Hiding Data Sources in P2P Networks

Marek Klonowski, Mirek Kutyłowski, Bartek Różański

Wrocław University of Technology

The 4th International Workshop for Applied PKI (IWAP'05)

Supporting Access to Crucial Data

- **specialized servers**
  - expensive
  - attacking a few servers may block the whole system
- **P2P distributed solutions**
  - cheap
  - resistant to attacks?

P2P Design Highlights

- primary goals: fair load sharing, data consistency
- anonymity and security received less interest

P2P Design Highlights

- primary goals: fair load sharing, data consistency
- anonymity and security received less interest

a P2P server holding crucial data can be attacked

Solution Idea

1. keep data on dedicated, but **hidden** server(s)
2. provide **access** to the server(s) **through anonymous paths** that start at known P2P addresses
3. let the paths **self-evolve** for resistance against traffic analysis

Some Applications

- key servers (like PGP)
- blacklists
- whitelists
- peer ranking in P2P networks

Blacklisting

- allows exclusion of unfair peers/users
- incentives for fair, cooperative behavior

Blacklisting – Existing Solutions

- “black records” on P are stored by node $H(P)$, where H is a secure hash function
- every network node can fetch blacklist information on P from node $H(P)$
- location of black records on P is known, so P can mount an attack towards $H(P)$ and “clean” or block records on P
- such attacks are quite realistic

Tools

- *universal re-encryption*
- a special kind of *onions*

Universal Re-Encryption 1/5

- based on ElGamal
- and a cyclic group G of order q with generator g, where discrete logarithm problem is hard

Standard ElGamal

- pick k, $0 < k < q$, at random
- compute $r := g^k$ and $s := m \cdot y^k$
- (s, r) is a ciphertext of m

Universal Re-Encryption 2/5

Ciphertext Re-Encryption

- everybody can re-encrypt message, no private key knowledge required
- **an external observer cannot check if C' is a re-encrypted version of C' for given ciphertexts C and C'**

Universal Re-Encryption 2/5

Ciphertext Re-Encryption

- everybody can re-encrypt message, no private key knowledge required
- **an external observer cannot check if C' is a re-encrypted version of C' for given ciphertexts C and C'**

Re-Encryption of (r, s)

- pick k' at random
- $r' := r \cdot g^{k'}$
- $s' := s \cdot y^{k'}$
- (r', s') is a valid ciphertext of m

Universal Re-Encryption 3/5

Modification: URE (Golle, Jakobsson, Juels, Syverson)

- knowledge of public key unnecessary for re-encryption
- control of ciphertext integrity

URE Encryption

- pick k_0 and k_1 at random
- URE-ciphertext of m:
  $(\alpha_0, \beta_0; \alpha_1, \beta_1) := \left(m \cdot y^{k_0}, g^{k_0}; y^{k_1}, g^{k_1}\right)$,
- obviously:
  - (α_0, β_0) encrypts m
  - (α_1, β_1) encrypts 1

Universal Re-Encryption 4/5

Re-Encryption

- choose k_0', k_1' at random
- $\alpha_0 := \alpha_0 \cdot \alpha_1^{k_0'}$
- $\beta_0 := \beta_0 \cdot \beta_1^{k_0'}$
- $\alpha_1 := \alpha_1^{k_1'}$
- $\beta_1 := \beta_1^{k_1'}$

Universal Re-Encryption 4/5

Re-Encryption

- choose k_0', k_1' at random
- $\alpha_0 := \alpha_0 \cdot \alpha_1^{k_0'} = m \cdot y^{k_0} \cdot y^{k_1 \cdot k_0'} = m \cdot y^{k_0 + k_1 \cdot k_0'}$
- $\beta_0 := \beta_0 \cdot \beta_1^{k_0'} = g^{k_0} \cdot g^{k_1 \cdot k_0'} = g^{k_0 + k_1 \cdot k_0'}$
- $\alpha_1 := \alpha_1^{k_1'} = y^{k_1 \cdot k_1'}$
- $\beta_1 := \beta_1^{k_1'} = g^{k_1 \cdot k_1'}$

Universal Re-Encryption 5/5

Decryption by Multiple Parties

A ciphertext of form:

$$E_{x_1,x_2,\ldots,x_\lambda}(m) = \left(m \cdot (y_1 y_2 \ldots y_k)^{k_0}, g^{k_0}; (y_1 y_2 \ldots y_k)^{k_1}, g^{k_1}\right)$$

can only be decrypted by the set of nodes with private keys x_1, $x_2, \ldots, x_k$ corresponding to $y_1, y_2, \ldots, y_k$ respectively.

$$E_{x_1,x_2,\ldots,x_\lambda}(m) = \left(m \cdot g^{k_0 \sum_{i=1}^{\lambda} x_i}, g^{k_0}; g^{k_1 \sum_{i=1}^{\lambda} x_i}, g^{k_1}\right)$$

partial decryption:

$$\left(m \cdot g^{k_0 \sum_{i=1}^{\lambda} x_i}\right) / (g^{k_0})^{x_1} = \left(m \cdot g^{k_0 \sum_{i=2}^{\lambda} x_i}\right)$$

URE-Onions 1/4

- regular onion encoding m to be sent along a random path $J_1, J_2, \ldots, J_\lambda$:

$$E_{J_1}(E_{J_2}(\ldots(E_{J_\lambda}(E_{J_D}(m), D), J_\lambda)\ldots), J_3), J_2)\ .$$

(E_Z denotes public key encryption aimed for user Z)

- an URE-onion is built from λ ciphertexts called *blocks*:
  - the ith block (for $1 \leq i < \lambda$) has the following form:

$$E_{x_{J_1}+\cdots+x_{J_i}}(J_{i+1})\ .$$

  - the last block:

$$E_{x_{J_1}+\cdots+x_{J_\lambda}}(m)\ .$$

URE-Onions 2/4

Properties of Onions

- each server can see only the previous and the next hop on the path
- a passive eavesdropper cannot derive any information of messages processed through the network

URE-Onions 3/4

Routing

- first, the onion is sent to J_1
- J_1 partially decrypts and re-encrypts all onion blocks: each $(\alpha_0, \beta_0; \alpha_1, \beta_1)$ is replaced by

$$\left(\frac{\alpha_0}{(\beta_0)^{x_1}}, \beta_0; \frac{\alpha_1}{(\beta_1)^{x_1}}, \beta_1 \right) .$$

and then re-encrypts the result at random.

URE-Onions 4/4

Routing

- J_1 can now read the next destination – J_2
- the fully decrypted block is not removed (for hiding the path position)
- blocks are permuted at random
- the result is sent to J_2

Navigators

- URE-ciphertext of message 1 is called a *navigator*

$$(\alpha_0, \beta_0; \alpha_1, \beta_1) = \left(y^{k_0}, g^{k_0}; y^{k_1}, g^{k_1}\right)$$

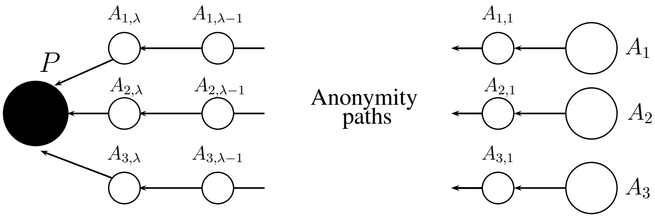
- navigator can be treated as some kind of envelope: any node can insert a message into it, by multiplying the first element of quadruple by a message m to be sent

Hiding Data Sources 1/4

- the protocol guarantees anonymity of data holders without preventing access to information
- instead of direct requests for x users now contact one of *access points* $A_1, \ldots, A_k$, with addresses derived from values $H(x,1), \ldots, H(x,k)$

Hiding Data Sources 2/4

Access Structure



- access points do not store x, but are connected via anonymity paths leading to node $P = P(x)$ storing data on x
- paths are based not on real addresses but on random identifiers (like for TOR)

Hiding Data Sources 3/4

Access Structure

- for each access point a path consisting of λ nodes $A_{i,j}$ for $1 \leq j \leq \lambda$ is chosen at random.

Hiding Data Sources 3/4

Access Structure

- for each access point a path consisting of λ nodes $A_{i,j}$ for $1 \leq j \leq \lambda$ is chosen at random.
- each $A_{i,j}$ stores its secret key $d_{i,j}$ and a navigator for communication with A_i
- each access point of x has a navigator for communication with $P(x)$

Hiding Data Sources 4/4

Request for x:

- U sends a request for x to an arbitrary access point A_i
- A_i uses a navigator obtained from $P(x)$, it inserts the request and the ID of U into the navigator
- the message is processed towards $P(x)$
- after arrival of the navigator server $P(x)$ sends information on x to U via an anonymous channel

Access Path Evolution 1/3

Traffic Analysis

- an adversary can trace traffic and perform traffic analysis
- fixed paths may reveal locations of data sources
- to alleviate this problem paths evolution is introduced – during each period of time every intermediate node is replaced with probability β
- replacements are local and independent from each other

Access Path Evolution 2/3

Node replacement

- in each round a node $A_{i,j}$ initiates replacement procedure with probability β
- $A_{i,j}$ picks its replacement A'
- public key and respective navigators are updated to reflect a node replacement

Access Path Evolution 3/3

Node replacement details

- Connections $(A_{i,j-1}, A_{i,j})$ and $(A_{i,j}, A_{i,j+1})$ are changed to $(A_{i,j-1}, A')$ and $(A', A_{i,j+1})$
- $A_{i,j}$ informs A' about its key $d_{i,j}$.
  Key offset δ is chosen by A' and $d_{i,j}$ is replaced by $d' = d_{i,j} + \delta$. The update $y' = g^{\delta}$ of the public key is transmitted to $P(x)$ (in a tricky way)
- $P(x)$ sends to A_i an updated navigator

Resistance to Dynamic Adversary 1/5

Attack Scenario

- an adversary starts by tapping the access point $A_i = A_{i,0}$
- by analyzing the communication sent by $A_{i,j}$ the adversary finally discovers $A_{i,j+1}$.
- after some number of steps the adversary locates $P(x)$

Resistance to Dynamic Adversary 2/5

Countermeasure -Path Evolution

- the node currently tapped by the adversary may get replaced
- should this happen, the adversary has to backtrack to the preceding path node
- the preceding node may as well be replaced in the meantime, hence the adversary needs to proceed backwards until a proper path node is found

Resistance to Dynamic Adversary 3/5

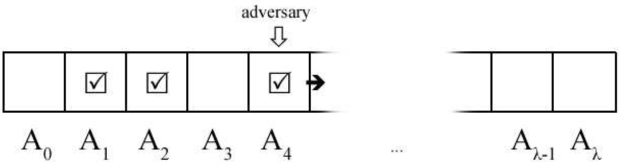
Attack Model - Weak Adversary- Assumptions

- the adversary performs a random walk on a path of length λ, starting from the leftmost point, aiming to reach the rightmost one
- during a round the adversary moves one step to the right with probability α

Resistance to Dynamic Adversary 4/5

Attack Model - Weak Adversary- Assumptions

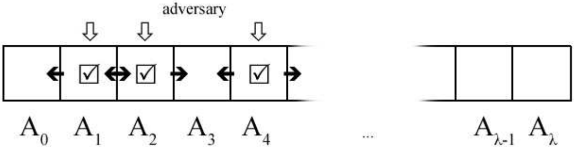
- each node visited by the adversary for the first time becomes *marked* (processing forward)
- during a round a marked node becomes unmarked with probability β (node replacements)
- if the node currently pointed by the adversary becomes unmarked, the adversary has to backtrack to the rightmost marked node



Resistance to Dynamic Adversary 5/5

Strong Adversary

- the difference is that the adversary marks the node next to a marked node with probability α (and not only at the last marked node as before)



Probability of Adversary's Success 1/5

		weak adversary			strong adversary		
path	rounds	$K = 2$	$K = 3$	$K = 4$	$K = 2$	$K = 3$	$K = 4$
10	50	23803	1806	230	50651	6403	712
10	100	45627	4271	531	82442	16426	2036
15	50	3631	63	3	17787	480	15
15	100	9204	147	7	47645	2193	62
20	50	556	2	0	4273	19	0
20	100	1594	4	1	22872	228	2

- simulation of 100.000 trials
- each attack was bound to 50 or 100 rounds
- path evolution probability $\beta = \frac{1}{2}$ at each round
- adversary's guessing probability $\alpha = \frac{1}{K}$

Probability of Adversary’s Success 2/5

Success Ratio Estimate

- if path length is 20 and the rate of path change is 2 times bigger than the advance rate of the adversary, then he succeeded for **none** of 100.000 trials to reach the end of the path within 50 steps– regardless of the adversary model.

Probability of Adversary's Success 3/5

Trajectories

Advances of adversaries at each round until the adversary must return to the start point:

experiment 1: 0 0 -1
experiment 2: 1 1 1 0 0 1 0 -5
experiment 3: 1 1 1 1 1 0 -1 1 0 1 0 -7
experiment 4: 0 0 0 -1
experiment 5: 0 0 -1
experiment 6: 1 1 1 1 0 -5
experiment 7: 0 1 0 1 0 0 0 0 -3
experiment 8: 1 1 1 0 -1 1 0 1 1 0 1 1 0 1 1 0 -2 0 0 -8
experiment 9: 0 0 0 0 0 0 0 0 -1
experiment 10: 1 1 0 -1 0 0 0 1 1 0 1 0 -1 0 -4

Probability of Adversary's Success 4/5

Exact distributions

- paths of length 8
- $\beta = 0.5$
- $\alpha = 0.20, 0.25, \ldots, 0.50$
- state transition matrices determined
- exact distributions computed for up to 32 rounds

Probability of Adversary's Success 5/5

Exact distributions

	α - pbb of advance by adversary						
rounds	0.20	0.25	0.30	0.35	0.40	0.45	0.50
20	0.000	0.001	0.003	0.008	0.017	0.031	0.051
24	0.000	0.002	0.005	0.012	0.024	0.043	0.069
28	0.001	0.002	0.006	0.015	0.031	0.054	0.085
32	0.001	0.003	0.008	0.019	0.037	0.065	0.102

Observations

- even for short paths of length 8 an adversary needs many rounds to raise the chance of reaching path end up to 0.1
- $\alpha = 0.3$ is sufficient to reduce the chance to 0.01
- if $\alpha = 0.5 \cdot \beta$ success ratio is only 0.030 for as much as 32 rounds

Thanks for your attention!