Towards Language-Based Network Anonymity

Aslan Askarov Stephen Chong
Harvard University

May 17, 2013

Introduction Traditional systems for network anonymity are designed to be application agnostic. While this enables relatively simple deployment, many applications remain unaware of the anonymous nature of the underlying communication.

Because security properties are usually application-specific, we believe there is an opportunity to improve reasoning about anonymity guarantees by making applications aware of the anonymity of underlying communication channels. This paper reports on our work in progress on a language-based approach to this question. We distinguish between direct (or identifiable) and anonymous channels at a program source level. Combined with language-based techniques for secure information flow, our approach may lead to the specification and enforcement of strong confidentiality and anonymity properties.

A case for mixed communication model Reasoning about anonymity at the programming language level allows us to relax the underlying communication model, and allow direct identifiable communication. This mixed communication model may be a realistic alternative to often expensive all-anonymous communication.

For example, an instant messaging application may use both anonymous and direct communication depending on the level of trust it assumes for the other party. Sending an instant message to somebody located in the same building may be done directly and could avoid performance penalties of anonymous communication. However, anonymity may be necessary when communicating with potentially untrusted parties.

Similarly, mixed communication may be suitable for web mashups that include third-party content such as advertisements. Primary content of a mashup, e.g., a news front page, may be downloaded directly—this may be justified because the content provider is trusted—but the third-party content would be requested via anonymous channels.

Yet another example is an online purchase scenario that involves three principals: the client, the merchant, and the client's bank. Here, the client may want to contact the merchant anonymously, but is required to use a direct identifiable channel with the bank for verifiability.

Adversary model To reason about security guarantees, we start with a simple observational model which only includes the sender, the receiver, and the content of messages that are sent across network. Despite its simplicity, the model is sufficient to introduce some classes of adversaries. We discuss two examples of such adversaries here. *A local network adversary* $\mathsf{Adv}_{url}^{\mathsf{local}}$ passively observes messages sent (and received) by network node url, but is unable to track the recipients (or senders) of the messages when communication occurs over anonymous channels. Assuming that messages are adequately encrypted, this attacker may not extract useful payload from the content of the messages other than their existence. *A remote recipient adversary* $\mathsf{Adv}_{url}^{\mathsf{remote}}$ observes messages that are sent to network node url, but is unable to track the sender. Remote recipient adversary is assumed to be the intended recipient of the messages, and therefore may extract useful information from their content.

Language-based example To demonstrate our approach, we give an example of how anonymous communication may be used to enforce strong confidentiality property in a distributed auction application.

```
1  /* auction participants submit their bids in a few rounds */
2  for (int round = 0; round < n_rounds; round ++)
3    for (Participant p: participants)
4      /* request updated bid from each participant */
5      int b = p.requestBid(current_max);
6      if (b > current_max) /* update winning bid */
7           current_max = b;
8           winner = p;
9  /* contact winner to process payment */
10 winner.requestPayment(); /* may reveal winner identity */
```

Figure 1: Server code for distributed auction example

In this example, we assume that auction participants are public, but the bids themselves are confidential. Furthermore, the winner must remain confidential until some further notice. We consider a family of local adversaries $\mathsf{Adv}_p^{\mathsf{local}}$, who may observe network activity of any of the auction participants p, and the local adversary $\mathsf{Adv}_{server}^{\mathsf{local}}$, who observes the server's network activity.

Figure 1 provides a pseudo-code for the server side of this example. Auction participants are recorded in a collection `participants`. The auction runs in a number of rounds; during every round each participant is requested to update their current bid on line 5. Because auction participants are public, the communication that corresponds to this request may occur over direct identifiable channels, assuming the value of the bids themselves are encrypted. The winning participant is updated accordingly on line 8.

When the bidding is over, the winner must be contacted for payment. However, because this network communication may reveal identity of the winner it must be done over an anonymous channel.

Security guarantees We are currently designing a type-based program analysis for enforcing confidentiality and anonymity properties. The analysis distinguishes confidentiality and anonymity levels of information, and regulates how this information propagates to network channels. The analysis takes into account the associated attacker model. If we consider only local network attacker, but trust remote recipients, like in the example above, then identifiable information may be sent to anonymous channels. When we consider remote recipient adversary, identifiable and anonymous information must be isolated to prevent immediate linkability by the untrusted adversary.

For scenarios where we assume local network attacker, like in the auction example above, we expect that mixed communication model makes things "no worse" than if all-anonymous communication is used.

We also believe there is a potential for applications to help enforce additional anonymity beyond that provided by the network. For example, server can batch requests together, to provide "k-anonymity" like guarantees. That is, an application can help achieve anonymity guarantees that may not be possible at just the network level.

Future work In addition to pursuing the ideas outlined above, we are also interested in investigating the following questions.

- Which anonymity systems are most suitable for our language-based model.
- When does a mixed communication model require running in "router" modes (assuming TOR-like anonymous network), and what are the associated performance pros and cons.
- Would it be possible to synthesize A^3-like declarative anonymity specifications based on the program source.
- How to extend attacker models to include timing observations, and whether this may be connected to the work on predictive mitigation of timing channels.