup \cdots \cup S_{i_m}$$

  the probability that one of the subsets S_{x_k} belongs to the partition $S_{i_1}, \ldots, S_{i_m}$ is high

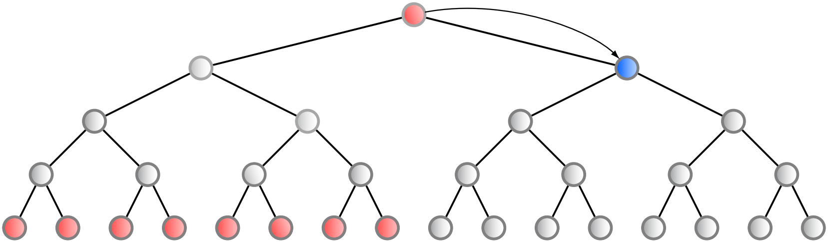
Subset Difference: Key Assignment



Key Assignment

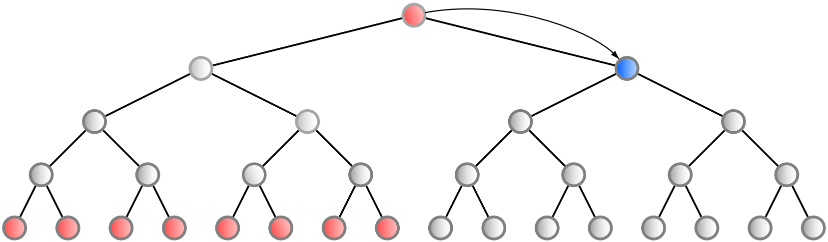
- Red: all nodes on the road from the user to the root
- Blue: all node hang-off the red road
- Label: from a red node to blue nodes in the subtree rooted at the red one

Remark on Key Assignment



- Red: all nodes on the road from the user to the root
- Blue: all node hang-off the red road
- Label: from a red node to blue nodes in the subtree rooted at the red one

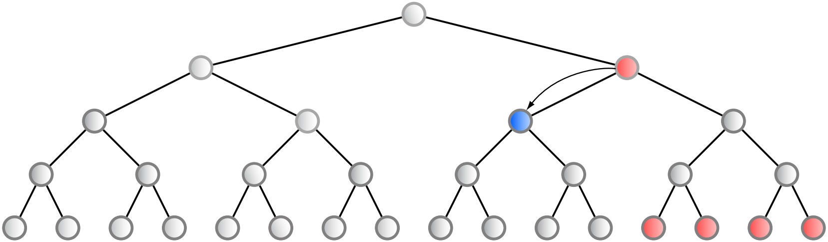
Pirates 2.0 against to Subset Difference



Strategy of Pirates 2.0

- Fix some level ρ

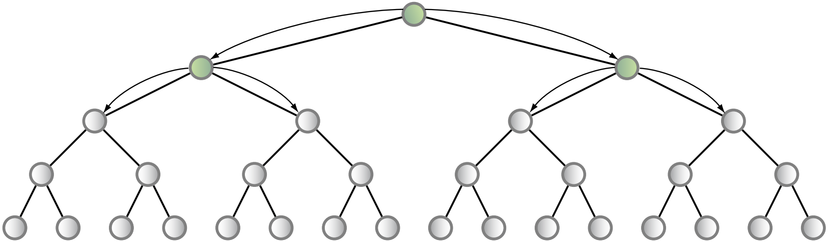
Pirates 2.0 against to Subset Difference



Strategy of Pirates 2.0

- Fix some level ρ
- A traitor only contributes a label $L_{i,j}$ when:
  - i is below or at level ρ
  - j is a direct descendant of i
- A revoked user can also contribute!
  Helps maintaining a high level of anonymity for contributors

Pirates 2.0 against to Subset Difference



Strategy of Pirates 2.0

- Fix some level ρ
- A traitor only contributes a label $L_{i,j}$ when:
  - i is below or at level ρ
  - j is a direct descendant of i
- A revoked user can also contribute!
  Helps maintaining a high level of anonymity for contributors

Broadcaster's Strategy

Lower bound for the number of subsets

- The broadcaster should use subsets $S_{i,j}$ where i is below ρ in order to thwart Pirates 2.0
- Each subset $S_{i,j}$ covers less than the number of leaves in the subtree rooted at i, i.e., less than $N/2^{\rho}$ users

Broadcaster's Strategy

Lower bound for the number of subsets

- The broadcaster should use subsets $S_{i,j}$ where i is below ρ in order to thwart Pirates 2.0
- Each subset $S_{i,j}$ covers less than the number of leaves in the subtree rooted at i, i.e., less than $N/2^{\rho}$ users
- To cover $N \setminus R$ users, the broadcaster has to use at least $2^{\rho}(N - R/N)$ subsets
- If there is less than half of the users revoked, the number of subsets to be used is greater than $2^{\rho-1}$

A Concrete Example

In the classical setting, covering 2^{32} users

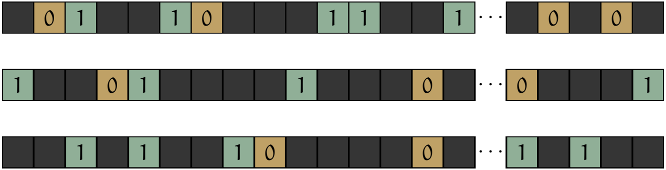
- A set of $\rho \log(\rho)$ randomly chosen traitors can decrypt all ciphertexts of rate less than $2^{\rho-1}$
- Anonymity level for each traitor: $2^{32-\rho}$

A Concrete Example

In the classical setting, covering 2^{32} users

- A set of $\rho \log(\rho)$ randomly chosen traitors can decrypt all ciphertexts of rate less than $2^{\rho-1}$
- Anonymity level for each traitor: $2^{32-\rho}$
- $\rho = 10$: 10000 traitors (1000 in adaptative attacks) can decrypt all ciphertexts with headers of size less than 128 Mb
- Each traitor is guaranteed an anonymity level of 2^{22} (each traitor is covered by 4 millions users)

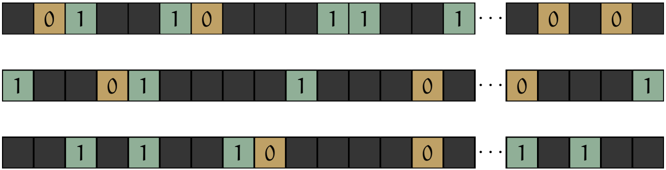
Pirates 2.0 against Code Based Schemes



Main idea

Each user only contributes its sub-keys at some positions

Pirates 2.0 against Code Based Schemes

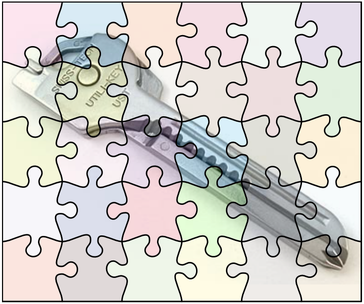


Example for Tardos' Code

For a 30-collusion secure code with 2^{32} users

- about 100000 traitors
- mount a Pirates 2.0 attack, each traitor would be masked by thousands of users

Conclusion: Variations on Pirates 2.0



Open problems

- Modification of tree-based and code-based schemes resisting to Pirates 2.0
- Pirates 2.0 attacks against algebraic schemes?