The dangers of composing anonymous channels

George Danezis[1] and Emilia Käsper[2]

[1] Microsoft Research
[2] Google
gdane@microsoft.com, ekasper@google.com

Abstract. We present traffic analyses of two anonymous communications schemes that build on the classic Crowds/Hordes protocols. The AJSS10 [1] scheme combines multiple Crowds-like forward channels with a Hordes reply channel in an attempt to offer robustness in a mobile environment. We show that the resulting scheme fails to guarantee the claimed k-anonymity, and is in fact more vulnerable to malicious peers than Hordes, while suffering from higher latency. Similarly, the RWS11 [15] scheme invokes multiple instances of Crowds to provide receiver anonymity. We demonstrate that the sender anonymity of the scheme is susceptible to a variant of the predecessor attack [21], while receiver anonymity is fully compromised with an active attack. We conclude that the heuristic security claims of AJSS10 and RWS11 do not hold, and argue that composition of multiple anonymity channels can in fact weaken overall security. In contrast, we provide a rigorous security analysis of Hordes under the same threat model, and reflect on design principles for future anonymous channels to make them amenable to such security analysis.

1 Introduction

The design of anonymous communication channels is a well established field [4] with applications in election protocols, censorship resistance and support of free speech. Several proposed anonymous communications channels take advantage of specific networking layers, such as ISDN telephony [14], TCP [9], email [6] or ad-hoc networks [11]. Similarly, the AJSS10 [1] channel is crafted for use in hybrid mobile networks, where a local peer-to-peer network of mobile devices (using WiFi) is provided with wide area connectivity by a mobile (3G, GPRS) telephony provider.

The degree of security for an anonymity system comes down to the difficulty an adversary has in identifying the originators or intended receivers of messages. The security guarantees can be formalized by probabilities over senders or receivers [19], or by equivalence classes of possible actors (anonymity sets) [13]. A probability distribution over all possible actors yields more fine-grained guarantees than a division of the actors into equivalence classes, and probabilistic modelling is thus also the approach we take in this paper.

The capabilities of adversaries are represented by different threat models against which the security of an anonymity system must be evaluated, such

as the global passive adversary [12], a fraction of malicious insiders [16], or specific corrupt entities. All schemes discussed in this paper aim to protect the identity of the sender from a malicious receiver as well as malicious insiders, while the RWS11 [15] scheme takes a step further by attempting to guarantee the anonymity of the receiver.

Performance metrics—communication overhead and latency as well as reliability—are important for anonymous communications as for any other networking primitive. Crucially, low performance of the communication channel leads to low usability of the system, which might in itself reduce anonymity [8]. Attacks on performance combined with retransmission mechanisms to ensure reliability may in fact lead to loss of anonymity [2].

Our contribution in this paper comprises three parts. We start by reviewing the classic Crowds [16] and Hordes [17] systems. We augment the security analysis of Hordes by proving new analytic bounds on sender anonymity in the case of a malicious receiver controlling a subset of nodes in the network. We proceed by studying the security and performance of two schemes that build on Crowds, AJSS10 [1] and RWS11 [15], and in both cases, demonstrate how the composition of multiple anonymous channels results in weaker anonymity as well as poorer performance. Finally, we reflect on our findings to provide a set of sanity checks for protocol designers. The key intuition behind our results is that multiple anonymous channels in general compose poorly: one cannot automatically argue that since they each separately provide some degree of anonymity, their composition will provide at least the same level of assurance. Further, while we are able to analytically bound the security of Crowds and Hordes, seemingly simple changes to those protocols make their modelling intractable. We argue that building anonymous communications channels whose security is easy to verify analytically should be a key design consideration for future proposals.

2 Crowds and Hordes

Crowds [16] uses a peer-to-peer network (a crowd) to pass messages anonymously from the sender to a receiver outside the crowd. A crowd member wishing to send a message first passes it to a random crowd node. Each subsequent node then flips a (biased) coin to decide whether to send the message to the destination (with probability p) or pass it to another crowd node. The latency of the channel, that is, the average number of hops a message travels in the crowd before being forwarded to the final destination is $1/p$.

The forward path of a Crowds message can also be used to anonymously receive replies. Crowds' bidirectional communication, however, is not robust in a mobile environment where nodes join and leave the crowd dynamically. Hordes [17] provides a solution by combining the forward path of Crowds with a multicast reply channel. In Hordes, the sender appends to the message a random session identifier as well as a *multicast group*—a subset of k crowd nodes that includes the sender herself. The outgoing message is then sent via a regular Crowds channel. The reply message, which must also include the session

identifier, is sent directly to the multicast group acting as an anonymity set. The original sender, who is part of this group, can use the session identifier to detect the reply, while the remaining members of the group will simply drop the message. The crowd forward latency of Hordes is still $1/p$, while the fast direct reply channel means that the latency of a round trip drops from $2/p$ to $1/p$.

2.1 Security analysis

Throughout this paper, we measure sender anonymity in terms of probabilities linking the message to its sender. While it is generally prudent to consider the worst-case confidence an adversary has in the true sender, taken over all possible observations of messages, we shall shortly see that for the protocols scrutinized in this paper, the worst-case confidence can be 1. Thus, we resort to measuring the expected success rate of the adversary guessing the sender of a message.

We say that a system offers perfect anonymity if for any observation, the adversary's best strategy is to choose at random from the entire set of n possible senders: $\Pr[\mathsf{success}] = 1/n$. Crowds provides the sender with perfect anonymity with respect to an adversarial receiver, since the receiver is equally likely to receive the message from any crowd member. Collaborating dishonest crowd members, on the other hand, can infer some information about the sender. More specifically, in a crowd of size n, a fraction $0 < f < 1$ of dishonest nodes can correctly guess the sender of a captured message with expected success rate [16]

$$\mathbb{E}(\mathsf{success}) = 1-(1-f)(1-p)+(1-f)(1-p)\frac{1}{(1-f)n} = f+(1-f)p+\frac{1-p}{n}\,. \quad (1)$$

Hordes provides k-anonymity of the sender w.r.t. the receiver: the receiver will only learn that the sender is one of the k members in the multicast group, $\Pr[\mathsf{success}] = 1/k$. Assuming communication between the sender and the receiver is encrypted such that intermediate nodes do not learn the multicast group, Hordes provides the same sender anonymity w.r.t. malicious crowd nodes as original Crowds.

Finally, we consider sender anonymity in the case of malicious crowd nodes collaborating with a malicious receiver. A malicious Crowds receiver contributes no additional information, implying that for messages captured in the crowd, the success of this adversary is still bounded from above by (1). The average success rate over all observed messages, including those captured by the receiver, is

$$\mathbb{E}(\mathsf{success}) = f + (1-f)\frac{1}{(1-f)n} = f + \frac{1}{n}\,. \quad (2)$$

For Hordes, this adversary model is not considered in the original paper, however, the simplicity of the protocol allows us to now devise a strict bound for sender anonymity in this model (see Appendix A for the proof).

Theorem 1. *In a crowd of n nodes, a fraction $0 < f = \frac{c}{n} < 1$ dishonest crowd members collaborating with a dishonest receiver can identify the sender of*

a Hordes message with average success rate

$$\mathbb{E}(\mathsf{success}) = f + \frac{1}{k}\left(1 - \frac{\binom{c}{k}}{\binom{n}{k}}\right) < f + \frac{1}{k}\,,$$

where k is the size of the multicast group chosen by the sender.

The effect of the security parameter k on the anonymity of Hordes is as expected: anonymity improves as k increases, and as k approaches n, the security of Hordes approaches that of Crowds (c.f. (2)). While the latency of a single message does not increase with k, a large multicast set causes heavier total load on the network, implying a natural security-performance trade-off.

Curiously, the Crowds parameter p does not influence the average success rate of the adversary; it does, however, affect the worst-case confidence the adversary has in the sender of a particular message: as the exit probability p approaches 1, fewer messages get captured before reaching the receiver, but those that do can be attributed to the sender with higher confidence, and vice versa. Finally, we observe that for $k \leq 1 + fn$, Hordes provides no anonymity in the worst case as a particularly poor choice of the multicast set can result in the entire set colluding with the receiver, thus pinpointing the single honest member in the set as the sender (see the proof in Appendix A for details).

2.2 Advantages and limitations

The security of Crowds and Hordes depends on the forwarding parameter p, as well as, in the case of Hordes, the size of the multicast group k. Both parameters have a negative impact on the latency of the protocol. In both cases, however, this relationship between latency and security is well understood and proven by rigorous security analysis. Indeed, the local randomized algorithm that determines the latency of a Crowds message provides optimal security [5].

On the other hand, both protocols are limited by only providing anonymity of the sender *with respect to the receiver*: neither can withstand attacks by even a passive network adversary that can eavesdrop communications in the crowd. Furthermore, the predecessor attack [21] can breach sender anonymity when forwarding paths are frequently resampled, and crowd paths should thus be fixed during a session between a sender and a receiver. We proceed to show that a failure to fully consider these limitations results in complex protocols that are harder to analyze, yet provide weaker anonymity.

3 The AJSS10 scheme

We present here the features of the AJSS10 channel relevant to its study in terms of security and performance. Full details are provided in the original work [1].

AJSS10 is designed to provide anonymity in a hybrid networking environment, where local devices can communicate with one another using a local Wifi

network, but need to communicate with the wider area network through a mobile telephony operator – this is an appealing model as it matches the capabilities of smartphones which now outnumber PCs in sales[3].

The AJSS10 protocol aims to achieve sender k-anonymity: the adversary should not be able to reduce the number of possible senders of a message to less than k participants. To construct her anonymity set, a sender using AJSS10 splits her message into k parts.[4] She sends one part directly to the operator and $k-1$ parts via other peers, using a variant of Crowds for each part. First, each message part is given to a random node in the local network; each receiving node flips a biased coin and with probability p sends the message to the operator *unless it has already sent a part of this message to the operator*; otherwise the message is sent to a random node in the network that repeats this process.

AJSS10 also attempts to conceal receiver anonymity from peers by using, for each message part, a different temporary identifier that only the operator can map to the recipient. Upon receiving a message part from a node, the operator thus looks up its destination address and forwards it to the receiving server. After all k parts have arrived, the server replies to the set of k peers (which, by design, includes the sender) and includes a server ID in that reply. The multicast reply channel of AJSS10 shares two properties with that of Hordes: the set of k peers aims to provide k-anonymity; while the inclusion of the original sender in this set guarantees robust reply message delivery in networks where peers join and leave dynamically.

The key difference between AJSS10 and Hordes routing is the use of multiple parallel paths on the forward channel, which has two effects. First, multiple paths are used to relay messages from the same sender to the same receiver—thus, we expect more opportunities for adversaries to perform traffic analysis. And second, the fact that the same node cannot send more than one part of the same message to the operator requires the inclusion of a message identifier visible to all peers, allowing them to link different parts of the same message together.

These changes also have repercussions on performance: all parts of the message need to be delivered for the message to be decoded, increasing protocol latency. Furthermore, some nodes will not be able to output a part as they have already delivered a previous part to the operator, forcing the message to continue its course within the network.

We note that the Hordes system assumptions must be satisfied to run the AJSS10 algorithm. Hordes relies on the sender directly choosing $k-1$ other peers to build a reply anonymity set and sending this set to the server. AJSS10 uses a variant of Crowds which implies that a client knows all other local peers in the crowd and can create these anonymity sets as in Hordes. Conversely, Hordes can also be used when peers join and leave the network dynamically, as the reply channel relies on a multicast that can safely fail for some peers. The question we

[3] `http://www.pcmag.com/article2/0,2817,2379665,00.asp`

[4] An estimate of dishonest peers can be used to choose a higher parameter to achieve k-anonymity, taking into account some of the potential members of the anonymity set may be under the control of the adversary.

ask next is thus whether AJSS10 offers significant security benefits that outweigh the additional performance cost compared to Hordes.

3.1 Security evaluation

We consider the security of the scheme against two threat models: a set of malicious peers in the wifi network, as well as a malicious operator that additionally controls a small fraction of peers.

The goal of the adversary in our analysis is to determine the most likely sender of a specific message for a set of observations of the anonymity network. The observations of malicious peers contain the identities of nodes that forwarded a message they wish to trace. In the second threat model, a malicious operator additionally records the identities of forwarding nodes for messages that do not get captured in the crowd. Our analysis relies on the following observations:

- Malicious nodes can link received message parts belonging to the same message together by their unique identifier. Similarly to the predecessor attack on dynamic Crowds paths [21], this results in an attack on sender anonymity, as the true sender will be observed on the paths of the message parts more often than any intermediate node. Even worse, while the predecessor attack relies on implicit identifiers such as user ID-s, cookies or other information at the application layer for linking messages together, these identifiers are explicitly included by design in AJSS10.
- A malicious operator cannot use this unique identifier as AJSS10 requires honest senders to strip the message parts of linkable information before giving them to the operator. Yet the server ID together with timing information may be sufficient in linking all parts back together in case of requests to relatively unpopular resources.
- Unlike Hordes, the receiver identity is initially concealed from peers; however, the server ID is also available in the reply message sent to the multicast group, allowing coalitions of malicious peers to restore the link between the message and its recipient. A successful attack on sender anonymity thus also results in sender-receiver linkability.

The state of the art in traffic analysis of anonymity protocols involves a full probabilistic modelling of the channel to extract posterior distributions over actions given the knowledge of an adversary [19]. Unlike Crowds, as well as Hordes which we modelled in Sect. 2.1, such an analysis is extremely complex for the AJSS10 protocol as modifications to the Crowds routing logic introduce temporal constraints. Instead, we provide an experimental upper bound on the security of the scheme by simulating the protocol and using heuristic analysis to decide upon the most likely sender of a message. The experimental bound is from above, as only partial information is used by our adversary to determine the sender (timing information is excluded), and even that partial information is not guaranteed to be used optimally.

Given a set of observed senders corresponding to parts of the same message, we simply pick as the most likely initial sender the peer that has sent the most

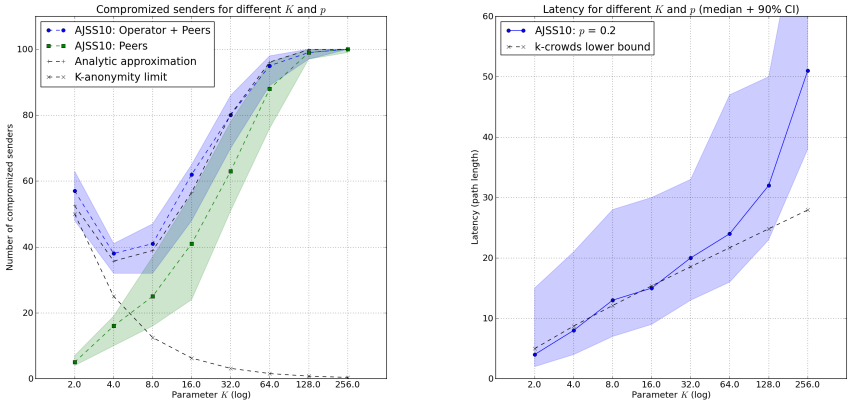

Fig. 1. The security and performance of AJSS10 versus the Hordes protocol.

parts. If the operator colludes with malicious peers, we further restrict the sender to be within the set of honest peers that forwarded the message to the operator, since the true sender is guaranteed to be in the k-anonymity set observed by the operator. In general, the higher the fraction of malicious peers in the network, the higher the probability that the actual sender of the message is observed multiple times. Surprisingly, the same is true for larger values of the security parameter k.

We can reason about the success probability of our heuristic algorithm as follows. Assuming peers collaborate with the operator, capturing at least one of the $k-1$ message parts sent via peers during its first hop is likely to identify the correct sender (who is then linked with at least two message parts); conversely, if no parts land in the hands of malicious peers during the first hop then the adversary's guess will effectively be a random choice from the anonymity set. In Appendix B, we estimate the success rate of a fraction $f = c/n$ corrupt peers collaborating with the operator to be

$$\mathbb{E}(\mathsf{success}) \approx 1 - (1-f)^{k-1} + \frac{(1-f)^{k-2}}{k}\left(1 - \frac{\binom{c}{k}}{\binom{n}{k}}\right).$$

Figure 1 (left) compares the anonymity provided by the AJSS10 scheme in the two threat models with the anonymity provided by Hordes in the stricter model when both operator and peers are malicious, denoted as the k-anonymity limit. Specifically, we consider a network of 500 peers out of which 25 are malicious. We perform 100 simulations for $p \in [0.1, 0.9]$ and plot the number of those experiments in which the actual sender was correctly identified. The solid lines are the median number of successes and the shaded regions represent the mini-

mum and maximum number of successes for different values of p. The analytic approximation for the security of AJSS10 is also plotted.

The complexity of the AJSS10 routing logic prevents us from accounting for the exact effect of the Crowds parameter p. We note that decreasing p increases latency, which results in more messages being captured before reaching the operator, thus on one hand increasing the number of different senders observed by peers, while on the other hand decreasing the size of the effective anonymity set of honest nodes observed by the operator. Our simulations confirm that varying p does not affect the security guarantees of AJSS10 significantly, while anonymity decreases rapidly as k increases, Even if only peers are malicious, even modest values of k lead to a greater probability of compromise than k-anonymity would suggest. In comparison, as k increases, the security of the Hordes channel increases together with the size of the anonymity set, as expected.

3.2 Performance evaluation

The modifications introduced by the AJSS10 scheme also have a serious impact on the latency of messages. Traditional Crowds latency follows a geometric distribution [5], which has a high variance. Requiring the delivery of $k-1$ message parts through the crowd in effect involves sampling $k-1$ random variables for the latency of each part, leading to the *maximum* delivery time being the latency of the whole message. The expected maximum latency of $k-1$ independent random variables following the geometric distribution with parameter p is

$$\mathbb{E}(l_{k-1,p}^{\mathsf{Geom}}) = \sum_{i=1}^{k-1} \frac{\binom{k-1}{i}(-1)^{i-1}}{1-(1-p)^i} \geq -\frac{H_{k-1}}{\log(1-p)}\,, \tag{3}$$

where H_{k-1} is the $(k-1)$th harmonic number, yielding a lower bound for the expected latency of an outbound message in the AJSS10 scheme. The fact that peers only deliver a single message further increases the delivery time and prohibits us from giving an *upper* bound to the latency.

Figure 1 (right) illustrates latency for AJSS10 with different parameters k and fixed $p = 0.2$ compared to Crowds/Hordes (median and 90% confidence intervals). It is clear for AJSS10 that as the parameter k increases, latency also increases to about an order of magnitude above Crowds. We also plot the theoretical lower bound (3) on the latency. As observed, assuming the AJSS10 system behaves like $k-1$ parallel Crowds is a good model for $k << n$ but path lengths increase significantly beyond this bound as k becomes larger, due to each peer being restricted to only sending out one message part.

4 The RWS11 scheme

The RWS11 scheme [15] also uses parallel Crowds paths, this time to conceal the identity of the receiver from the crowd. To send a message m to a receiver R_r, the sender first constructs a path $R_1, R_2, \ldots, R_{r-1}, R_r$ of crowd nodes (in

RWS11, the receiver is considered part of the crowd). She computes, for each node R_i on this path, a set of messages $s_i^{(0)}, s_i^{(1)}, \ldots, s_i^{(k-1)}$ and padding values such that

$$\begin{array}{lll} s_1^{(0)} & \oplus s_1^{(1)} \oplus \cdots \oplus s_1^{(k-1)} & = R_2 \,||\, s_2^{(0)} \\ s_2^{(0)} \,||\, \mathsf{pad}_1 & \oplus s_2^{(1)} \oplus \cdots \oplus s_2^{(k-1)} & = R_3 \,||\, s_3^{(0)} \\ & & \cdots \\ s_{r-1}^{(0)} \,||\, \mathsf{pad}_2 & \oplus s_{r-1}^{(1)} \oplus \cdots \oplus s_{r-1}^{(k-1)} & = R_r \,||\, s_r^{(0)} \\ s_r^{(0)} \,||\, \mathsf{pad}_{r-1} \oplus s_r^{(1)} & \oplus \cdots \oplus s_r^{(k-1)} & = \mathsf{null} \,||\, m\,. \end{array} \tag{4}$$

Padding is used to replace the removed address field, so that the size of the message remains constant throughout its course in the network. The padding values $\mathsf{pad}_1, \mathsf{pad}_2, \ldots, \mathsf{pad}_{r-1}$ are defined such that node R_i can compute $\mathsf{pad}_i = f(s_i^{(0)}, s_i^{(1)}, \ldots, s_i^{(k-1)})$ as a pseudorandom function of her shares.

The sender then forwards $s_1^{(0)}$ to R_1, as well as $s_i^{(1)}, s_i^{(2)}, \ldots, s_i^{(k-1)}$ for $i = 1 \ldots r$ to R_i, using Crowds for each share. The node R_i, upon receiving her k shares—one share from the previous node as well as $k-1$ shares from the sender—reconstructs $R_{i+1}||s_{i+1}^{(0)}$ and pad_i, and forwards $s_{i+1}^{(0)}||\mathsf{pad}_i$ to R_{i+1}, again using Crowds. The final receiver R_r, upon seeing null in the address field, thus knows that the message was intended for her.

The core idea behind the security of RWS11 is that all k shares $s_r^{(i)}$ intended for R_r are required to learn that R_r is the final receiver, and her identity thus remains hidden from malicious nodes who only intercept some of the shares. We proceed to show, however, that this construction significantly weakens sender anonymity, and further demonstrate active attacks on receiver anonymity.

4.1 Security evaluation

Sender anonymity. As before, we first consider the threat model where a malicious receiver R_r collaborates with a subset of malicious crowd nodes in order to learn the identity of the sender. Our observations on RWS11 are similar to those of AJSS10:

- Observing that $k-1$ of the k shares meant for R_r originate from the initial sender, we expect to see the true sender on the path to R_r more often than any other node.
- While RWS11 does not explicitly specify this, the k shares must include a common identifier that allows R_r to link the observed parts back together, making it easy for an attacker to distinguish shares belonging to the same message (*case 1*). In the absence of noise caused by other traffic, all kr shares can be linked together (*case 2*).

Again, if we assume that the adversary is successful with probability ≈ 1 in identifying the sender whenever she captures at least two message shares directly from the sender, we can predict the overall success rate to be

$$\mathbb{E}(\mathsf{success}) \approx 1 - (1-f)^{k-1} - (k-1)f(1-f)^{k-2} \text{ in case 1;} \tag{5}$$

$$\mathbb{E}(\mathsf{success}) \approx 1 - (1-f)^{r(k-1)+1} - (r(k-1)+1)f(1-f)^{r(k-1)} \text{ in case 2.} \tag{6}$$

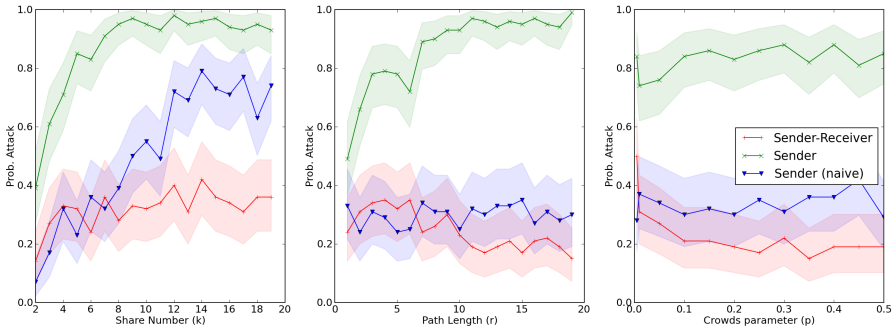

Fig. 2. Attack probabilities against the RWS11 protocols for different values of the security parameters. Green lines ("x" ticks) represent attacks against sender anonymity with a corrupt receiver; Blue lines (triangle ticks) represent attacks against a sender only using the shares sent to a corrupt receiver; Red lines ("+" ticks) represent attacks against an honest sender and receiver. (Shaded regions represent the 99% confidence intervals.

Sender-Receiver anonymity. Second, we assume the receiver is honest, and consider the security of the RWS11 scheme against a delaying adversary that is otherwise passive. Strictly speaking, this adversary is outside the scope of RWS11, which assumes only a purely passive adversary [15]. But an adversary that can merely delay any observed traffic and is otherwise passive is a realistic extension, thus instructive to consider. As before, we assume that message parts are linkable, either through a common header, or simply through lack of noise.

We note that an adversary that observes shares from a specific sender is left with the task of identifying the ultimate receiver of the message out of potentially r choices. We use causality, namely that R_{i+1} cannot output a message before receiving her share from R_i, and that the receiver R_r is the last in this chain, to attack the scheme.

An adversary delays all shares received within the crowd for a specific time frame. These messages are considered to be within the same, first, epoch. The adversary then releases the shares and observes the sequence of captured shares as the message continues its course through the sequential chain of intermediate nodes to R_r. This allows the adversary to build an ordering over the observed potential receivers of the target message. Any receiving peers observed before the last observation can be discarded as potential receivers.[5]

As before, the adversary again selects as a candidate sender, the sender that sent most messages within the delay period. The candidate receiver is selected

[5] The original protocol does not specify whether the r nodes must be distinct; for simplicity, we assume they are. The attack also works if repetitions are allowed, since message part identifiers allow us to consider each appearance of a node separately.

at random from the set of potential receivers observed, minus adversary peers and peers that were not last in the observed route.

Given the complexity of the attack, we simulate runs of the RWS11 protocol and perform the attack against them to evaluate its true success probability. This provides an *upper bound* on its security, as our heuristic adversary is, once again, not guaranteed to be optimal. The results are summarised in Figure 2 as the red line ("+" ticks). (Standard parameters used: 1000 peers including 10% corrupt; $k = 5$ shares; $r = 5$ path length; $p = 0.05$.) The figure illustrates the security of RWS11 as we vary its security parameters k, r and p.

Figure 2 also plots the success probability of the first attack with a malicious receiver. The green line ("x" ticks) assumes all message parts are linkable, while the blue line only uses the k shares destined for the corrupt receiver. As predicted by (5) and (6), the adversary is very likely to trace the sender for high k or r, while the average success rate is largely independent of p.

Broadly, increasing the security parameters introduced by RWS11 does not result in an significant increase in security. Surprisingly, we observe quite the opposite: as k and r increase, attacks become more successful. For no values of the security parameters does this probability becomes negligible.

Receiver anonymity. Finally, we should not exclude the possibility that some crowd nodes may actively misbehave, intercepting as well as modifying and injecting messages in the network. State-of-the-art packet formats for anonymous communications offer provable security against active attacks [7], and conversely, heuristic security claims may not cover design flaws leading to replay and oracle attacks that completely foil the purported security of the system [3, 20].

Lack of integrity protection in the RWS11 protocol opens up way to an oracle attack whereby any single corrupt node C on the path of any message part to R_i can determine whether R_i is the final receiver of the message. Namely, upon receiving a Crowds message $s_i^{(j)}$ meant for R_i, C sets $\hat{s}_i^{(j)} = s_i^{(j)} \oplus (C||\mathsf{null})$ and forwards the modified message $\hat{s}_i^{(j)}$ to R_i. By Eq. (4), upon receiving her k shares, R_i reconstructs

$$s_i^{(0)}||\mathsf{pad}_i \oplus s_i^{(1)} \oplus \cdots \oplus \hat{s}_i^{(j)} \oplus \cdots \oplus s_i^{(k-1)} = (R_{i+1} \oplus C)||(s_{i+1}^{(0)} \oplus \mathsf{null}),\ i+1 \neq r$$
$$s_i^{(0)}||\mathsf{pad}_i \oplus s_i^{(1)} \oplus \cdots \oplus \hat{s}_i^{(j)} \oplus \cdots \oplus s_i^{(k-1)} = (\mathsf{null} \oplus C)||(m \oplus \mathsf{null}),\ i+1 = r\,.$$

Thus, if R_i is the final receiver, the message m gets routed back to C, else the share $s_{i+1}^{(0)}$ gets routed to a random address $R_{i+1} \oplus C$. The node C, upon receiving m, will thus know that R_i was the true receiver[6]. C can easily distinguish the reply message m from an ordinary RWS11 message, for example by observing that it has no other matching parts.

Receiver anonymity is thus compromised whenever either R_{r-1} is corrupt, or at least one of the k Crowds messages $s_r^{(i)}$ meant for R_r gets captured by a corrupt node. From the analysis of Crowds, we know that the probability of a

[6] As a bonus, C will learn the contents of the message m.

single message being captured before reaching the receiver is $f/(1-(1-f)(1-p))$, so the success rate of the attacker is

$$\mathbb{E}(\mathsf{success}) = 1 - \left(1 - \frac{f}{1-(1-f)(1-p)}\right)^k \cdot (1-f) = 1 - \frac{p^k(1-f)^{k+1}}{(p+f-pf)^k}\,,$$

which approaches 1 quickly as k grows. In particular, the success probability of the active adversary is higher than the probability of a passive adversary learning the identity of the receiver in traditional Crowds by simple eavesdropping, so by attempting to protect receiver anonymity against the passive adversary, RWS11 in fact opens up an opportunity for a much more powerful active attack.

4.2 Performance evaluation

Similarly to AJSS10, the RWS11 scheme requires the delivery of k message parts for each hop through parallel Crowds channels. By eq. (3), the expected latency for R_1 to receive all k parts is $\mathbb{E}(l_{k,p}^{\mathsf{Geom}}) \geq -H_k/\log(1-p)$, and the expected latency of the channel is thus bounded from below by

$$\mathbb{E}(l_{k,r,p}^{\mathsf{RWS11}}) \geq -\frac{H_k}{\log(1-p)} + (r-1)\frac{1}{p}\,,$$

where $(r-1)/p$ is the expected latency of the $r-1$ sequential hops from R_1 to R_r. The bound is loose, as we ignore the delay effect of the remaining $k-1$ shares per hop that are delivered in parallel.

5 Design principles for anonymous channels

We have seen that as the parameter k increases, both the quality of protection and performance of the AJSS10 and RWS11 schemes deteriorate. This is true for both threat models considered, while the simpler Crowds and Hordes schemes provide higher security even in the stringent threat model where operator and peers collaborate. Following our analysis we draw a few conclusions regarding design principles for robust and secure anonymity systems.

Composition. Composing secure anonymous channels does not guarantee that the resulting channel will be secure. We have seen how k-anonymous multicast and Crowds on their own are secure, but running multiple instances of Crowds in parallel is in itself fragile. Leaking further information through the k-anonymous reply channel is significantly weaker than any of the channels on their own. Failure to account for this lead the designers of RWS11 to assume there is no need to analyse sender anonymity at all. The literature on predecessor attacks [21] and disclosure attacks [10] provides a guide to understanding how parallel composition of channels leaks information.

Security parameters. It is important to specify the security parameters and ensure that security increases as they increase. For instance, k-anonymous channels should provide better protection as k grows. Instead, we demonstrate

that the security of AJSS10 and RWS11 decreases significantly as the security parameter k increases. This counter-intuitive result was first observed for Crowds itself a decade ago [18]. Thus any scheme, especially any scheme building on Crowds, should be designed mindful of the possibility. In comparison, the parameter p, which is the traditional Crowds security parameter, has little effect on the security of the new schemes.

Security assumptions. It is crucial to distinguish the security-relevant state from incidental operational noise. Robust security analysis should assume that the adversary knows all security-irrelevant state. For example, the AJSS10 scheme does not disclose the message identifier to the operator, presumably in an attempt to keep separate message parts unlinkable. While this is prudent, it is a fragile security assumption, as the designers or users of the system have no way of ensuring that more than one message will be sent to a specific server. If only one message is sent to the server, then the identity of the server itself acts as an identifier that links the message parts. Similarly, RWS11 makes no explicit assumptions about delaying or mixing messages at the intermediate nodes, or the sender delaying messages. Thus, it is prudent to assume an adversary should be given information that links those messages together through timing when performing a security analysis. The assumption that the adversary is provided with all non-security-related information when attacking a system is common place in cryptology (through the use of artificial oracles), but not well established in the design of anonymity systems.

Threat modelling. When modelling the adversary, we must always consider the possibility that all malicious parties collaborate. Crowds-like peer-to-peer systems that attempt to protect the identity of the sender from the receiver as well as the crowd must thus provide protection against a malicious receiver controlling a subset of crowd nodes. Further, adversarial behaviour is unpredictable, and designs whose security collapses completely in the presence of an active adversary are too fragile for general purpose applications.

Ease of analysis. Schemes should be designed so that the security of the system can be analysed on the basis of a small amount of security state, assuming arbitrary values for the non-security relevant incidental operational noise. The AJSS10 mechanism makes such an analysis difficult: since a peer will never forward to the operator a second part of the same message, a race condition occurs. To analyse the performance and security of the system in an exact fashion one would need to introduce models of timing and network delays, and perform inference over different traces and timings of message transmissions. This is impractical, making the system difficult to analyse, without greatly improving its security.

Compare with simple designs. Finally, no anonymous channel is perfect, but some are better than others. For this reason it is important to compare new proposals with previous ones, making small modifications to existing protocols and comparing them all the time to ensure additional complexity in fact provides the advantages hoped for. For example the AJSS10 channel operational constraints, in terms of churn or knowledge of local peers, allow the application

of the simple Hordes protocol, and as such Hordes can be used as a baseline for evaluating its security.

Acknowledgments. The authors would like to thank the anonymous reviewers for their comments, as well as the paper shepherd Paul Syverson for great advice on improving the work.

References

1. C. A. Ardagna, S. Jajodia, P. Samarati, and A. Stavrou. Providing mobile users' anonymity in hybrid networks. In D. Gritzalis, B. Preneel, and M. Theoharidou, editors, *ESORICS*, volume 6345 of *Lecture Notes in Computer Science*, pages 540–557. Springer, 2010.
2. N. Borisov, G. Danezis, P. Mittal, and P. Tabriz. Denial of service or denial of security? In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 92–102, 2007.
3. G. Danezis. Breaking four mix-related schemes based on universal re-encryption. In *Proceedings of Information Security Conference 2006*. Springer-Verlag, September 2006.
4. G. Danezis and C. Diaz. A survey of anonymous communication channels. 2008.
5. G. Danezis, C. Diaz, E. Käsper, and C. Troncoso. The wisdom of crowds: attacks and optimal constructions. *Computer Security–ESORICS 2009*, pages 406–423, 2010.
6. G. Danezis, R. Dingledine, and N. Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *Security and Privacy, 2003. Proceedings. 2003 Symposium on*, pages 2–15, 2003.
7. G. Danezis and I. Goldberg. Sphinx: A compact and provably secure mix format. In *Proceedings of the 30th IEEE Symposium on Security and Privacy (S&P 2009), 17-20 May, Oakland, California, USA*, pages 269–282. IEEE Computer Society, 2009.
8. R. Dingledine and N. Mathewson. Anonymity loves company: Usability and the network effect. *Designing Security Systems That People Can Use. OReilly Media*, 2005.
9. R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th conference on USENIX Security Symposium-Volume 13*, pages 21–21, 2004.
10. D. Kesdogan, D. Agrawal, and S. Penz. Limits of anonymity in open environments. In F. A. P. Petitcolas, editor, *Information Hiding*, volume 2578 of *Lecture Notes in Computer Science*, pages 53–69. Springer, 2002.
11. J. Kong and X. Hong. ANODR: anonymous on demand routing with untraceable routes for mobile ad-hoc networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 291–302, 2003.
12. S. Murdoch and P. Zieliński. Sampled traffic analysis by internet-exchange-level adversaries. In *Proceedings of the 7th international conference on Privacy enhancing technologies*, pages 167–183, 2007.
13. A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymitya proposal for terminology. In *Designing privacy enhancing technologies*, pages 1–9, 2001.

14. A. Pfitzmann, B. Pfitzmann, and M. Waidner. ISDN-MIXes: Untraceable communication with very small bandwidth overhead. In *In Proceedings of the GI/ITG Conference on Communication in Distributed Systems*, 1991.
15. S. Rass, R. Wigoutschnigg, and P. Schartner. Doubly-anonymous crowds: Using secret-sharing to achieve sender- and receiver-anonymity. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, 7(4):25–39, 2011.
16. M. Reiter and A. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security (TISSEC)*, 1(1):66–92, 1998.
17. C. Shields and B. N. Levine. A protocol for anonymous communication over the internet. In D. Gritzalis, S. Jajodia, and P. Samarati, editors, *CCS 2000, Proceedings of the 7th ACM Conference on Computer and Communications Security, Athens, Greece, November 1-4, 2000*, pages 33–42. ACM, 2000.
18. V. Shmatikov. Probabilistic analysis of anonymity. In *CSFW*, pages 119–128. IEEE Computer Society, 2002.
19. C. Troncoso and G. Danezis. The bayesian traffic analysis of mix networks. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 369–379, 2009.
20. B. Westermann and D. Kesdogan. Malice versus an.on: Possible risks of missing replay and integrity protection. In *Proceedings of Financial Cryptography and Data Security (FC'11)*, February 2011.
21. M. K. Wright, M. Adler, B. N. Levine, and C. Shields. The predecessor attack: An analysis of a threat to anonymous communications systems. *ACM Trans. Inf. Syst. Secur.*, 7(4):489–522, 2004.

A Proof of Theorem 1

The intuition behind our analysis is this: if the sender forwards the message to a malicious peer during the first hop, the adversary can correctly attribute the message to the sender. In all other cases, the adversary's guess is random.

Adversary strategy. Given an observed message obs and a probability distribution $\Pr[\mathsf{Sender} = s_i|\mathsf{obs}]$ over all possible senders s_i, the optimal adversarial strategy is to choose the most likely sender by $\max_i \Pr[\mathsf{Sender} = s_i|\mathsf{obs}]$. If multiple senders are equally likely, there exists no better strategy than to choose one of them at random.

Let $\mathcal{H}$ be the subset of honest peers in the multicast group. Since the receiver is malicious, $\mathcal{H}$ is known to the adversary, and her complete observation is $\mathsf{obs} = (s_j, \mathcal{H}, c)$, where s_j is the honest node that forwarded the message and $c \in \{\mathsf{crowd}, \mathsf{recv}\}$ indicates whether the message was captured by a crowd node, or only at the receiver. We group all possible observations into three groups.

Type 1. The message is captured in the crowd, and the forwarding node s_j is a member of the multicast group $\mathcal{H}$: $\mathcal{O}_1 = \{(s_j \in \mathcal{H}, \mathcal{H}, \mathsf{crowd})\}$. In this case, we know from the analysis of Crowds that the adversary should pick s_j as the most likely sender.

Type 2. The message is captured in the crowd, but the forwarding node s_j is not a member of the multicast group $\mathcal{H}$: $\mathcal{O}_2 = \{(s_j \notin \mathcal{H}, \mathcal{H}, \mathsf{crowd})\}$. In this case, we know that s_j is *not* the sender. Thus, all senders $s_i \in \mathcal{H}$ are equally likely, and the adversary's best strategy is to guess at random.

Type 3. The message is captured at the receiver: $\mathcal{O}_3 = \{(s_j, \mathcal{H}, \mathsf{recv})\}$. Similarly to Case 2, all nodes in $\mathcal{H}$ are equally likely senders.

Success probability. With probability f, the sender forwards the message to a malicious peer during the first hop, thus generating an observation $\mathsf{obs} \in \mathcal{O}_1$ in the first set that leads to a correct guess. The remaining Type 1 observations, as well as all Type 2 and Type 3 observations lead to a guess that is correct with probability $1/|\mathcal{H}|$.

Noting that the multicast group contains the sender as well as $k-1$ nodes randomly chosen by the sender, and can thus include anywhere between 1 and k honest nodes, we can compute the success probability of the adversary as

$$\begin{aligned}
\mathbb{E}(\mathsf{success}) &= f + (1-f) \cdot \sum_{k'=1}^{k} \Pr[|\mathcal{H}| = k'] \frac{1}{k'} = f + (1-f) \sum_{k'=1}^{k} \frac{\binom{n-c-1}{k'-1} \cdot \binom{c}{k-k'}}{\binom{n-1}{k-1} k'} \\
&= f + (1-f) \frac{n}{k(n-c)} \sum_{k'=1}^{k} \frac{\binom{n-c}{k'} \cdot \binom{c}{k-k'}}{\binom{n}{k}} = f + \frac{1}{k} \left(1 - \frac{\binom{c}{k}}{\binom{n}{k}} \right) \\
&< f + \frac{1}{k} \,.
\end{aligned}$$

□

B Analysis of AJSS10

Assuming that the algorithm succeeds with probability ≈ 1 whenever peers capture a message on the first hop[7]; and makes a random guess from the subset of honest nodes $\mathcal{H}$ in the k-anonymity set otherwise, we can predict the success probability of our algorithm to be

$$\begin{aligned}
\mathbb{E}(\mathsf{success}) &\approx 1 - (1-f)^{k-1} + (1-f)^{k-1} \sum_{k'=1}^{k} \Pr[|\mathcal{H}| = k'] \frac{1}{k'} \\
&= 1 - (1-f)^{k-1} + \frac{(1-f)^{k-2}}{k} \left(1 - \frac{\binom{c}{k}}{\binom{n}{k}} \right) ,
\end{aligned}$$

where c is the number of corrupt peers, as before.

[7] The exact probability is analytically intractable, yet our simulations confirm that this assumption is reasonable.