# Internal Attacks in Anonymous Publish-Subscribe P2P Overlays

Jörg Daubert[a] and Tim Grube[a] and Max Mühlhäuser[a] and Mathias Fischer[b]
[a]Telecooperation Group, Technische Universität Darmstadt / CASED, Germany
{daubert, tim.grube, max}@tk.informatik.tu-darmstadt.de
[b]Networking and Security Group, International Computer Science Institute, Berkeley, USA
mfischer@icsi.berkeley.edu

***Abstract*—Privacy, in particular anonymity, is desirable in Online Social Networks (OSNs) like Twitter, especially when considering the threat of political repression and censorship. P2P-based publish-subscribe is a well suited paradigm for OSN scenarios as users can publish and follow topics of interest. However, anonymity in P2P-based publish-subscribe (pub-sub) has been hardly analyzed so far. Research on add-on anonymization systems such as Tor mostly focuses on large scale traffic analysis rather than malicious insiders. Therefore, we analyze colluding insider attackers in more detail that operate on the basis of timing information. For that, we model a generic anonymous pub-sub system, present an attacker model, and discuss timing attacks. We analyze these attacks by a realistic simulation model and discuss potential countermeasures. Our findings indicate that even few malicious insiders are capable to disclose a large number of participants, while an attacker using large amounts of colluding nodes achieves only minor additional improvements.**

## I. Introduction

Online Social Networks (OSNs) such as Twitter and Facebook are evidently popular and gain more users every year [1]. OSNs play a major role in the distribution of news in areas where classic media is not reliable anymore due to political repression and censorship [2]. However, OSNs can also turn out as a powerful spying and persecution tool. Thus, anonymization tools play a major role not only for whistleblowers but also ordinary OSN users.

However, anonymization tools such as *Tor* [3], *MIXnets* [4], and *Crowds* [5] suffer from a variety of drawbacks such as traffic monitoring attacks [6] and bad performance due to abuse [7]. While current research focuses on traffic analysis, i.e., attacks performed by monitoring communication externally, only minor effort is put into analyzing internal attacks that actively exploit protocol features. Even so, internal attacks are very likely when considering compromised devices and key material. Timing attacks on anonymization system can be used to fingerprint and recognize devices and communication flows. Colluding malicious nodes that monitor traffic can be used to learn message flows in MIXnets [8]. With a more powerful global attacker, even the topology of the whole anonymization network can be learned [9].

Another form of anonymous content dissemination is anonymous pub-sub [10], [11], [12] that derives techniques from *Tor, MIXnets* and *Crowds*.

978-1-4799-5804-7/15/$31.00 

Publish-Subscribe (Pub-Sub) is a communication pattern that connects producers and consumers of information. Opposed to other communication patters, pub-sub routes information according to content rather than participant identifiers: a producer (*publisher*) generates a message (*notification*) and forwards it to the pub-sub system. Consumers (*subscribers*) express their interests in form of attributes (*subscription*) to the pub-sub system. The system then mediates notifications between publishers and subscribers. For example, Twitter applies such a pub-sub communication paradigm via hashtags. Anonymous pub-sub systems disseminate information by protecting the privacy of their publishers and subscribers. For that, sensitive messages such as notifications and subscriptions are encrypted and the anonymity of publishers and subscribers itself is protected. Thus, it prevents to link publishers and subscribers on the basis of a certain interest and thus attribute [13].

The main contribution of this paper is a detailed analysis of internal attacks on pub-sub anonymization systems. For that, we introduce a novel attacker model and assume an external attacker that has successfully deployed multiple malicious nodes within an anonymous pub-sub system. Hence, malicious nodes are in possession of all necessary cryptographic material to participate in the system, so that they are able to generate valid protocol messages. Malicious nodes are controlled remotely by the external attacker and collude to break the anonymity of other participants via timing attacks. We evaluate such timing attacks extensively within a simulation of an anonymous pub-sub system that was introduced by us in former work [12]. Our findings indicate that the introduced attacker can successfully de-anonymize participants, even with very few colluding malicious nodes. Finally, as the system under evaluation shares features with other systems and our attacker is generic, we believe that gained insights and results are likewise applicable to similar anonymous pub-sub systems like [11].

The remainder of this paper is structured as follows: Section II describes an internal attacker model for an anonymous pub-sub system. Section III proposes several internal attacks according to these models and evaluates the attack formally as well as via simulation. Section V then briefly proposes counter

measures to detect and mitigate these attacks and discusses costs and benefits of these counter measures. Section VI concludes our findings.

## II. Internal attacks on anonymity in Publish-Subscribe

We introduce anonymous pub-sub as a solution for anonymous OSNs, establish a generic system model, and complement it with a model for an anonymity attacker.

### A. Anonymous Publish-Subscribe

A distributed pub-sub system is based upon an overlay that determines the neighborhood relationship between participants and that employs forwarding rules for messages on top. There is a specific overlay for each attribut $a \in \mathcal{A}$ (interests). The graph $G := (V, E)$ denotes a basic overlay that comprises all participants $V := \bigcup_{a \in \mathcal{A}} \mathcal{P}_a \cup \mathcal{S}_a \cup \mathcal{F}_a$, where $\mathcal{F}_a := V_a \setminus (\mathcal{P}_a \cup \mathcal{S}_a)$. The set $\mathcal{P}_a$ denotes the publishers of $a$, which are initiating the dissemination of new information for interest $a$; the set $\mathcal{S}_a$ is the set of subscribers, which are interested in information about $a$. The nodes $\mathcal{F}_a$ are not interested in $a$ but rather another attribute $a'$, and thus contribute by relaying messages about $a$. An anonymous pub-sub system employs its functionality without the requirement of global identifiers for the nodes. An attribute mesh $\mathcal{M}_a := (V_a, E_a)$ describes the overlay network for an attribute $a$ with $V_a := \bigcup_{\forall p \in \mathcal{P}_a} \bigcup_{\forall s \in \mathcal{S}_a} path_a(p, s)$. The function $path_a(p, s)$ returns all nodes that a message traverses from $p$ to $s$ on the shortest path in $\mathcal{M}_a$. The set $N(v)$ defines the neighbors of node $v$ within $G$. $N_a(v)$ denotes neighbors in $\mathcal{M}_a$.

A centralized pub-sub system [14] consists of only one forwarder for all attributes ($|\bigcup_{\forall a \in A} \mathcal{F}_a| = 1$) and thus cannot protect anonymity against this node. P2P-based pub-sub systems may contain multiple forwarders. Some approaches [15], [16] require $\mathcal{P}_a$, $\mathcal{F}_a$, or $\mathcal{S}_a$ to know each other—either to establish $path_a(p, s)$ or to exchange key material—which violates publisher or subscriber anonymity. Hence, we focus on systems like [11], [12] that do not require knowledge about other participants to avoid coupling between nodes and thus anonymity violations. However, even with decoupled nodes, $\mathcal{F}_a$ are assumed to be "honest but curious" [17]. Therefore we analyze the influence of dishonest nodes.

We use our anonymous pub-sub system [12] that distributes advertisements upfront compared to the system [11] that distributes subscriptions upfront. The system works as follows: for the establishment of a pub-sub attribute overlay, a publisher in $\mathcal{P}_a$ distributes routing informations via advertisement messages $m_{ad}$ in $G$. Subscribers compare those with their attributes and upon a match join the overlay $\mathcal{M}_a$ by answering with a subscription message $m_{su}$. The advertisements $m_{ad}$ are flooded into $G$, the subscription messages $m_{su}$ are returned along the path on which the $m_{ad}$ was received. An important property of the pub-sub system is the request/response semantic of the messages, a $m_{su}$ subscription message is a response to a $m_{ad}$ advertisement message, the $m_{unad}$ unadvertisement message and the $m_{notif}$ notification message are potential responses on a $m_{su}$ subscription.

Messages in such a pub-sub system have to include some information to optimize the overlay in the absence of global topology information and node identifiers, for instance a hop-counter. Furthermore, since flooding can cause loops, the system also needs a mechanism to detect loops. For example, in former work [12] we proposed an anonymous pub-sub system that uses hash chains for loop prevention and overlay optimization.

### B. Attacker model

An attacker breaks anonymity by identifying publishers and subscribers given a certain attribute. The attacker constructs a node set $\mathcal{S}'_a$ and tries to assign the nodes of $\mathcal{M}_a$ to achieve $\mathcal{S}'_a = \mathcal{S}_a$ to break the subscriber anonymity and constructs $\mathcal{P}'_a$ to break the publisher anonymity, respectively. We focus on subscriber anonymity throughout the remainder of this paper as publisher anonymity can be attacked analogously.

An attacker can have different views on an attribute mesh. Either it has a global or local view on the pub-sub system. An attacker with global view observes all communication that is taking place but cannot decipher encrypted messages, while a local view attacker deploys malicious nodes in the system that can decipher their incoming messages. The view of this attacker is composed of all local views of these nodes. We denote $C_a$ as the set of malicious nodes that an attacker has successfully deployed within a mesh for attribute $a$.

Furthermore, the attacker can be either passive or active. An active attacker can interact with the system with the capabilities of a valid participant and is in possession of the necessary cryptographic material for overlay $\mathcal{M}_a$. Thus, it can delay, alter, replay, drop, and insert new, valid messages. In this paper, we focus on an attacker that is capable to deploy malicious publishers in the system, so that $\mathcal{C}_a \cup \mathcal{P}_a \neq \varnothing$. We use an attacker that only deploys publishing nodes, since these capabilities exceed subscribing nodes for the task of identifying subscribers.

Within the context of this paper, we focus on an active attacker with global topology knowledge that successfully deployed several malicious nodes in the system.

### C. Information Gain as Metric

The attacker gains information on $\mathcal{S}_a$ during his attack. To asses the effectiveness of the attack, we introduce two metrics.

First, we calculate the relation of the estimated size of the subscriber set $|\mathcal{S}'_a|$ and the real size $|\mathcal{S}_a|$. Using the ratio $|\mathcal{S}'_a|/|\mathcal{S}_a|$, we have a measure similar to the *k-anonymity* [18]. The attacker tries to reduce the candidate set $\mathcal{S}'_a$ towards the real subscriber set $\mathcal{S}_a$. Obviously, $\mathcal{A}$ can exclude $\mathcal{C}_a$, so $\mathcal{S}'_a$ is initialized with $V_a \setminus \mathcal{C}_a$. A high value shows the inefficiency of the attack, the members of $\mathcal{S}_a$ are hidden in a large *anonymity set*, the overlay.

However, since the first metric is coarse and does not allow a detailed statement about the membership to $\mathcal{S}_a$ per node in the overlay, we build a probability map to describe the

information gain of the attacker in more detail. The probability map contains the probability of each node in $V_a$ being part of $\mathcal{S}_a$. The probability of a single node $v \in V_a$ of being part of $\mathcal{S}_a$ is described as follows: The probability is 0 if $v \in \mathcal{C}_a$, as the node obviously can be excluded since it is controlled by the attacker itself, if $v \notin \mathcal{C}_a$, the probability is the $|\mathcal{S}_a|$ divided by the number of remaining candidates $|V_a| - |\mathcal{C}_a|$. Using this probability, the sum of all probabilities is equal to the number of subscribers. To build the probability map, we normalize the probability of each node by dividing it by $|\mathcal{S}_a|$. Hence, for each node in $V_a$, except the adverse ones, we use the same initial probability and compose a map $\mathbf{v}$ of values as given by Equation 1.

$$\forall v \in V_a : \mathbf{v}[v] = \begin{cases} 0 & if \quad v \in \mathcal{C}_a \\ \frac{|\mathcal{S}_a|}{(|V_a|-|\mathcal{C}_a|)*|\mathcal{S}_a|} & else \end{cases} \tag{1}$$

$$\sum_{v_i \in V_a} \mathbf{v}[v_i] = 1 \tag{2}$$

Constraint 2 ensures that the probabilities remain valid. The attacker adjusts the map $\mathbf{v}_s$ in every attack step $s$. We calculate the entropy $H_s$ to quantify the information gain $g_s$ of the attacker in every attack step. Equation 4 defines the entropy based upon the difference between the initial and the current probability per node. The information gain $g_s$ is defined as difference of the entropy of the preceding attack step $s-1$ and the entropy of the current attack step $s$, shown in Equation 5.

$$p_{\text{diff}}(v_i, s) = 1 - |\mathbf{v}_0[v_i] - \mathbf{v}_s[v_i]| \tag{3}$$

$$H_s = -\sum_{v_i \in |V_a|} p_{\text{diff}}(v_i, s) * log_2(p_{\text{diff}}(v_i, s)) \tag{4}$$

$$g_s = H_{s-1} - H_s \tag{5}$$

Hence, an attacker attempts to minimize $H$ by maximizing $g$ in each step, which can be accomplished by eliminating nodes from $\mathbf{v}_s$ (setting their probability value to 0). The elimination of nodes is possible if the attacker can estimate the position of subscribers in the overlay $\mathcal{M}_a$ and therefore exclude nodes from begin members of $\mathcal{S}_a$.

## III. Analysis of internal attacks

In this section, we analyze the strengths and weaknesses of the active local attacker for breaking participant anonymity. In particular, we determine how accurate the attacker can identify subscribers $\mathcal{S}_a$ within a mesh for a given attribute $a$. The following subsections introduce a timing attack in anonymous pub-sub, explain how the attacker obtains necessary delay information, and how the attack is carried out.

### A. Timing attack in anonymous Publish-Subscribe

An attacker that successfully infiltrated the attribute mesh for $a$ by $x$ malicious nodes can exploit the request/response semantic of the system to identify the members of the subscriber set $\mathcal{S}_a$. For that, the attacker uses publishing nodes to send advertisements that trigger response messages from other nodes, and then uses these responses as well as information about the message flow, such as round-trip time (RTT), to learn about subscribers. For attacking subscriber anonymity in P2P-based pub-sub, the attacker uses the RTT between advertisement and subscription to estimate the delay to the closest subscriber. Given this delay in between a malicious node and a subscriber as well as an estimate for the average node-to-node delay (cf. Section III-B), the attacker can approximate the number of nodes between its malicious publishers and the subscriber. A malicious node can repeatedly perform this attack for each of its neighbors to find more subscribers and to rule out forwarders. More malicious nodes controlled by the attacker provide additional neighbors over which the attack can be performed.

The attacker observes the RTT between sending an advertisement and receiving a subscription as in Equation 6. The symbol $d$ denotes the distance to the subscriber $v_s$, i.e., $d = |path(v_0, v_s)|$, and is yet unknown.

$$\overbrace{\delta(m^x_{ad}, m^x_{su})}^{\text{observable}} = \overbrace{\sum_{j=0}^{d-2} \delta(m^j_{ad}, m^{j+1}_{ad})}^{\text{advertisements}} + \overbrace{\sum_{j=d-2}^{0} \delta(m^{j+1}_{su}, m^j_{su})}^{\text{subscriptions}} \tag{6}$$

Function $\delta(m^x_w, m^y_v)$ measures the time in between sending message $m_w$ from node $v_x$ and receiving $m_v$ at node $v_y$. To derive the subscriber distance $d$ from the observed $\delta(m_{ad}, m_{su})$, the attacker furthermore requires an average delay estimate, which is given by Equation 7. However, the attacker cannot observe this delay directly. Thus, we provide methods to obtain such an estimate in Section III-B.

$$\delta_{avg} = \frac{\sum_{j=0}^{d-2} \delta(m^j_{ad}, m^{j+1}_{ad})}{d} \tag{7}$$

Given the left hand sides of Equations 6 and 7, the attacker approximates $d$ as in Equation 8. The division by 2 is necessary as $\delta(m_{ad}, m_{su})$ describes the RTT but the attacker requires the one-way node distance.

$$d \approx \frac{\delta(m_{ad}, m_{su})}{2 \times \delta_{avg}} \tag{8}$$

With the node distance $d$ and background knowledge of $G$, the attacker can rule out nodes closer than $d$, assign high probability to nodes at distance $d$, and leave nodes further away than $d$ as undecidable. Hence, the attacker adjusts his initial probability map $\mathbf{v}_0$ (cf. Equation 1) to Equation 9 such that the gained knowledge is reflected and Constraint 2 remains satisfied.

$$\mathbf{v}[v_x] = \begin{cases} 0, & \text{if } v_x \text{ closer than } d \text{ or } v_x \in \mathcal{C}_a \ (j \text{ times}) \\ \frac{\lambda}{(k+l)}, & \text{if } v_x \text{ at distance } d \ (k \text{ times}) \\ \frac{\phi}{(k+l)} & \text{if } v_x \text{ further than } d \ (l \text{ times}) \end{cases} \tag{9}$$

The Equation 9 shows how the attacker can eliminate $j$ out of $|V_a|$ nodes that are closer than $d$ hops on the shortest path in basic overlay $G$. Hence, all remaining ($k$ at distance $d$, $l$ at distance greater than $d$) nodes get a higher probability. To distinguish nodes at distance $d$ (one or more subscribers) from nodes with greater distance (zero or more subscribers), we use a boost factor $\lambda \geq 1$ and a reduce factor $\phi \leq 1$ to model this difference. Simulations with our model (cf. Section IV) indicate that $\lambda = 1.187$ results in the best information gain; $\phi$ has to be solved in each step to satisfy Constraint 2. As the sum over all probability map entries remains constant over all attack iterations, the attacker attempts to maximize $j$ as nodes once ruled out remain ruled out.

### B. Estimating the average delay

The average delay $\delta_{avg}$ is crucial for the attacker to calculate the node distance to the closest subscriber. However, the attacker cannot observe or calculate $\delta_{avg}$ directly. Existing delay approximations such as [19] use specific protocols such as DNS that do not incorporate the application layer delays of an anonymous pub-sub system such as cryptography. Hence, we propose the following approximations:

*a) Direct Neighbor Extrapolation:* The attacker can use mechanisms of the basic overlay $G$ and send messages to direct neighbors, e.g., a heartbeat, and use the responses to calculate the average delay. Using neighborhood relations, any node $v \in V$ can approximate the average delay in $G$ as given by Equation 7.

However, this method is biased by the connectivity of such a node itself. Hence, a bad connection would cause high delays for all neighbors.

*b) Request Round-Trip:* As an alternative to the previous approach, the attacker can exploit a time to live (TTL) feature of a protocol in a (partially) two-connected basic overlay: the attacker sends a message that propagates to one neighbor and receives it later on via another neighbor. As anonymous pub-sub systems may use flooding for advertisement or subscription dissemination, it is likely that the attacker will receive a forwarded message back again after some time. He can then approximate the average delay via the elapsed time and the TTL difference. For example, within our anonymous pub-sub protocol, proposed in former work [12], the attacker can send out advertisements and will obtain TTL information by an attached hash value. Such hash values belong to hash chains that realize privacy-friendly TTL counters to avoid loops in attribute meshes.

The request round-trip approach incorporates more nodes and thus better approximates the average delay than direct neighbor extrapolation. However, highly connected basic overlays lead to short cycles and thus few traversed nodes.

*c) Collusion:* Similar to initiating round-trips, an attacker that successfully deployed two malicious nodes in the topology can exploit the TTL feature for deriving the average overlay delay. For that, one malicious node sends a message that the second node will eventually receive. With shared knowledge in terms of send and receive time, as well as a TTL feature, the attacker can approximate the average delay.

This approach reflects an average path in the basic overlay and does not suffer from short cycles. However, a delay anomaly on the single path between controlled nodes may influence the average delay calculation.

### C. Attack procedure

After the attacker has obtained background knowledge and average delay, he can exploit protocol features and perform a timing attack. The attacker needs to have an estimate on the average delay $\delta_{avg}$. The attacker uses a subset $N(v)' \subseteq N(v)$ of the neighborhood of its malicious nodes $v \in C_a$ for this attack. The usage of the complete neighborhood of $v$ is possible but could render $v$ suspicious and empower countermeasures. For every neighbor in $N(v)'$, the attacker will first un-advertise the attribute and then send a new $m_{ad}$ to the same neighbor. After sending the advertisement, the attacker waits for possible responses. The received $m_{su}$ messages will trigger the computation of the distance to the responding subscriber(s). Using this distance, the attacker is able to compute three subsets of nodes. First, the subset of nodes closer than the distance of the responding subscriber. These nodes are excludable since they can not be subscribers. Second, the set containing the nodes in the recognized distance. These nodes have an equal probability of being subscribers. The last set computed by the attacker contains all nodes further away than the recognized subscriber. As these nodes are located "behind" the subscriber(s), there is no statement about their membership in $\mathcal{S}_a$ possible. Using these information, the attacker updates its probability map $\mathbf{v}$ in every step. This map associates every node $v$ with its probability of being a subscriber. The attacker considers every node that exceeds the threshold as subscriber, i.e., $v_x \in \mathcal{S}'_a : \mathbf{v}_s[v_x] > \mathbf{v}_0[v_x]$.

Algorithm 1 summarizes the full attack. We use the example as given in Figure 1 to explain the attack for an attacker that successfully deployed one malicious node $v_0$ in an attribute mesh. Furthermore, we have $|V| = 9$ nodes and $|\mathcal{S}_a| = 2$ subscribers. The attacker has complete topology knowledge on $G(V, E)$ and thus can order nodes according to their distance in terms of overlay hops, which is indicated in the figure by rings. Furthermore, the attacker is in possession of a delay estimate and sets up its initial probability map $\mathbf{v}_0$ as shown by the node annotations in Figure 1.

The attacker first sends $m^0_{ad}$ via $v_0$ to neighbor $v_1$, but does not receive any response (Algorithm 1, lines 1-4). Thus, the attacker sets the values of $v_1$ and $v_4$ in $\mathbf{v}_0$ to zero (lines 11-12) and builds a more precise $\mathbf{v}_1$ as shown in Figure 2, i.e., $j = 2$, $k = 0$, $l = 7$. In the second step, the attacker sends $m^0_{ad}$ from $v_0$ to neighbor $v_2$ and observes $\delta(m^0_{ad}, m^0_{su}) = 4 \times \delta_{avg}$ (lines 4-5). Hence, the closest subscriber must be two hops away and the attacker adjusts $\mathbf{v}_1$ to $\mathbf{v}_2$ as shown in Figure 3, i.e., $j = 3$, $k = 2$, $l = 4$.

In this third and last step the attacker performs two operations: first, he sends $m^0_{unad}$ to $v_2$ as he received his $m^0_{ad}$ via $v_3$ again—hence both nodes are connected and therefore $v_3$

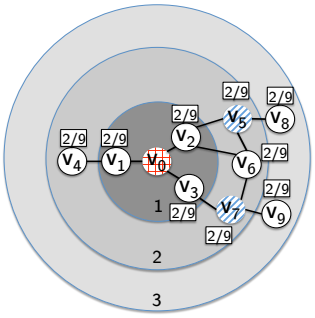


Fig. 1: Initial probability distribution assuming $|\mathcal{S}_a| = 2$ and $|V| = 9$.

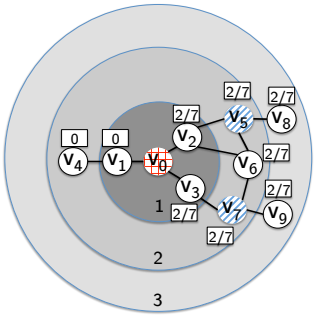


Fig. 2: First attack step via neighbor $v_1$. No response, hence probabilities become 0.

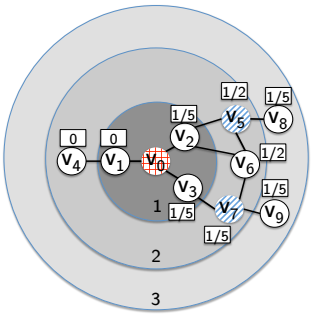


Fig. 3: Second attack step via neighbor $v_2$. Response from distance 2.

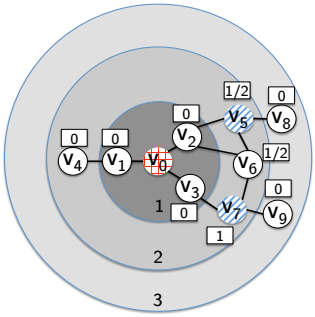


Fig. 4: Third attack step via neighbor $v_3$. Response from distance 2. Node $v_7$ as well as either $v_5$ or $v_6$ are subscribers. Hence $v_8$ and $v_9$ are out.

already knows the advertised topic and would discard the new advertisement for this topic. Second, the attacker sends $m^0_{ad}$ to neighbor $v_3$ and again observes $\delta(m^0_{ad}, m^0_{su}) = 4 \times \delta_{avg}$. The only node within the range of 2 hops must be $v_7$ and is therefore a subscriber. Hence the attacker adjusts the values of $v_3, v_7$, and $v_9$ in $\mathbf{v}_2$ to 0 and 1, respectively, as shown in Figure 4, i.e., $j = 3$, $k = 3$, and $l = 2$.

## IV. Evaluation

In this section, we analyze the influence of the attackers' capabilities as well as the influence of the topology of the basic overlay on the attack success. In particular, we phrase the following research questions:

1) What is the influence of the connection delay of the basic overlay on the attackers ability to determine the distance to the subscriber?
2) How much does the connectivity of malicious nodes $\mathcal{C}_a$, i.e., the number of neighbors, influence the information gain?
3) Does the distance between colluding nodes $\mathcal{C}_a$ influence the information gain?
4) How much do additional controlled nodes, i.e., more collusion, improve the information gain?

**Algorithm 1:** attack$(G, c)$

```
// Initialization
foreach v ∈ N(c) do
    send(m_unadv, c, v)
    send(m_adv, c, v)
    d ← ∞
    if response_received then
        d ← δ(m_adv, m_sub) / (2·δ_avg)
    k ← candidatesAt(d)
    l ← candidatesFurther(d)
    foreach u ∈ branch(v) do
        d' ← |path(c, u)|
        if d' < d then
            v[u] ← 0
        else if d' = d then
            v[u] ← λ · |S'_a| / (k+l)
        else if d' > d then
            v[u] ← φ · |S'_a| / (k+l)
```

### A. Simulation setup

To evaluate the attack, we implemented a simulation model for the *OMNeT++*[1] discrete event simulator with the *INET*[2] framework.

We assume a fixed basic overlay for our experiments. For that, we use scale-free networks generated according to the power law out degree algorithm [20] with recommended parameters $\alpha = -0.08$ and $\beta = 4.5$ to create basic overlays with an average diameter of $5-6$ and node degree $|N(v)| \approx 6.875$, and simulate with parameters and metrics as summarized in Table I.

We assume that no prior attribute overlay for $a$ exists as malicious nodes in $\mathcal{C}_a$ may spoof messages to disassemble the overlay before each attack step. If not noted otherwise we simulate with $|V| = 200$ nodes. The message delay for edges $E$ is randomly chosen from a uniform distribution $\mathcal{U}(1ms, 20ms)$. Furthermore, we repeat each experiment 25 times with different random seeds ($runs$) and calculate 95% confidence intervals. We assume a strong attacker (cf. Section II-B) that knows the topology of the basic overlay $G$ as well as the subscriber ratio $|S_a|/|V|$. Hence, given an attribute $a$ the attacker tries to establish a set $\mathcal{S}'_a$ as an approximation of the real subscriber set $\mathcal{S}_a$.

For the attack evaluation, we use the *information gain* (cf. Section II-C) as well as the path distance between nodes. The information *gain* is calculated as entropy delta between two succeeding attack steps (cf. Section II-C) and represents the overall success of the attack.

[1] http://www.omnetpp.org
[2] http://inet.omnetpp.org

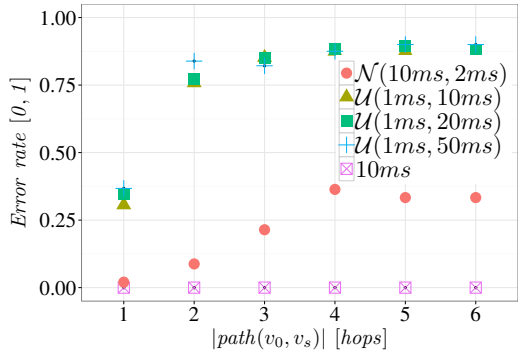


Fig. 5: Subscribe distance estimation error over increasing real distances.

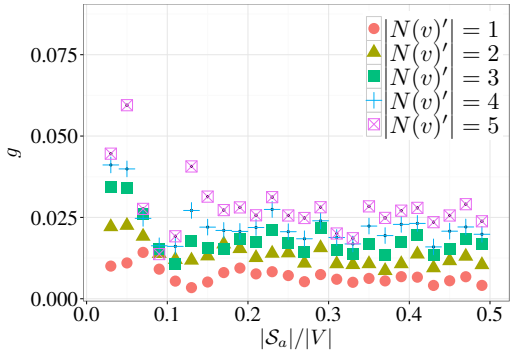


Fig. 6: Information gain over subscriber ratio each for the number of neighbors a malicious node uses for attacking.

| Parameters and Default Values | |
|---|---|
| $\|V\| = 200$ | // size of the basic overlay |
| $\|S_a\|/\|V\| \in [0, 1]$ | // ratio of subscribers |
| $\|\mathcal{C}_a\| \in [1, 5]$ | // number of colluding malicious nodes |
| $\|\mathcal{C}_a\|/\|V\| \in [0.005, 0.025]$ | // ratio adverse to genuine nodes |
| $\|\mathcal{A}\| = 1$ | // number of attributes in the system |
| $\|N(v)'\| \in [1, 5]$ | // neighbors used by attacker |
| $\delta = \mathcal{U}(1ms, 20ms)$ | // delay of edges $E$ |
| $runs = 25$ | // repetitions per setup |

TABLE I: Simulation parameters.

### B. End-to-end delay

To analyze the influence of connection delay between nodes on the correct estimation of the subscriber distance, we variate the delay distribution as well as subscriber and attacker position, and let a single attacker determine the distance to the closest subscriber via each neighbor from $N_a(v)$. We use the settings as given in Table I, and variate the delay model $\delta$ with uniform $\mathcal{U}(left, right)$ and normal $\mathcal{N}(mean, variance)$ distributions in comparison to a static distribution with a delay of $10ms$.

We perform 200 repetitions per distribution setting with random placements each, and then compare the attackers distance estimation $|path(v_0, \widetilde{v_s})|$ with the real distance $|path(v_0, v_s)|$. The attacker uses the *Request Round-Trip* approach to approximate $\delta_{avg}$. We group the results by the real path length.

The static delay serves as reference and we expect a linear increase of the hop error rate for the normal distribution as the average accumulated variance should not exceed one edge delay for several hops. Compared, the uniform distributions should quickly approach the error rate of a random guess.

Figure 5 shows the error rates in dependence on the distance $|path(v_0, v_s)|$ in between $v_0$ and $v_s$ for the five delay distributions. As expected, the attacker well compensates normal distributed delay across the average basic overlay diameter of $\approx 6$ hops. The uniformly distributed delay already produces high error rates from the second hop onwards.

Concluding, delays from a narrow normal distribution, e.g., within a LAN, have only minor impact on the correct hop-count estimation. However, delays from a uniform distribution effectively prevent the attack. A uniform delay with a parameter range that covers the diameter of the basic overlay renders the attack infeasible.

### C. Attacker connectivity

To analyze the influence of the attackers' node degree on the information gain, we vary the number of neighbors the attacker uses. Furthermore, we use the settings as given in Table I, vary the ratio of subscribers $|S_a|/|V| \in (0, 0.5)$, and let the attacker successively use $|N(v_0)'| \in [1, 5]$ of his available $|N(v_0)|$ neighbors.

We expect that increasing $N(v)'$ causes a linear increase in information gain as the attacker can exploit more attack paths in case no close cycles exist.

Figure 6 shows the information gain per subscriber over the subscriber ratio for the different number of used neighbors. For an increasing subscriber ratio, the attacker gains slightly less as more subscribers on the same path mask each other. However, a second neighbor almost doubles the information gain.

Concluding, the attackers' additional advantage of good connectivity becomes smaller with every neighbor. Furthermore, large subscriber ratios render the attack harder.

### D. Distance between colluding nodes

To analyze the influence of the hop distance between colluding nodes, we use two malicious nodes $|\mathcal{C}_a| = 2$, variate the length of the path between these nodes, and compare the information gain they achieve together.

We use the settings as given in Table I, a low and high subscriber ratio $|S_a|/|V| = \{0.05, 0.15\}$, and compute the results over 300 repetitions of the experiment. For every experiment, we use two random nodes $v_x, v_y$ as adverse ones and group the experiments by the shortest path length $|path(v_x, v_y)|$ between the attackers. We measure the information gain after each attack over all neighbors.

We expect that the large subscriber ratio experiment yield to a lower information gain per subscriber as shown in the previous experiment. Furthermore, we expect a longer path between the attacking nodes to increase the information gain. Given two attacking nodes $v_x, v_y$ and a subscriber $v_s$, a longer

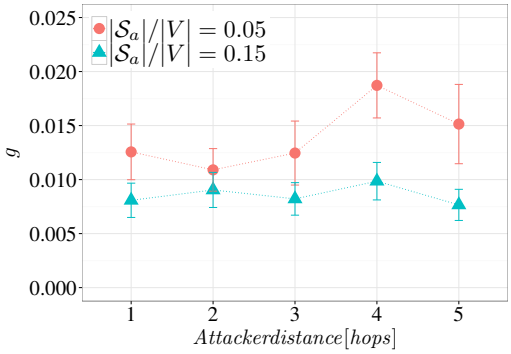


Fig. 7: Information gain over the distances of attacker $v_x, v_y$ for $|S_a|/|V| = \{0.05, 0.15\}$.

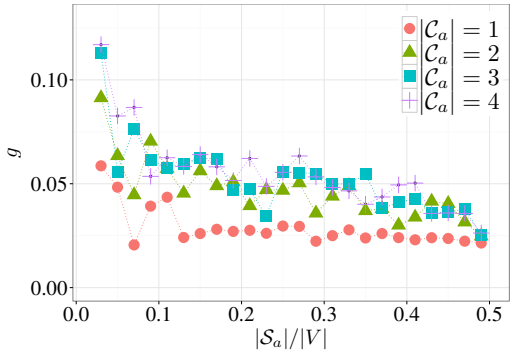


Fig. 8: Information gain of a colluding attacker over subscriber ratio. The additional gain drops with each additional node.

path $|path(v_x, v_y)|$ should increase the probability of pairwise distinct nodes in $path(v_x, v_s)$ and $path(v_y, v_s)$. Hence, the attacker gets more distance measurements to the subscriber and can rule out more forwarders.

Figure 7 shows the average information gain per subscriber over the distance between attacking nodes. An increasing distance between controlled nodes shows no significant improvement of the information gain. It appears that the attackers improve on distant subscribers, but loose on very close subscribers.

Concluding, the distance between colluding nodes, and thus the placement of the attackers' nodes, has only minor impact on the attack outcome. Additional simulations conducted by us with higher graph diameters indicate a slight decrease of the gain for large distances.

### E. Collusion of malicious nodes

To analyze the influence of colluding nodes, we vary the number of malicious nodes and measure the information gain the attacker achieves for different subscriber ratios. We use the settings as given in Table I, vary the subscriber ratio $|S_a|/|V| \in (0.0, 0.5)$, as well as the number of malicious nodes $|\mathcal{C}_a| \in [1, 4]$.

We expect that this experiment behaves similar to the experiment on attacker connectivity (cf. Section IV-C): additional nodes should improve the information gain as the delay approximation becomes more accurate and as the attacker can identify more close by subscribers accurately. However, we also expect that the additional gain becomes smaller for every additional attacking node.

Figure 8 shows the information gain per subscriber of colluding nodes over the subscriber ratio. The attackers $|\mathcal{C}_a| = 2$ clearly improves over $|\mathcal{C}_a| = 1$, however the addition gain between $|\mathcal{C}_a| = 3$ and $|\mathcal{C}_a| = 4$ becomes smaller. Furthermore, the gain of additional nodes for very low ratios of subscribers is relatively low. The scenario with two attacking nodes causes larger confidence intervals due to the varying distance between both nodes over the 25 runs each (cf. Section IV-D).

Concluding, many colluding nodes only benefit scenarios with high subscriber ratios. Furthermore, the additional gain decreases with every additional node.

## V. Countermeasures

We introduce counter measures against the internal attacker in this section. First we show how to reveal the existence of an internal attacker, second we briefly introduce approaches to preserve privacy in presence of the attacker.

### A. Detection of malicious behaviour

Attackers have to destroy the overlay $\mathcal{M}_a$ before each attack step to get optimal results. Nodes can detect this behavior in form of "flapping" neighbors. In our example, attacker $v_0$ sends an un-advertisement $m_{unad}$ first, attack via its first neighbor by sending $m_{ad}$, wait for results, and un-advertise again to continue with further neighbors. Hence, nodes in close vicinity of $v_0$ can detect a high frequency of advertisement and un-advertisements as potential attacker.

Another method to detect the presence of an attacker is the recognition of changed neighbor behavior. Assuming some local topology knowledge, e.g., gained from past behavior, a node can detect if an attacker sends messages to only one neighbor at a time. For instance, assuming two paths, $path_1 = [v_0, v_1]$ and $path_2 = [v_0, v_2, v_1]$, genuine node $v_1$ can detect attacker $v_0$ if it receives messages via one path but not via the second one. Furthermore, malicious nodes $\mathcal{C}_a$ are valid participants in the system, they can drop messages from other nodes. This can be detected likewise.

### B. Proactive countermeasure

Genuine nodes can take countermeasures to protect themselves from attackers, e.g., delaying messages before sending or forwarding them.

Nodes can introduce random delay before forwarding messages to disturb the attackers' delay estimation and thus the localization and de-anonymization of nodes. For instance a node $v_i$ may delay messages to its neighbor $v_{i+1}$ by a uniformly chosen time from $[a, b]$, so that $\delta'(m^i, m^{i+1}) = \delta(m^i, m^{i+1}) + \mathcal{U}(a, b)$. This delay may influence the attacker's estimation of

$\delta_{avg}$, and the attacker's distance calculation. Furthermore, the delay may even cause the selection of alternate paths, causing noise between attack steps.

### C. Combination of reactive detection and proactive prevention

Nodes can also prevent further attacks once an attacker has been identified by blacklisting nodes or by using a reputation system. Assuming a node has been identified as attacker due to flappy behavior, neighbors can blacklist this node and thus isolate it from $G$. However, the attacker could easily detect blacklisting and perform "slower" attacks to cause less flapping. Generalizing blacklisting, nodes could establish a distributed reputation system, e.g., via the anonymous pub-sub system itself, and drop messages from all nodes with insufficient reputation.

## VI. Conclusion

In this paper we present an internal attacker model for anonymous pub-sub systems, introduce a colluding internal attack in detail, and analyze benefits and shortcomings of colluding attackers.

Related work mostly tackles large scale traffic analysis and traffic watermarking to break anonymity in anonymous communication system. Complementary, we analyze an internal attack in which the attacker possesses secrets to forge new and valid packets, and breaks anonymity by analyzing responses. Furthermore, we address the myth of the power of collusion attacks by analyzing the benefits of collusion and comparing it to a single attacker.

Our results indicate that two colluding attacker gain approximately the same benefits as one attacker with a doubled neighborhood size. The results furthermore indicate that the additional gain of adding an attacker decreases with the number of controlled nodes, i.e., more nodes disclose more subscribers but are less economical.

Artifical delay works well as a counter measure, even against attackers with colluding nodes. A uniform delay distribution provides the best protection against the attacker; narrow distributions already perform very well. This counter measure however also degenerates the performance of the system in terms of total message delivery time and its capability to adapt quickly.

Future work will address more internal attack approaches, e.g., the exploitation of protocol features such as node identifier changes. Furthermore, we will analyze sophisticated counter measures in detail, e.g., the detection of attackers.

## Acknowledgment

This work was supported by the ICT R&D program of MSIP/IITP. [13-921-03-001, Development of Smart Space to promote the Immersive Screen Media Service] The authors furthermore would like to thank Marc André-Bär for his support.

## References

[1] M. Watanabe and T. Suzumura, "How social network is evolving?: a preliminary study on billion-scale twitter network," in *WWW (Companion Volume)*. International World Wide Web Conferences Steering Committee / ACM, 2013, pp. 531–534.

[2] G. Lotan, E. Graeff, M. Ananny, D. Gaffney, I. Pearce *et al.*, "The arab spring— the revolutions were tweeted: Information flows during the 2011 tunisian and egyptian revolutions," *International Journal of Communication*, vol. 5, p. 31, 2011.

[3] R. Dingledine, N. Mathewson, and P. F. Syverson, "Tor: The second-generation onion router," in *USENIX Security Symposium*. USENIX, 2004, pp. 303–320.

[4] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Commun. ACM*, vol. 24, no. 2, pp. 84–88, 1981.

[5] M. K. Reiter and A. D. Rubin, "Crowds: Anonymity for web transactions," *ACM Trans. Inf. Syst. Secur.*, vol. 1, no. 1, pp. 66–92, 1998.

[6] P. Winter and S. Lindskog, "Spoiled onions: Exposing malicious tor exit relays," *CoRR*, vol. abs/1401.4917, 2014.

[7] R. Jansen, P. F. Syverson, and N. Hopper, "Throttling tor bandwidth parasites," in *USENIX Security Symposium*. USENIX Association, 2012, pp. 349–363.

[8] B. N. Levine, M. K. Reiter, C. Wang, and M. K. Wright, "Timing attacks in low-latency mix systems (extended abstract)," in *Financial Cryptography*, ser. Lecture Notes in Computer Science, vol. 3110. Springer, 2004, pp. 251–265.

[9] T. Abraham and M. Wright, "Selective cross correlation in passive timing analysis attacks against low-latency mixes," in *GLOBECOM*. IEEE, 2010, pp. 1–5.

[10] M. A. Tariq, B. Koldehofe, A. Altaweel, and K. Rothermel, "Providing basic security mechanisms in broker-less publish/subscribe systems," in *DEBS*. ACM, 2010, pp. 38–49.

[11] A. Shikfa, M. Önen, and R. Molva, "Privacy and confidentiality in context-based and epidemic forwarding," *Computer Communications*, vol. 33, no. 13, pp. 1493–1504, Aug. 2010.

[12] J. Daubert, M. Fischer, S. Schiffner, and M. Mühlhäuser, "Distributed and anonymous publish-subscribe," in *Network and System Security*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, vol. 7873, pp. 685–691.

[13] C. Wang, A. Carzaniga, D. Evans, and A. L. Wolf, "Security Issues and Requirements for Internet-Scale Publish-Subscribe Systems," in *HICSS*. IEEE, 2002, pp. 3940–3947.

[14] C. Raiciu and D. S. Rosenblum, "Enabling confidentiality in content-based publish/subscribe infrastructures," in *Second International Conference on Security and Privacy in Communication Networks and the Workshops, SecureComm 2006, Baltimore, MD, Aug. 28 2006 - September 1, 2006*. IEEE, 2006, pp. 1–11.

[15] A. Shikfa, M. Önen, and R. Molva, "Privacy-preserving content-based publish/subscribe networks," in *Emerging Challenges for Security, Privacy and Trust, 24th IFIP TC 11 International Information Security Conference, SEC 2009, Pafos, Cyprus, May 18-20, 2009. Proceedings*, ser. IFIP Advances in Information and Communication Technology, vol. 297. Springer, 2009, pp. 270–282.

[16] W. Chen, J. Jiang, and N. Skocik, "On the privacy protection in publish/subscribe systems," in *Proceedings of the IEEE International Conference on Wireless Communications, Networking and Information Security, WCNIS 2010, 25-27 June 2010, Beijing, China*. IEEE, 2010, pp. 597–601.

[17] M. Nabeel, N. Shang, and E. Bertino, "Efficient privacy preserving content based publish subscribe systems," in *17th ACM Symposium on Access Control Models and Technologies, SACMAT '12, Newark, NJ, USA - June 20 - 22, 2012*, V. Atluri, J. Vaidya, A. Kern, and M. Kantarcioglu, Eds. ACM, 2012, pp. 133–144.

[18] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 5, pp. 557–570, 2002.

[19] P. K. Gummadi, S. Saroiu, and S. D. Gribble, "King: estimating latency between arbitrary internet end hosts," *Computer Communication Review*, vol. 32, no. 3, p. 11, 2002.

[20] C. R. Palmer and J. G. Steffan, "Generating network topologies that obey power laws," in *GLOBECOM*. IEEE, 2000, pp. 434–438.