DOVETAIL: STRONGER ANONYMITY IN NEXT-GENERATION INTERNET ROUTING

by

JODY MARK SANKEY

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

May 2013

Copyright © by Jody Mark Sankey 2013

All Rights Reserved

ACKNOWLEDGEMENTS

I would like to thank my supervising professor, Dr. Matthew Wright, for his frequent advice and keen insight. Without his exceptional teaching I would not have selected this field, and without his tutelage I could not have completed this thesis. I am grateful to Dr. Hao Che and Dr. Donggang Liu for their interest in my research and for taking the time to serve on my thesis committee.

I also wish to thank BAE Systems for the financial support they have provided through my studies, and the members of the Information Security lab at UTA for providing an open and welcoming environment.

I am indebted to all those who inspired and challenged me through the years: my father, my teachers, and those who took the time to mentor me at work. Their example is the light I always strive to follow.

Most importantly, I would like to express my deepest gratitude to my wife, Annie. Without her patience, kindness, and unconditional support I would not have made it through the last three years.

April 15, 2013

ABSTRACT

DOVETAIL: STRONGER ANONYMITY IN
NEXT-GENERATION INTERNET ROUTING

Jody Mark Sankey, M.S.

The University of Texas at Arlington, 2013

Supervising Professor: Matthew Wright

Current low-latency anonymity systems use complex overlay networks to conceal a user's IP address, introducing significant latency and network efficiency penalties compared to normal Internet usage. Rather than this obfuscation of network identity through higher level protocols, we recommend a more direct solution: a routing protocol that allows communication without exposing network identity, providing a strong foundation for Internet privacy, while allowing identity to be defined in those higher level protocols where it adds value. We propose *Dovetail*, a next-generation Internet routing protocol that provides anonymity against an active attacker located at any single point within the network. Key design features include the choice of many different paths through the network and the joining of path segments without requiring a trusted third party. We demonstrate the privacy and efficiency of our proposal by simulation, using a model of the complete Internet at the AS-level.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Our society has recently seen a dramatic increase in the extent to which daily life is conducted online, with socializing, shopping, learning, and banking via the Internet now an accepted norm rather than the exception. However, parallel advances in technology have enabled widespread tracking, storage, and cross correlation of our online activities, and business models have evolved for companies to monetize the user data they collect [1, 2]. Taken together, these factors mean that Internet privacy has become a pressing issue for today's society.

Privacy may be considered most easily in the negative sense, in terms of what identifying information is revealed and to whom. In particular, when we use the Internet, a wide range of identifying information is commonly revealed. One of the hardest identities to remove is that defined by the network routing protocol (*layer 3* in the OSI networking model [3]), since this identity is used to route and deliver data. In today's Internet, Internet Protocol (IP) is used as the layer 3 protocol, and this defines an IP Address that is globally unique[1] and present in every data packet. Recording a user's IP address can allow an adversary to uniquely identify her, link that identity with her actions, correlate multiple actions, and partially reveal her geographical and network locations.

Previous work has proposed *low-latency anonymity systems* to conceal a user's identity [4, 5, 6], including her IP address. Implementations have been developed for some of these proposals, and Tor in particular has been adopted by hundreds of thou-

[1]excepting the use of NAT with IPv4

sands of privacy-concious users worldwide[7]. However, current anonymity systems work by creating an overlay network on top of the layer 3 protocol, requiring a sequence of IP transmissions to disguise the original sender. This sequential forwarding and the processing required in intermediary nodes can incur substantial latency and network efficiency penalties.

We prefer an alternative formulation for this problem: Rather than attempting to conceal a global layer 3 identifier by adding substantial complexity to higher layers of the protocol stack, we believe the layer 3 protocol should not define a global identity, instead leaving identity management to higher layers in the protocol stack. In this way, we confine identity to those applications where it provides mutual benefit.

Privacy by itself is unlikely to motivate a change in the Internet routing protocol, but a range of additional concerns with IP have emerged within the networking field, including scalability, security, mobility, challenged environments, and network management [8], leading to major research initiatives investigating "clean slate" Internet designs [9, 10, 11]. A wide range of different routing concepts have already been proposed as a result of these activities [12, 13, 14, 15, 16, 17, 18]. Network virtualization research, showcased in testbeds such as GENI [19], offers hope for a progressive transition to a future routing protocol.

We believe privacy is a valid requirement for these next-generation routing protocols, and we thus propose such a protocol that prevents the association of source and destination by an attacker located at any fixed point within the network. Our key contributions are:

- An exploration of the design space for this class of anonymity system;
- A novel privacy-preserving network routing protocol, which is the first in this space;
- A comprehensive security analysis of this protocol;

- A systematic mechanism for measuring anonymity in terms of topological identity and application of this mechanism to our protocol;
- A detailed simulation environment capable of evaluating our protocol at Internet scale.

The remainder of this paper is structured as follows: Chapter 2 introduces our objectives and the adversary we are designing against. Chapter 3 discusses previous work in the field and summarizes two systems that we will later build upon. Chapter 4 presents our design from a sequence of different perspectives in increasing detail. Chapter 5 analyzes the security of our system by considering potential attacks and our defenses, and it derives the information available to a passive attacker. Chapter 6 describes our evaluation of the protocol by simulation, presenting the results in terms of additional path length and anonymity. Chapter 7 concludes with a summary of our findings and recommendations for further work.

CHAPTER 2

OBJECTIVES

2.1 Introduction

In this chapter, we consider the goals of the system we intend to deliver and the attacker we design against. Any anonymity system must be considered in terms of the benefit it provides to users through protecting their identity and the overheads users must pay in order to acquire this benefit. Our design requirements are motivated by these two points and also by practicality of application.

2.2 System Objectives

2.2.1 Anonymity

We refer to the party who initiates a communication session, or *connection*, as the *source*, and the opposite party as the *destination*, although data is able to pass in both directions once a connection has been established. Using the terminology of Pfitzmann and Hansen [20], we aim initially to provide *source anonymity*, such that the initiator of a connection cannot be identified within a set of possible initiators, referred to as the *anonymity set*. However, this property will be unavoidably weak at certain points within the network, such as the source's Internet service provider (ISP). A more critical concern is therefore *unlinkability* between the source and destination, such that no location within the network is able to sufficiently distinguish whether the source and destination are related. This implies that network locations with good information on the source identity have little information on the destination identity, and vice versa.

Optionally we allow *destination pseudonymity*, allowing a destination to be contacted through a pseudonym that does not reveal his[1] true identity. We assume that these pseudonyms are long lasting and publicly known, and so a source communicating via a pseudonym must remain unlinkable to that pseudonym in order to conceal her purpose. For example, consider the destination to be a controversial blog, run by Bob using a publicly available pseudonym to hide his involvement. If Alice wishes to anonymously post comments on this blog, her adversary already knows the pseudonym is associated with the blog, and so Alice must remain unlinkable to the pseudonym. To protect Bob's identity, he should also remain unlinkable to his pseudonym. Unfortunately, as we explain in Section 5.5, this cannot be achieved globally. Therefore, we weaken the destination unlinkability requirement to allow association by locations the host explicitly trusts.

Throughout our consideration of anonymity, we constrain ourselves to the identifying properties defined at the network layer: network identity and network location, or *topographical anonymity* [21]. Many other forms of Internet identification exist, such as device fingerprinting [22] and persistent cookies [23], but these are features of different layers in the protocol stack and therefore fall outside the scope of our work.

To summarize this discussion, we require that our system provides the following anonymity properties:

Requirement 1. *When destination pseudonymity is not in use, no entity at any location within the network shall be able to link the source and destination, expect for the source itself*

[1]Throughout this paper we use the genders of the standard security actors: The source, Alice, is female, the destination, Bob, is male, and the attacker, Eve, is female.

Requirement 2. *When destination pseudonymity is in use, no entity at any location within the network shall be able to link the source and the destination pseudonym, expect for the source itself.*

Requirement 3. *When destination pseudonymity is in use, no entity at any location within the network shall be able to link the destination and the destination pseudonym, expect for the destination itself and for locations explicitly trusted by the destination.*

2.2.2 Performance

An anonymity system must provide a level of performance that its users judge acceptable in order to gain widespread adoption and thus to provide a large set of potential users for each transaction to hide within. Performance problems with Tor have been widely discussed, and they are considered an important factor limiting its adoption [24, 25, 26]. We aim to provide a lightweight anonymity system that is implemented directly at layer 3 within the networking infrastructure. Remaining at this low level offers two distinct performance advantages in comparison to existing overlay anonymity networks: First, communication remains within the core Internet routing infrastructure, avoiding the frequently slow *last mile* connections [27] to end hosts that necessarily occur in an overlay network. Second, the queues used to transfer data between layers in the protocol stack are eliminated.

The latency of a connection is related to the length of the path taken. For anonymity reasons discussed in Section 4.5, we do not always wish to select the shortest network path, but it is important that the user be able to balance the additional latency of longer paths against anonymity. Finally, the transmission efficiency of the protocol must be reasonable. We consider this overhead based on the length of the packet header compared to that of current routing protocols.

Requirement 4. *Once a connection has been established, all processing between the source and destination shall be performed at layer 3.*

Requirement 5. *The design shall include mechanisms to trade performance and anonymity.*

Requirement 6. *The protocol header length shall be comparable to existing layer 3 protocols.*

2.2.3 Practicality

Our work considers a change to the Internet routing protocol and is therefore not intended for near- term adoption. However, we must still show that the resources required to implement the protocol at some future time would be practical. Our protocol must be capable of operating at *line speed* within an Internet router where throughput is high and computing resources per flow are extremely low. To comply with this environment, we require the following features of our protocol:

Requirement 7. *No per-connection state shall be necessary within routing nodes*

Requirement 8. *Limited cryptographic operations shall be performed within routing nodes*

2.3 Attack Model

We consider an adversary who is *active* but *local*. Active means the adversary is able to initiate connections and to violate the rules of the protocol for the connections in which she is involved, in addition to passively monitoring these connections. We define local as confined to a single *Autonomous System* (AS) within the Internet, noting that this is a more restrictive definition of the term than used in the context of Tor [28].

We consider the autonomous system to be the most natural granularity for assessing a threat, since this is the level at which routing information and policies are shared. A compromise in security at any router within an AS is likely to affect all routers controlled by the same organizational entity. In contrast, in order to span multiple ASs, an attack must either involve the compromise of multiple organizations or collusion between these organizations. Note that previous work for current low-latency anonymity systems has also focused on an AS-level threat [29].Where multiple ASs are known to be controlled by the same organization, they may be treated as a single autonomous entity for the purposes of anonymity.

Other practical threat models infer a higher degree of power or collusion, such as Internet Exchange monitoring [30], or state level organizations [28], but since current overlay-based low-latency anonymity systems offer limited protection against these threats, it is not reasonable to expect our lightweight scheme to fair better.

Our adversary is assumed to have global knowledge of the network topology and routing information, even though she only has local knowledge of traffic. This would be a difficult goal to achieve in practice, since mapping the Internet is a challenging research problem. However, we do not require confidential communication of routing information, and therefore there are no barriers to the accumulation of topology and routing knowledge over time.

We do not consider attacks that utilize the anonymity provided by our system for nefarious purposes against targets outside the system: Low-latency anonymity systems are already widely deployed, and as such we are not providing a new attack vector. For those Internet transactions that require it, identification and authentication should be built in, securely, using a higher-layer protocol.

CHAPTER 3

BACKGROUND

3.1 Introduction

In this chapter, we cover two areas of research which are of direct relevance to our problem – source-controlled network routing protocols and low-latency anonymity systems. Within each area, we describe one particular proposal that our design builds upon.

3.2 Source-Controlled Routing

One theme spanning a number of next-generation Internet routing proposals is that of source-controlled routing, in which the originator of a data packet has some control over the route it takes through the network, by way of routing control information carried in the data packet. For some protocols, the source has influence over the route but not complete control [16, 18]; for others the source explicitly declares the route that should be taken [14, 17]. As we explain in Section 4.2, this ability to express a route at the source has benefits for anonymity in addition to the robustness and flexibility considerations that motivated the initial research.

The concept of source-controlled routing is not new, and IPv4 includes source control options [31]. However, these capabilities are frequently disabled for security reasons; in the IPv4 implementation, a source could attempt to specify a route that violates routing policies or bypasses firewall rules. Although valid in the context of IPv4, we assert that these concerns are limited to insecure protocol implementations of source-controlled routing rather than an inherent flaw in the concept.

3.2.1 Pathlet Routing

The pathlet routing system [14] is one example of a source-controlled routing system. Each entity within a network defines a number of virtual nodes (or *vnodes*), and then advertises path segments (or *pathlets*) that pass between these vnodes. Vnodes are a virtual construct, and so a single physical router may process packets for multiple vnodes, or a single vnode may be distributed across multiple physical routers. Each vnode is defined by a forwarding table containing the set of allowed outgoing pathlets.

All packets arriving from a particular communication peer are processed by a single vnode, with the forwarding table for that vnode defining the set of allowed routes for that communication peer. The number of vnodes a given entity must define is driven by the complexity of its routing policy: For example, if all customers are allowed to use the same set of outgoing routes, then a single vnode can be used to process the traffic from all customers. The pathlet protocol provides an expressive system that is able to represent many different types of routing policy, but in particular, Gedfrey et al. demonstrate that entities with *local transit* policies (i.e. policies only dependent on ingress and egress points for their own network) may be represented efficiently and independently of the total network size. This is in contrast to the BGP exterior gateway protocol [32] commonly used today, where the forwarding information base must scale linearly with the total number of advertised IP prefixes.

To send a packet, the source assembles a list of adjacent pathlets used to deliver the packet by the selected route, and includes this list in the packet header. Each pathlet is represented by its Forwarding ID (*FID*), an index into the forwarding table of the vnode that defined the pathlet. FIDs are of variable length based on the size of each forwarding table. When a vnode receives a packet, it removes the first FID and uses this as an index into its forwarding table to determine which link the packet should be sent over. Note that only legal routes are defined in the forwarding tables, and therefore, unlike BGP, it is impossible to violate the routing policy by invoking private unannounced routes, since no such routes exist. This feature eliminates many of the security concerns often associated with source-controlled routing.

When a vnode learns of adjacent single hop pathlets, it may choose to aggregate these together into a composite pathlet representing a sequence of vnodes rather than only a pair. The translation from this composite into single hop pathlets is stored in the forwarding table of the vnode that defines it. When a packet arrives requesting this composite path, the forwarding table is used to replace the composite with the set of component pathlets, and then the packet may be processed as normal. Composite pathlets can reduce the number of pathlets required to specify a path and reduce the size of routing database necessary to reach all destinations.

Pathlet routing moves the responsibility for network route creation from the network infrastructure to the end hosts originating traffic. This means the large routing information base embodying network topology need only be consulted each time a new route is constructed, and not each time a packet is forwarded, but it also provides additional flexibility for a message source to control its own destiny. A source may rapidly select alternative routes to achieve better performance or compensate for network failures, instead of waiting minutes for an exterior gateway protocol to converge upon a new route.

3.3 Low-Latency Anonymity Systems

Previous works have introduced a variety of low-latency anonymity systems. The use of these systems incurs a latency penalty, but the average response time remains sufficient for general purpose interactive use, such as web browsing and shell sessions. A wide range of practical systems have been proposed and some of these have been fielded for general use [33, 4, 5, 6].

Current low-latency anonymity systems may be categorized as either centralized or distributed. Centralized systems simply pass all traffic though an anonymizing proxy and therefore require that users trust the central proxy. Distributed systems overlay an additional network on top of the current layer 3 protocol and therefore require multiple IP transmissions to deliver each packet from source to destination. These multiple transmissions, together with processing inside the intermediate hosts, contribute to latencies that can be substantially higher than Internet usage without anonymization [34].

3.3.1 Lightweight Anonymity and Privacy

In Lightweight Anonymity and Privacy (LAP) [21], Hsiao et al. propose the routing protocol anonymity scheme that inspires our work. Their protocol relies upon *packet-carried forwarding state*, in which the information required to deliver a packet is stored within the packet itself, and applies encryption to this forwarding state as the packet moves towards its destination. Each packet originates at its source containing a route through a sequence of *autonomous domains* (ADs). As each of these ADs receives the packet it locates its own routing instruction, encrypts this instruction using a symmetric key known only to itself, and forwards the packet to the next AD. Once a connection has been constructed in this manner, data may be exchanged between the two endpoints using the resulting encrypted header, with a system of sizes stored inside the encrypted information allowing each AD to locate its routing instruction within the encrypted byte-stream. Padding may be inserted into each encrypted block, providing partial obfuscation of the path length before and after a participating AD. However, the ability to pad is bounded by the need to maintain a manageable header size, and so the practical degree to which path length uncertainty may be created remains an open question.

The anonymity properties of this system are simple; during construction, each AD on the path learns the identity of all ADs that follow it but not the identity of the ADs before it. Some information on predecessor identity may be inferred based on knowledge of the preceding AD, network topology, routing policy, observed path length, and observed response time, but these are not quantified. Each path construction request contains a nonce that influences the encryption process, allowing a source to construct multiple unlinkable connections over the same route, using different nonces and producing different ciphertext. The symmetric keys used by each AS are changed frequently to further perturbate the ciphertext and to limit the impact of a key compromise.

LAP assumes the user's own ISP is trustworthy, and provides no protection of source-destination unlinkability against an observer at the source ISP. Given previous well-publicized ISP indiscretions [35, 36] and the possibility of a hacker infiltrating this single point of failure, it seems unlikely that privacy-conscious users will share this assumption. The authors do not consider modifying the standard shortest path route selection criteria, preferring instead an approach with *low path stretch* in the interest of efficiency.

CHAPTER 4

DESIGN

4.1 Introduction

In this chapter, we begin by considering the range of available options for designing anonymity systems at layer 3 and use this to justify our solution, the Dovetail protocol. This protocol is then described from four different perspectives in increasing detail: the network model, the high-level path construction process, the selection of routes for each segment within the path, and finally the detailed packet structure and protocol rules.

4.2 Layer 3 Anonymity Design Space

To provide an effective and broadly applicable anonymity system, we assert that any layer 3 solution must provide two features:

Deviation from shortest path. An attacker who observes a connection can make measurements to learn information on the length of the network path before and after their vantage point. Depending on the details of the protocol, many techniques may be available for these measurements, including round trip timing, time to live indicators, packet header length, route tracing, and active probing. If a routing protocol always selects the shortest possible route, then when the source and destination are significantly closer together or further apart than the Internet average, the protocol will reveal this information and limit their anonymity. Thus, we believe a protocol must be capable of selecting routes other than the shortest.

Partitioned routing knowledge. If the information required to route data is stored as a single field, such as an IP address, any entity with access to the field may calculate the network location of the destination. If instead, the information is divided across many different fields, then an entity must access multiple fields to calculate the destination location, and each field may be protected independently to prevent this access.

Source controlled routing, introduced in Section 3.2, is a useful technology for anonymity, since it accommodates both of these features: When the source of a message can dictate a path, she is free to pick one that is not the shortest, and the path may be expressed by the source as a set of instructions for each entity along the path, each of which may be encrypted separately.

If the previous and subsequent nodes in a path were visible during construction, clearly an intermediate node would be able to identify and link the source and the destination. If neither the previous nor subsequent nodes in a path are visible during construction, then unlinkability is possible. As we will show later, this is also true if the subsequent but not previous nodes in a path are visible during construction.

One could design an anonymity system in which the path is not visible in either direction by encrypting each stage of the selected path with a public key for the node that will handle it. Since nodes would not know each other's private keys, they would be unable to decrypt routing information for adjacent nodes and thus be unable to identify the source or destination. To mitigate the poor performance of public key encryption, a path construction phase could be introduced prior to data transfer, with each node converting its routing information from public key encryption to symmetric key encryption. This type of anonymity system would allow simple route selection and offers attractive anonymity properties, but it would require an extensive public key infrastructure (PKI) to distribute a public key for every node capable of routing anonymous traffic. We do not pursue this design any further, in order to avoid the barriers to adoption presented by this extensive PKI.

Instead, Dovetail uses the basic principle of construction requests that are traceable in a single direction as presented by Hsiao et al. [21] and discussed in Section 3.3.1. Our detailed description uses the pathlet routing protocol presented by Godfrey et al.[14] as discussed in Section 3.2.1. The pathlet routing protocol works well for our system, but note that we are not reliant on any unique feature of this protocol. The principles we describe could be applied to any lower level protocol that provides the following features: complete control over the selected route, a wide range of allowable routes, and secure enforcement of routing policies.

4.3 Network Model

Unlike the basic Pathlet routing protocol, we propose a clear distinction in routing at the AS boundary; each autonomous system should expose the minimum possible number of vnodes necessary to satisfy its routing policies, and only export pathlets that are necessary to provide an allowed route. This distinction is helpful from our perspective since the threat model is AS centric, but it also offers two practical benefits: First, minimizing the number of externally visible vnodes reduces the size of routing information base that must be held in end hosts. Second, distinguishing between internal and external connectivity allows an AS to retain a dynamic and responsive internal routing policy as provided by current internal gateway protocols.

The most common form of routing policy used in the Internet today is *valley-free routing* [37], which reflects the contractual relationships between ASs. A *customer* AS is one who pays a *provider* AS to forward its traffic, while two autonomous systems with a *peer* relationship will each forward the other's traffic without payment. In valley-free routing, each AS will only forward traffic when there is a financial incentive to do so, i.e. when the traffic originates from or is destined for a paying customer. In this strict definition of valley-free routing, each path is therefore comprised of zero or more customer-to-provider links, followed by zero or one peer-to-peer links, followed by zero or more provider-to-customer links. As illustrated in Figure 4.1a, a maximum of two vnodes per AS are required to enforce a valley-free routing policy. One is used to receive traffic from customer ASs and has permission to send to peer and provider ASs, and one is used to receive traffic from peer and provider ASs and only has permission to send to customer ASs. From this figure, it is simple to see that a paying customer must be included in every route through the AS.

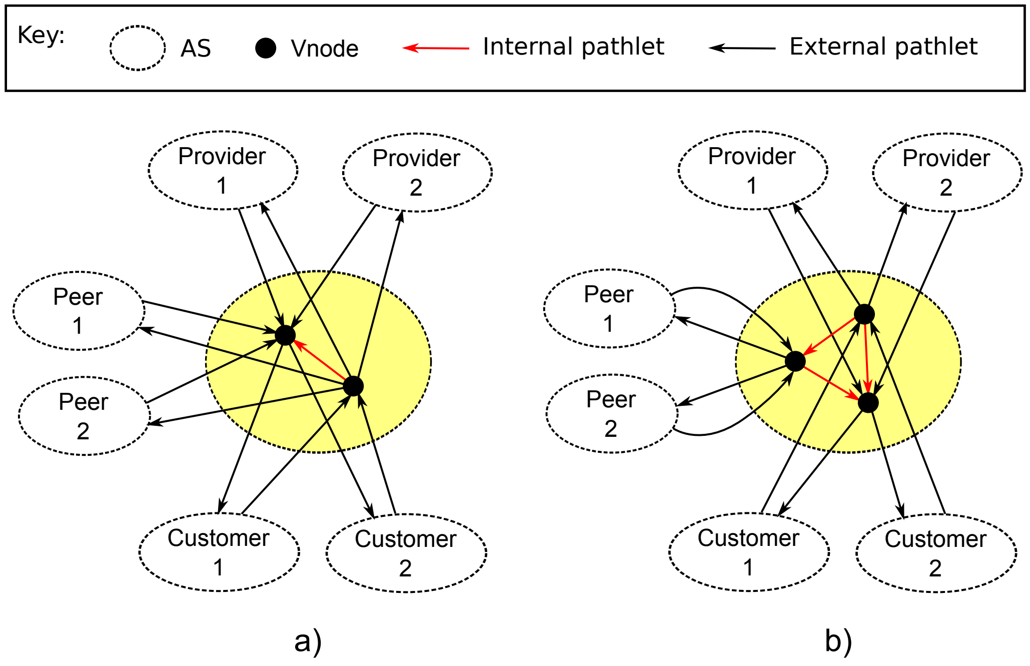


Figure 4.1. a) AS vnode and pathlet structure for strict valley free routing policy. b) AS vnode and pathlet structure for loose valley free routing policy.

It is likely that a large AS would define many more vnodes internally to represent its many routers, but the pathlets protocol allows each externally visible composite pathlet to be converted into a multi-step internal path upon entering the AS. This conversion is based on the forwarding table at the entry router, and it therefore may be changed dynamically to accommodate a rapidly changing internal environment without involving any entities external to the AS.

Although valley-free routing is common, BGP allows for arbitrarily complex routing policies and valley-free routing is not ubiquitous [38]. We also consider a slightly relaxed routing policy, which we refer to as *loose valley-free*. In this scheme, an AS will allow traffic to pass between its peers. The AS would not receive payment from a customer for performing this service, but also is not required to make a payment to a provider and may benefit at other times by being able to route through a chain of peers rather than paying a provider. In loose valley-free routing, each path is comprised of zero or more customer-to-provider links, followed by zero or more peer-to-peer links, followed by zero or more provider-to-customer links. As shown in Figure 4.1b, to enforce a loose valley-free routing policy, a maximum of three vnodes are required per AS: One is used to receive traffic from customer ASs and has permission to send to peer, provider, and customer ASs, one is used to receive traffic from provider ASs and only has permission to send to customer ASs, and the third is used to send and receive traffic from peers.

We assume that all hosts know the numeric identity of the vnodes they wish to contact. In a complete system, this would not be a realistic requirement and a lookup service would be required to translate between a human readable form of identity and vnode identity, similar to DNS in today's Internet. These supporting protocols must be carefully designed to avoid side channel attacks on anonymity, but this is outside the scope of our current work.

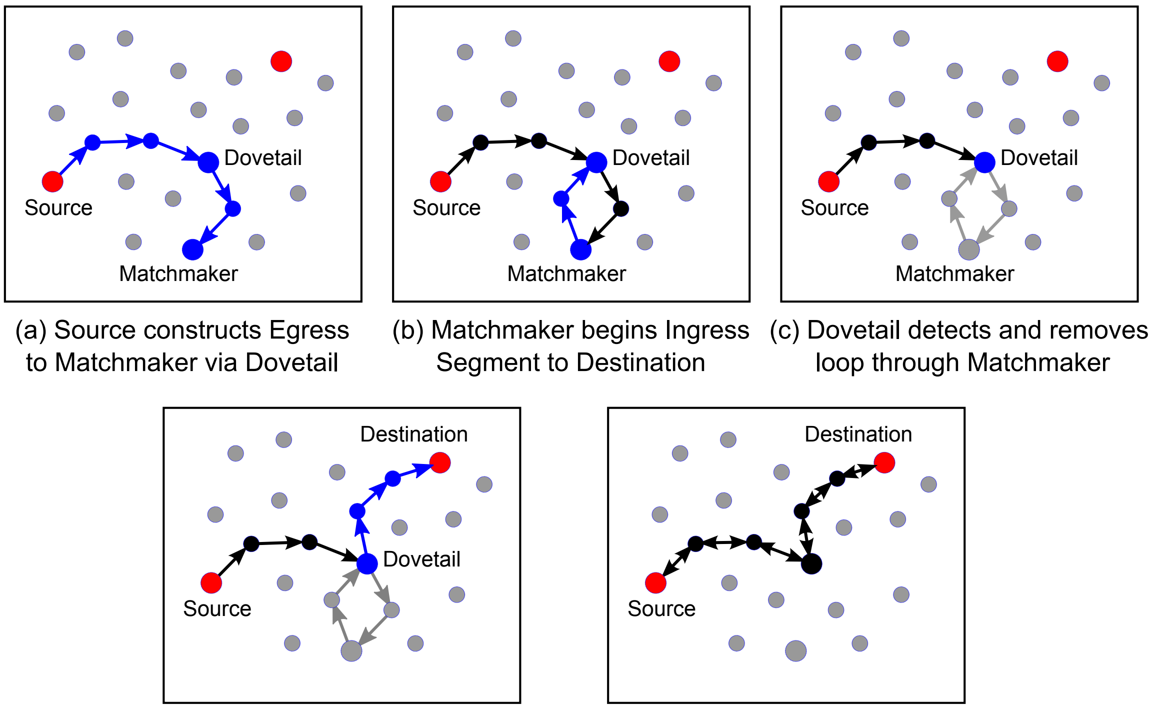
4.4 Path Construction

Each segment of the path in Dovetail begins with a construction request containing unencrypted forwarding instructions for each vnode in the segment. As each AS transfers this construction request, it encrypts its own forwarding instruction using a private symmetric encryption key, and therefore ASs later in the path cannot learn the forwarding instructions for earlier ASs. Once the construction request reaches the end of the segment, all forwarding instructions have been encrypted by their respective nodes, and this fully encrypted path representation may then be used to transfer data in both directions.

Clearly a path cannot be constructed directly from the source to the destination, since the source's ISP would see the unencrypted destination identity in violation of Requirement 1. Instead, we make use of a randomly selected untrusted third party host referred to as the *matchmaker*. The source encrypts the identity of the final destination using a public key for the matchmaker and builds an *egress* path segment to the matchmaker, who then extends the path to the destination with an *ingress* path segment. Here the source ISP no longer learns the identity of the destination, only of the matchmaker. The matchmaker learns the identity of the destination, but does not know the identity of the source, and so basic source-destination unlinkability is achieved.

However, we prefer that the matchmaker not be involved in the exchange of data between source and destination for two reasons: first, to meet Requirement 4 that all communication after path construction be native layer 3 traffic, and second, to minimize the trust we must place in the matchmaker. This is achieved by requiring that the egress and ingress paths cross at some vnode referred to as the *dovetail*[1].

[1]We use the term to reflect a dovetail joint in carpentry, where two elements are joined securely and compactly



(a) Source constructs Egress to Matchmaker via Dovetail

(b) Matchmaker begins Ingress Segment to Destination

(c) Dovetail detects and removes loop through Matchmaker

(d) Dovetail continues Ingress Segment to Destination

(e) Source and Destination communicate over established connection

Figure 4.2. Construction of a Dovetail connection.

The dovetail vnode detects this condition and joins the two paths, removing the loop in the path along with the matchmaker. As will be discussed in Section 4.6, this join technique is secured to prevent attackers using joins to probe the path. Figure 4.2 illustrates the overall path creation process.

Our anonymity can be degraded if an AS appears early on the egress and also on the ingress, and therefore we would like the ingress path to avoid the ASs used for egress. We cannot reveal the set of ASs used on the egress path to the matchmaker, since this would reveal substantial information on the source identity. However, we can request that the matchmaker returns a small set of potential ingress routes for the source to select from, concealed using a session key sent by the source alongside the identity of the final destination. This introduces additional latency into connection construction, but the matchmaker does not learn whether the unselected routes were acceptable or not, and therefore does not learn the source identity. An intersection attack based on these discarded routes may be possible over many requests, but matchmakers are selected randomly from a large set and an attacker located at any particular matchmaker is unlikely to receive a large number of connection requests from the same source. For brevity, we do not include this ingress selection mechanism in our detailed description of the packet design in Section 4.6, but we do examine its effect on anonymity in Section 6.4.

We now consider the case in which the destination wishes to use a pseudonym for communication, concealing his true identity from the source. This situation is similar to a *hidden service* in Tor [6], although our solution is different. A destination can publish a particular vnode, which we refer to as the *ingress target*, along with an encrypted *approach* path segment from this vnode to himself, all signed with his private key. A source wishing to communicate with a pseudonymous destination includes the ingress target and approach path in the information she encrypts for the matchmaker in place of the final destination. Figure 4.3 illustrates this complete scheme.

The use of a dovetail/matchmaker and the use of an ingress target are both optional parts of our protocol. It would be possible to use an ingress target but not a matchmaker. Both these options provide differing types of anonymity for the source and destination and are allowed in our detailed packet design, but they do not align with our anonymity objectives and therefore are not addressed further.

4.5 Segment Route Selection

A source-controlled routing system may attempt to obfuscate the length of the path preceding and following some intermediate node, but in general an attacker lo-

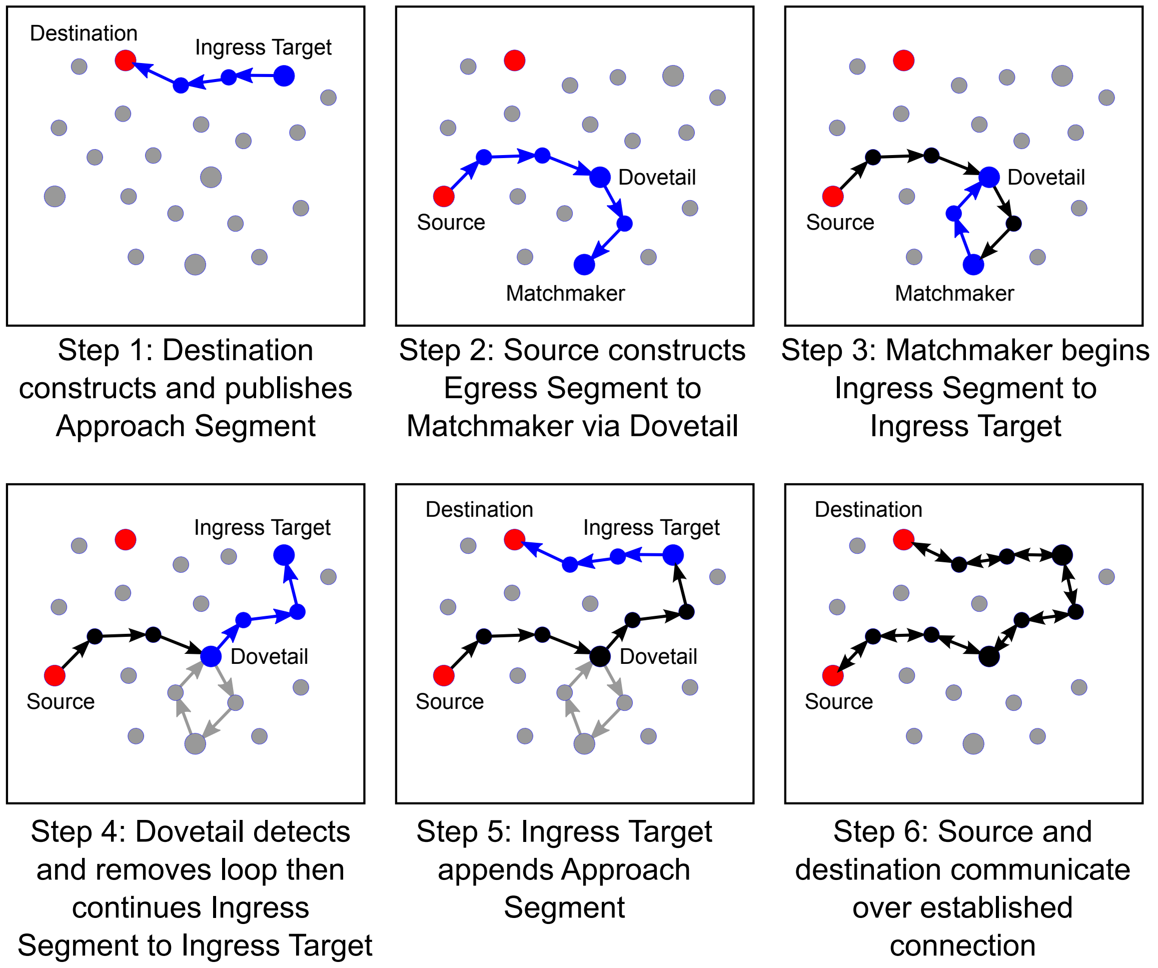

Destination
Ingress Target
Step 1: Destination constructs and publishes Approach Segment
Dovetail
Source
Matchmaker
Step 2: Source constructs Egress Segment to Matchmaker via Dovetail
Dovetail
Source
Matchmaker
Step 3: Matchmaker begins Ingress Segment to Ingress Target
Ingress Target
Dovetail
Source
Step 4: Dovetail detects and removes loop then continues Ingress Segment to Ingress Target
Destination
Ingress Target
Dovetail
Source
Step 5: Ingress Target appends Approach Segment
Destination
Source
Step 6: Source and destination communicate over established connection

Figure 4.3. Construction of a Dovetail connection to a pseudonymous destination.

cated at an intermediate node will be able to learn information about these quantities through combinations of round trip timing, packet length and structure analysis, and active probing. We prefer a system that is robust even when an attacker learns path length to one that relies on keeping it hidden. For the remainder of the discussion, we assume the attacker has perfect knowledge of the number of autonomous systems preceding and following her own each time she observes a path segment, even though gathering this information is non-trivial in our system.

Each segment of our constructed path serves a particular purpose in concealing the location of one participant from another, as we will explore in Section 5.5. If the route selected for a path segment were very short, it would fail in achieving this, since only a small set of source-destination pairings would be able to achieve the observed length. We believe that deviation from the shortest path is a necessary property for this class of anonymity system and demonstrate the value of this approach in Section 6.3. Our mechanism for selecting each path segment is based upon the principle of *path diversity*, where a large number of possible paths may be taken from any given source to any given destination; this is beneficial for the robustness of the system in addition to its anonymity.

Path diversity requires that the host constructing a path knows of more than one route to each destination. Each vnode following the pathlet routing protocol exports the set of pathlets forming its shortest path tree (SPT) to its neighbors, ensuring all vnodes learn a path to all destinations, but this alone does not lead to significant path diversity. It would be possible for each vnode to export all known pathlets, which would lead to a global knowledge of all Internet routes, but this is unnecessarily expensive, requiring that every route change be distributed through the complete Internet. Instead, we propose a middle road, where each vnode exports all pathlets on its SPT and then an additional set of known pathlets. The size of this set is expressed as a fraction of the size of the SPT. An important consequence of this approach is that routing knowledge differs across the network, and any assessment of path options or probabilities can only be made in the context of the vnode selecting the path.

The additional pathlets selected for export are those closest to the exporting vnode that do not form a part of its SPT. These pathlets will in turn be close to the receiving vnode, and therefore will be able to influence its path choice for a comparatively wide range of downstream destinations. The optimal number of additional pathlets to export depends on the size and topology of the network, but our experiments show that exporting 50% of the SPT quantity is suitable in the current Internet. Maintenance of the exported pathlets in response to network routing changes can be performed using path vector distribution methods similar to BGP [32], but this is not relevant to the anonymity properties of the system and so is not discussed further.

The pathlet dissemination mechanism above ensures that a host will usually have a wide range of routing options available each time it must construct a path segment. These options will have different *costs*, where we define cost as the number of times the route moves from one AS to another. The distribution of options across cost reflects the network topology between the source and destination. For example, the lowest cost at which an option is available is always the shortest path cost. Selecting a random path uniformly from among the complete set of options would reveal information about this distribution and therefore leak information about the topology. Instead, we select a path by first selecting an available path cost and then randomly selecting one of the paths at this cost. We can use a variety of techniques to select the path cost, each with a different balance between between performance (favoring paths that are close to the shortest) and anonymity (favoring paths that are longer than necessary). Variation in the cost selection method is our primary mechanism to exchange anonymity for performance in accordance with Requirement 5. Section 6.3 compares a series of different selection techniques.

4.6 Data Packet Structure

Dovetail extends the basic packet format used in pathlet routing, providing a range of different packet types to serve different steps in the construction of a connection and the exchange of data. Each Dovetail packet contains a type identifier in the first byte, followed by a series of one or more variable-length header segments determined by the packet type, followed by the information payload. Figure 4.4 presents the available types and the header segments they include.

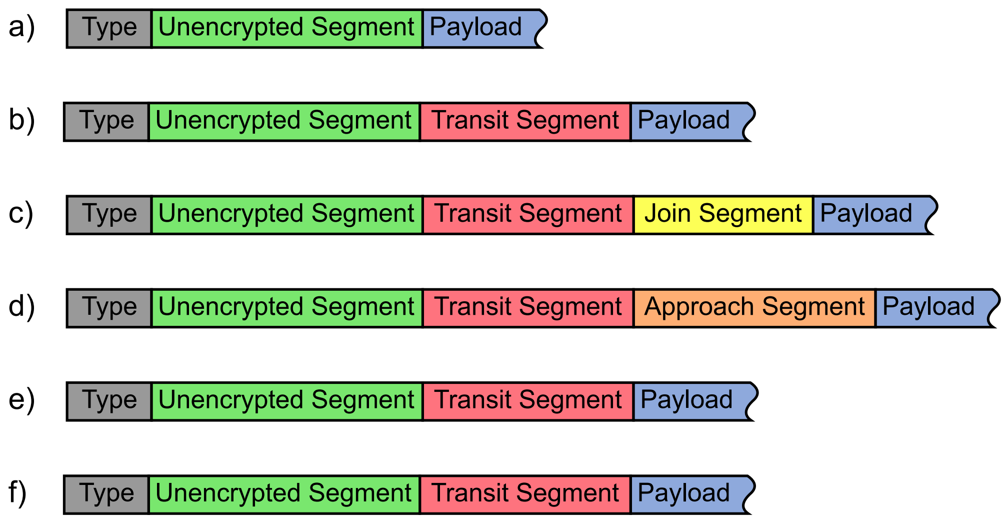


Figure 4.4. Structure for each Dovetail packet type: a) Plain packet. b) Path construction packet. c) Joinable path construction packet. d) Approach construction packet. e) Encrypted data packet. f) Encrypted response packet.

Dovetail does not perform any cryptographic operations on the packet payload, only on routing information contained in the packet header. Restricting our operations to a small number of bytes in the header allows for fast operation to meet Requirement 8, and since we are not concerned with collusion between different network locations, there is no need to provide bitwise unlinkability of a message at different locations. Higher layers in the protocol stack, such as TLS, may be used to encrypt or authenticate data in cases where confidentiality or integrity are important.

To create a Dovetail connection, the source issues a joinable path construction packet, leading the destination to respond with an encrypted response packet. Once the response is received, the source and destination may continue to communicate over the connection using a sequence of encrypted data and encrypted response packets. We note that it may be possible to combine this layer 3 construction phase with the initial handshake required in higher level protocols such as TCP.

We now discuss each of these packet types in turn, covering its purpose and the behavior required when a vnode receives the packet. Packet types are introduced in order of increasing complexity. Table 4.1 summarizes the notation used in this section, while Figure 4.5 illustrates the structure of each header segment. Each segment begins with a segment size, allowing the start of the next segment to be located. The FID strings used by Pathlets are always a multiple of four bits, and therefore we define all sizes in terms of *nibbles*, or half-bytes. Appendix A provides an example of a Dovetail communication session showing the population of these fields at each point in the network during an example connection and may help in visualizing the operation of the protocol.

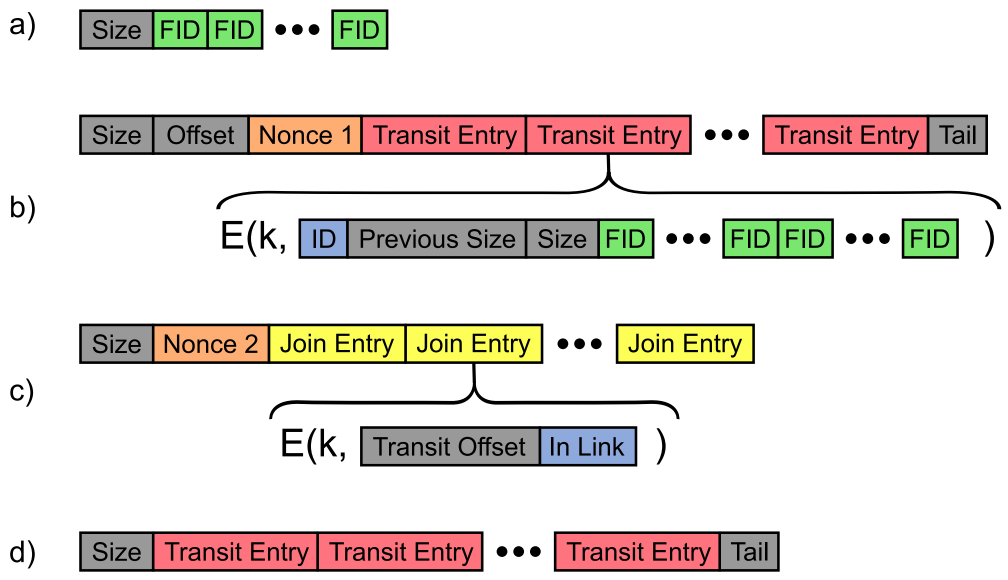


Figure 4.5. Structure for each Dovetail header segment type: a) Unencrypted header segment. b) Transit header segment. c) Join header segment. d) Approach header segment.

Table 4.1. Packet structure notation

Term	Definition
id_A	Identifier for AS A
li_A	Transmission link used to enter AS A
k_A	Symmetric encryption key for AS A
$p_{A1}...p_{An}$	Sequence of pathlets traversing AS A in the forward direction
$q_{A1}...q_{An}$	Sequence of pathlets traversing AS A in the reverse direction
T_A	Transit entry for AS A
$T_{A'}$	Transit entry preceding that for AS A
J_A	Join entry for AS A
R_A	Approach entry for AS A
$R_{A'}$	Approach entry preceding that for AS A
$N1$, $N2$	Random nonce
$\text{len}(x)$	Length of field x
$\text{offset}(x)$	Offset of field x from segment start
$H(x)$	Cryptographically secure hash of x
$E(k, v, x)$	Encryption of x with key k and IV v

4.6.1 Plain Packet

Plain packets follow the underlying pathlet routing protocol without the anonymity features introduced by Dovetail. A plain packet consists only of the unencrypted header segment. Following the size, an unencrypted header segment contains a string of *forwarding identifiers* (FIDs), each of which is an index into the forwarding table for a receiving vnode.

Each time a vnode receives a plain packet, the vnode performs the following actions:

1. Remove the first FID from the start of the unencrypted segment and look up this entry in the forwarding table;
2. Prepend any additional FIDs listed in the forwarding table to the start of the FIDs field;

3. Forward the packet over the link specified in the forwarding table.

4.6.2 Path Construction Packet

A path construction packet is used by a source to establish an encrypted path directly from source to destination, without a dovetail or matchmaker. This direct connection is similar to the technique used by Hsiao et al. [21] and would not provide protection against an attacker early in the path. Direct connections are not considered in our analysis or evaluation, but the packet type is natural to include in our design and its introduction here simplifies the description of later, more sophisticated, packet types.

The source begins by populating the packet with a complete unencrypted segment defining the intended path and a transit segment (shown in Figure 4.5) containing a segment size, a random nonce, an offset and tail set to zero, and no transit entries.

Each time a path construction packet enters a new autonomous system A, the receiving vnode performs the following actions:

1. Read the portion of the unencrypted path that passes through A, $p_{A1}, ...p_{An}$;
2. Query AS internal routing information to determine the shortest externally visible path that passes between these neighboring ASs in the opposite direction, $q_{A1}, ...q_{An}$;
3. Read the size of the previous transit entry, $\text{len}(T'_A)$ from the tail field;
4. Create a new transit entry, T_A:

   $T_A = E(k_A, T_{A'}||N1, id_A||\text{len}(T'_A)||\text{len}(T_A)||p_{A1}, ...p_{An}||q_{A1}, ...q_{An})$;
5. Append T_A to the end of the transit entries field;
6. Update the tail field to $\text{len}(T_A)$;
7. Perform the actions required to process a plain packet.

Subsequent vnodes within an AS may handle path construction packets as if they were plain packets.

In the topologies we consider, the path portion crossing an AS will be at most two pathlets. The reverse path portion might not pass through the same vnodes as the outgoing portion, but such a reverse path is guaranteed to exist because the routing policy must support bidirectional communication. We include $N1$ in the IV to prevent different connections over the same path leading to the same ciphertext, and also include the previous transit entry in the IV to prevent potential ciphertext splicing attacks. Each encrypted transit entity includes an identifier for the creating AS, so that corruption or key change errors may be quickly detected instead of inadvertently sending the packet down an incorrect path. This integrity verification also dissuades active attacks based on header manipulation. The number of pathlets required to transit an AS may vary, as may the size of FID used to encode each pathlet, and therefore transit entries are variable in length. We accounted for this by including the size of the current and previous transit entries in the entry itself.

When a path construction packet reaches its destination (i.e. when it is received with an empty unencrypted path segment), the receiver examines the packet payload. If the payload specifies an encrypted approach path, the receiver should change the packet type to *approach construction*, copy the encrypted approach path into the approach header segment, and then continue processing following the approach construction rules. In all other cases, the receiver should respond with an encrypted response packet.

4.6.3 Joinable Path Construction Packet

A joinable path construction packet is used by a source to establish an encrypted path through a matchmaker and dovetail. The unencrypted segment and transit segment are present and follow the same rules as in the case of a path construction packet, but an additional join segment is present in the header. The source begins by populating this segment with a second nonce $N2$, but no join entries.

Each time a joinable path construction packet enters a new autonomous system A, the receiving vnode performs the following actions:

1. Perform the actions required to process a path construction packet;
2. Test for the join condition by decrypting all previous entries in the join segment with a key of k_A and an IV of $N1||N2$. If any entry J_X decrypts to a valid link ID and offset:
   (a) Use this offset to locate and decrypt the original transfer entry passing through the AS, T_X;
   (b) Adjust T_X to include the current AS exit point;
   (c) Remove all transfer entries following T_X;
   (d) Remove all join entries following J_X;
3. If the join condition is not met:
   (a) Create a new join entry, J_A: $J_A = E(k_A, N1||N2, \text{offset}(T_A)||li_A)$;
   (b) Append J_A to the end of the join entries field;
4. Set $N2 = H(N2)$.

Subsequent vnodes within an AS may handle joinable path construction packets as if they were plain packets.

This design ensures that an AS will only honor an attempt to join path segments when it has previously been included on the path and when it receives the same value of $N2$ both times. $N2$ is updated by each AS in a chained hash scheme similar to that proposed by Lamport [39], preventing an AS from predicting the nonce seen by earlier ASs and so thwarting an active attack discussed in Section 5.3. The transit entry offset and link ID are included in the join entry as a practical matter, allowing an AS to find the variable-length transit record it previously created and to recall the original incoming link, facilitating construction of a return path over this link.

When a joinable path construction packet reaches its destination (i.e. when it is received with an empty unencrypted path segment), the receiver examines the packet payload. Encrypted approach payloads are handled as for a path construction packet, but there is one more possible payload type, the *continuation request*: $E(KU_M, id_M||id_D||id_V||\lambda_{MV}||f(N2)||P)$. When the matchmaker detects a continuation request, it decrypts the payload using its public key KU_M to learn:

- The identities of the destination and dovetail;
- The desired cost from the matchmaker to the dovetail, λ_{MV};
- A previous hash of the nonce, $f(N2)$, representing the value of $N2$ λ_{MV} steps prior to the dovetail on the egress segment;
- A new payload, P.

The matchmaker uses this information to place an ingress path in the unencrypted segment of the packet, selecting a path containing the dovetail at a cost of λ_{MV} and terminating at the destination. The packet payload is set to P, and $N2$ is set to $f(N2)$, such that when the packet arrives at the dovetail λ_{MV} steps later, $N2$ will match the value observed during construction of the egress path.

4.6.4 Approach Construction Packet

An approach construction packet is used to incorporate approach path information provided in an approach header segment during the journey from an ingress target to the destination. The packet includes the unencrypted, transit, and approach header segments.

The approach segment is originally created by the destination and published as a part of the destination's pseudonym. A destination may create an approach path by simply opening a connection to the ingress target using the well-known $N1$ of zero. The transit segment from this connection is published (with offset and nonce removed) as the approach path. Clearly this segment contains all the information necessary to securely travel from ingress target to destination, but the entries are in reverse order. Thus, while processing an approach construction packet, each AS must remove the last entry from the approach segment, reverse the information, and append to the end of the transit segment.

Each time a path construction packet enters a new autonomous system A, the receiving vnode performs the following actions:

1. Remove any existing entries in the unencrypted path;
2. Decrypt its approach entry R_A from the approach segment, beginning at the approach segment size minus the approach tail, using an IV of $R_{A'}||0$
3. Copy $q_{A1}, ...q_{An}$ from R_A to the start of the unencrypted path;
4. Delete R_A from the approach segment;
5. Update the approach tail field to $\text{len}(R_{A'})$;
6. Read the size of the previous transit entry, $\text{len}(T'_A)$ from the transit tail field;
7. Create a new transit entry, T_A:

   $T_A = E(k_A, T_{A'}||N1, id_A||\text{len}(T'_A)||\text{len}(R_A)||q_{A1}, ...q_{An}||p_{A1}, ...p_{An})$, where p and q are the values read from R_A;
8. Append T_A to the end of the transit entries field;
9. Update the transit segment tail field to $\text{len}(T_A)$;
10. Perform the actions required to process a plain packet.

Subsequent vnodes within an AS may handle approach construction packets as if they were plain packets.

When an approach construction packet reaches its destination (i.e. when it is received without approach entries in the approach segment and with an empty unencrypted segment), the receiver should respond with an encrypted response packet.

4.6.5 Encrypted Data Packet

An encrypted data packet is used by the source to send data over an existing Dovetail connection. The source initializes the unencrypted segment to contain no entries and initializes the offset of the transit segment to zero.

Each time an encrypted data packet enters a new autonomous system AS_A, the receiving vnode performs the following actions:

1. Remove any existing entries in the unencrypted path;
2. Decrypt its transit entry T_A beginning at the value stored in the offset field;
3. Copy $p_{A1}, ...p_{An}$ from T_A to the start of the unencrypted path;
4. Increment the offset field by the size of T_A;
5. Perform the actions required to process a plain packet.

Subsequent vnodes within an AS may handle encrypted data packets as if they were plain packets.

4.6.6 Encrypted Response Packet

An encrypted response packet is used by the destination to send data over an existing Dovetail connection. The destination initializes the unencrypted segment to contain no entries and initializes the offset of the transit segment to the offset of the last transit entry, derived using the transit segment size and the tail field.

Processing of an encrypted response packet is identical to that of an encrypted data packet, except that the return path $p_{A1}, ...q_{An}$ is copied from each T_A and the offset field is decremented by the size of the previous transit entry.

CHAPTER 5

SECURITY ANALYSIS

5.1 Introduction

In this chapter, we assess the extent to which the Dovetail protocol achieves its security goals. We begin by considering a range of attacks that might be applied against the protocol, both from a system availability perspective and an anonymity perspective. Then, we discuss the information available to a passive attacker at each point in the network, and how this information may be used to assess the probability that any particular host is the source or destination of an observed communication. In Section 5.4 we begin by considering the anonymity along a single segment of the path, and then in Section 5.5 we extend the analysis to the complete Dovetail system. These sections form the basis for the numerical anonymity assessment we present in Chapter 6.

5.2 Availability and Integrity Attacks

Construct a path that violates routing policy. All entries stored in a forwarding table are valid expressions of the routing policy, and therefore it is not possible to violate this policy.

Construct arbitrarily long paths or looping paths. Our design constrains the maximum length of both encrypted and unencrypted packet header segments, and thus limits the longest path an adversary can construct. It is important that looping paths are possible, otherwise loop detection could be used as an oracle, and so a single AS may appear on a path multiple times.

Overload a matchmaker with continuation requests. A matchmaker could be overloaded by sending a large number of requests, but matchmakers are distributed throughout the network and the effect on clients is minor if the first matchmaker they contact is unavailable. Matchmakers are selected randomly, so an attacker cannot predict which clients will be delayed by blocking a particular matchmaker.

Overload a routing vnode. The forwarding operations we perform are simple and intended to operate at the full data rate of a router, while the lack of any per-connection state inside a vnode means passing many different connections through a vnode does not consume memory. Vnodes are virtual entities, and therefore an AS is free to move processing and traffic flow within its network to balance load.

Pollute routing tables. Securing the integrity of routing tables updates is an important requirement, but we do not consider the routing maintenance protocol here hence it is outside the scope of our work.

Modify packet contents. Dovetail is a layer 3 protocol and does not provide any protections for the data it is used to carry. In cases where integrity is important, a higher layer protocol should be used to provide authentication.

Discard packet data. If the quality of service provided by a connection drops below some threshold, this would be observed as a failure, for which the recommended remedy is to reconnect over a different path. Paths are constructed by random selection from the available routes, and so this reconnection is likely to remove any intermediate AS discarding data.

5.3 Anonymity Attacks

Observe packet content. Dovetail is a layer 3 protocol and does not provide any protections for the data it is used to carry. In cases where confidentiality is important, or where packet content would reveal identity, a higher layer protocol should be used to provide encryption.

Isolate connection by packet content. An attacker who is able to observe a connection at different points in its path is able to correlate the two data streams by matching packet content. Our threat model does not include collusion between different ASs. Our route selection process includes measures to reduce the probability of using an AS multiple times in a path, but when an AS does appear at multiple locations it may correlate packet content to learn this fact. We note that in general it is not possible to eliminate the possibility of an AS appearing in the path twice. For example, if a source and a pseudonymous destination are served by the same ISP, neither should be able to learn this, and yet the ISP will be able to monitor both ends of their communication.

Isolate connection by traffic analysis. Either passive or active timing correlation attacks are simply a more complex method to reach the result discussed in the packet content correlation attack above.

Correlate different connections from the same source. Each connection includes a nonce. When the source changes this nonce a different ciphertext will be produced, preventing an observer from associating multiple connections over the same path by their header content. Correlation using repeated patterns in the packet payload is a function of the application layer protocol and therefore outside the scope of our work.

Observe path length and entry/exit direction. Estimates of the cost to both the source and destination are available at intermediate points on a Dovetail path, along with the identity of the preceding and following ASs. The utility of this information in identifying the source and destination is an important discussion which is deferred to Sections 5.4 and 5.5.

Observe round trip timing. Response timing data, both for the source and the destination, is another valuable source of information to a passive attacker, but one whose assessment is mainly left for future work. The impact of timing data is discussed briefly in Section 5.6.

Replay encrypted packet. A replayed packet will take the same path as its original transmission and therefore not provide an attacker with new information. An adversary might try to probe for the entry point by prepending different unencrypted segments to the start of a recorded packet, but each AS empties the unencrypted segment on receipt to prevent this attack.

Splice multiple encrypted headers. An adversary may try to splice together partial encrypted paths to test whether a particular AS is common to the two. However, each AS uses the ciphertext from the previous transit entry in its own initialization vector, so changing an earlier segment will cause decryption to fail.

Modify requested path. An AS along the construction path is able to modify the route taken for the remainder of the current segment by altering the unencrypted segment, but stands little to gain from doing so. All vnodes along a path segment have complete knowledge of the destination, and vnodes closer to the start of the path have a better knowledge of the source. An AS is able to place itself later in the same path, but this does not provide any additional information regarding the source or destination. An AS might modify the path in order to benefit a different AS, but this collusion is outside our threat model and does not provide more information than the two ASs would gain by sharing their observations directly.

Probe for an earlier AS by stimulating a join. The joining of a Dovetail path provides confirmation that the joining AS appeared on the path twice, and an attacker may wish use this feature to probe for suspected predecessors. During connection construction, an attacker may attempt to extend the path to a suspect and then back to herself, where she could observe whether a join had occurred. Our use of hash chaining on $N2$ prevents this attack, since the attacker cannot replicate the value of $N2$ initially presented to the suspect. The matchmaker is provided with an earlier version of $N2$ in order to create a legal join and may therefore perform some probing, but this is heavily constrained by the dovetail-matchmaker cost. When the dovetail-matchmaker cost is set to the recommended value of two, the matchmaker may only probe for the AS immediately preceding the dovetail, and then only in cases where this AS has a direct link to the matchmaker.

Compromise AS symmetric key. Each AS is considered a potential adversary, and compromising the key for a single AS does not reveal more information than would be available to an attacker at that AS. An adversary strong enough to compromise multiple ASs is beyond our threat model. Symmetric keys are changed on a regular basis and so a key compromise need not have long lasting effect.

Compromise Matchmaker private key. An attacker who observed a connection close to the source and could compromise the matchmaker's private key would be able to decrypt the final destination and link the source and destination. However, matchmakers are selected randomly and so there is no way for an attacker to determine in advance which matchmaker will be used.

Fingerprint end server by traffic analysis. The patterns of file quantity, size, and timing produced by a particular web server have previously been shown to be an effective fingerprint for identifying its traffic [40]. Dovetail does not contain defenses against this form of attack: we present our work as a layer 3 anonymity protocol to conceal only network identity, and in the spirit of modularity avoid embedding an understanding of higher layer forms of identity.

Congestion based side channel attacks. Previous works have presented attacks that selectively induce high loads on particular portions of the network and monitor for corresponding delays in an anonymous connection [41, 42]. As with more complex low-latency anonymity systems, Dovetail is vulnerable to this form of attack.

5.4 Single Segment Anonymity Analysis

To analyze the information available to a passive attacker, we first consider the source and destination anonymity provided at each point along a single path segment. This problem is a subcomponent of the larger complete path assessment problem. Dovetail selects a path for each segment by first choosing a desired cost, and then randomly choosing one of the routes available at this cost. The simplest approach would be to always select the lowest possible cost. This will often give the lowest latency path, and is the technique used by nearly all previous routing algorithms. However, the manner in which a source selects a path segment determines the information that can be gained from observing the segment, and we show that an alternative path selection algorithm offers better anonymity.

5.4.1 Anonymity Set Size

Consider an attacking AS, AS_i, located at cost i in a path segment $AS_0, AS_1, \cdots AS_{i-1}, AS_i \cdots AS_n$. This AS may observe which AS it received the construction packet from, AS_{i-1}, but cannot directly identify any earlier ASs, since their routing instructions have been encrypted. The remaining portion of the segment is not encrypted and therefore all following ASs, including the destination, are known. AS_i can accurately measure the total path cost from the source to itself by using the length of the join segment in the packet header, and therefore can deduce the total path cost, n.

We use $S_y^{AS_x}$ to represent the set of possible sources that have a shortest path to AS_x of cost y. When paths are selected using the shortest path algorithm, an adversary knows that the observed cost for AS_{i-1} and for all subsequent ASs must be the shortest cost from the true source. The set of possible sources is therefore the intersection of the possible source sets for each of the observable ASs:

$$\text{sources}(\textit{shortest path}, AS_i) = \bigcap_{j=(i-1)}^{n} S_j^{AS_j} \tag{5.1}$$

If i is close to the minimum or maximum costs present in the Internet, then few sources will have shortest paths that meet these criteria, leading to an uncomfortably small anonymity set.

An alternative, *cost window*, selection algorithm is to randomly select a length between some global minimum p (or the shortest path cost if this is greater) and some global maximum q, where q is greater than the maximum shortest path cost in the network. Here, an attacker cannot make any statement about the possible message senders except each must be able to form a path of the observed length to the observed predecessor AS_{i-1}.

Our experiments show that once even a small fraction of ASs use a loose valley-free routing policy, long path choices are plentiful. This means that in most cases a source can produce a path at any given cost greater than the minimum. We find this to be true 96% of the time when 10% of ASs are loose valley-free. Making an approximation that this is always true, then the set of possible sources is simply the union of the sources at every distance less than or equal to the observed value:

$$\text{sources}(\textit{cost window}, AS_i) = \bigcup_{j=0}^{i-1} S_j^{AS_{i-1}} \tag{5.2}$$

By examining the relationship to $A_{i-1}^{AS_{i-1}}$, it is clear that:

$$\text{sources}(\textit{shortest path}, AS_i) \subseteq \text{sources}(\textit{cost window}, AS_i)$$

Hence, *cost window* selection provides an equal or larger source anonymity set in all cases. In addition, n is capped to a minimum of q, and therefore very short path costs with unavoidably small anonymity sets will never be generated.

Using cost window selection, even sources with a low shortest path cost will occasionally select very long paths, and therefore the average latency is higher than shortest path selection. A probability distribution can be used to control how frequently larger costs are selected and limit this performance penalty, as we investigate in Section 6.3.

5.4.2 Effective Anonymity Set Size

The preceding analysis for source anonymity set size is simple and efficient for an attacker to compute using only shortest path distances, but with complete knowledge of routing tables and sufficient processing capability she can achieve a better result. The probability that each potential source selected the observed path depends on the available path options for that source, and therefore is not uniform across the set of potential senders. These differences in probability allow calculation of an effective anonymity set size based on the entropy of the distribution. The notation we use in this section is summarized in Table 5.1.

For each possible source, t, the probability of selecting the observed path cost, $P_{cost\,match}(t)$, depends only on the path selection algorithm and the presence or absence of paths in $R(t, d)$ at each cost, not the number or definition of these paths, nor the location of the attacker. Section 6.3 presents a series of different options for path length selection, along with their cost selection probabilities.

Once a cost has been selected, a path of this cost is chosen randomly from the available set. The probability that t chose a path that matching the observation is

Table 5.1. Source entropy notation

Term	Definition
S	The set of all possible sending vnodes
s	The true source vnode
d	The destination vnode
r	The route selected to deliver a message from s to d
a	An attacking vnode located on r and wishing to identify s within S
a'	The vnode on r immediately preceding a
$\mathrm{tail}(r, x)$	The portion of path r after vnode x
$\lambda_r(x)$	The cost along a path r from the source vnode to vnode x
OBS_a	The complete set of observations available to a
$R(x, y)$	The set of all possible routes from vnode x to vnode y
$R(x, y, \lambda)$	The set of all routes from vnode x to vnode y of cost λ

therefore given by the fraction of paths that place the observed predecessor at the observed cost, and that match the observable portion of the path:

$$P_{predecessor\,match}(t) = \frac{|q, q \in R(t, d, \lambda_r(d)) \wedge \lambda_q(a') = \lambda_r(a') \wedge \mathrm{tail}(q, a') = \mathrm{tail}(r, a')|}{|R(t, d, \lambda_r(d))|} \tag{5.3}$$

We assume that the a priori probability of each routing node t being the source of a given message, $P_{apriori}(t = s)$, is constant across the set of possible sources. The probability that t is the true source given an attacker's observation may be calculated using the a priori probabilities and the conditional probabilities of obtaining the attackers's observation by Bayes Theorem:

$$P(\mathrm{OBS}_a | t = s) = P_{cost\,match}(t) \times P_{predecessor\,match}(t) \tag{5.4}$$

$$P(t = s | \mathrm{OBS}_a) = \frac{P(\mathrm{OBS}_a | t = s_m) \times P_{apriori}(t = s)}{\sum\limits_{i \in S} P(\mathrm{OBS}_a | i = s_m) \times P_{apriori}(i = s)} \tag{5.5}$$

Finally, we may use this set of potential source probabilities to compute an effective source anonymity set size based on the entropy of the distribution, using the technique proposed by Serjantoz and Danezis [43]:

$$S = -\sum_{t \in S} P(t = s|\mathrm{OBS}_a)\ log_2(P(t = s|\mathrm{OBS}_a)) \tag{5.6}$$

5.5 Complete Path Anonymity Analysis

A complete Dovetail path contains a series of different path segments. A passive adversary who observes construction of a path segment has full knowledge of the remainder of the segment, and is able to make some assessment of the segment source using the methods presented in Section 5.4. In addition, she may learn further information from observing the return path. We now consider the complete set of information available regarding source and destination identity at each location on a Dovetail path.

5.5.1 Source Identity

At Source. Source is known.

At Source Service Provider. Source is known.

Between Source and Matchmaker. An attacker can identify the preceding AS and cost to the preceding AS, allowing a measurement of the source anonymity set. All subsequent pathlets up to the matchmaker are known, and the fact that these must be present in the source's routing information base may allow a further reduction in the anonymity set. Both measurements decrease in utility with increasing distance from the source.

At Matchmaker. An attacker can identify the preceding AS and cost to the preceding AS, allowing a measurement of the source anonymity set, but distance from the source means this is likely to be a weak form of identification.

Between Matchmaker and Dovetail. An attacker can identify the preceding AS and cost to the preceding AS on the ingress segment, allowing a calculation of the matchmaker anonymity set. However, matchmakers are selected at random, and so there is no correlation between matchmaker and source. This means identifying a set of possible matchmakers does not help identify a set of possible sources. An attacker may measure the total cost from the source to the preceding AS along the concatenation of egress and ingress segments, but distance from the source means this is likely to be a weak form of identification.

From Dovetail to Ingress Target (or Destination when no Ingress Target exists). An attacker can identify the preceding AS and cost from the matchmaker to the preceding AS, but as before there is no correlation between matchmaker identity and source identity so this does not help identify the source. An attacker may measure the total cost from the source to the preceding AS along the data path, but distance from the source means this is likely to be a weak form of identification.

Between Ingress Target and Destination. An attacker can measure the total cost from the source to the preceding AS along the data path. As we will explain shortly, an attacker at these locations knows the approach segment, and therefore the cost from the ingress target to herself. This lets her deduce the cost from the source to the ingress target, but distance from the source means this is likely to be a weak form of identification.

5.5.2 Destination Identity without Pseudonym

At Destination. Destination is known.

At Destination Service Provider. Destination is known.

Between Destination and Matchmaker on Ingress segment. Destination is known from the construction request.

Between Matchmaker and Dovetail on Egress segment. No knowledge of destination.

From Dovetail to Source. An attacker can measure the cost from destination to her own AS using the data return path. If the attacker is able to guess which AS on the egress segment serves as the dovetail, she can measure cost to the dovetail, and therefore infer cost from the destination to the dovetail.

5.5.3 Destination Identity with Pseudonym

At Destination. Destination is known.

At Destination Service Provider. Destination is known.

Between Destination Service Provider and Ingress Target. An attacker can identify the preceding AS and cost to the preceding AS on the return segment, allowing a measurement of the destination anonymity set. Note the pathlets used in the approach segment are not guaranteed to be in the destination's routing information base, and so do not assist in reducing the anonymity set.

Between Ingress Target and Matchmaker on Ingress segment. Ingress target is known from the construction request, and cost from the ingress target to the destination is known from the length of the approach segment.

Between Matchmaker and Dovetail on Egress segment. No knowledge of destination.

From Dovetail to Source. An attacker can measure the cost from destination to her own AS using the data return path. If the attacker is able to guess which AS on the egress segment serves as the dovetail, she can measure cost to the dovetail, and therefore infer cost from the destination to the dovetail. In this case the cost will be the sum of the dovetail to ingress target cost and ingress target to destination cost, and is likely to be a weak form of identification.

5.5.4 Destination–Pseudonym Linkability

At Destination. Association between destination and destination pseudonym is known.

At Destination Service Provider. The destination creates the approach segment by constructing the desired path using a well-known nonce (see Section 4.6.4). The destination's service provider learns the destination identity and approach segment ciphertext during this construction, and so when the same ciphertext is published as a public pseudonym, the service provider may link the pseudonym to its destination.

Between Destination and Ingress Target. Similarly, these nodes learn an anonymity set for the destination during approach segment construction that they may later associate with the pseudonym. The size of this anonymity set increases with distance from the destination.

All other locations. Cost from the ingress target to destination is known from the publicly available approach segment, but no further information is known.

5.5.5 Discussion

Overall, this analysis shows that our design supports the unlinkability of source and destination; locations where the source is easily identified have little information about the destination and vice versa. The dovetail is the closest AS to the source that learns destination identity, and is therefore normally the strongest location for a passive attacker. An AS on the approach segment is able to associate the destination pseudonym with a set of potential destinations, and this set reduces in size as the AS approaches the true destination. This means it is important for a pseudonymous destination to begin his approach segment with a trusted service provider and trusted ASs.

Each point in the preceding sections defining a measurable cost between a known location and an unknown location implies a set of potential identities for the unknown location may be built using the techniques given in Section 5.4. These sets always increase in size with an increasing cost, and therefore the minimum length selected for each segment of the dovetail path serves a purpose in maintaining a particular anonymity property. These constraints are summarized in Table 5.2.

Table 5.2. Path segment length constraints

Segment	**Description**	**Length Requirement**
Egress	Leads from the source to the matchmaker, via the dovetail. Constructed by the source.	Length must be sufficient to conceal source identity from the dovetail.
Ingress	Leads from the matchmaker to the ingress target (or destination), via the dovetail. Constructed by the matchmaker.	Length must be sufficient to conceal destination identity from the source's service provider.
Approach	Leads from the ingress target to the destination. Constructed by a human operator in advance of communication.	Length must be sufficient to conceal destination identity from the ingress target.

An AS that is present on the path at multiple locations may combine the information it learns from each location. For certain strong points in the path, it is important that the AS not be reused at a different location to avoid elevating the capability of an attacker. There are two locations where we apply this constraint:

1. *The dovetail AS must only appear on the egress segment once.* The dovetail is normally the strongest point in the protocol and the minimum cost of the egress segment is set to ensure that source identity is concealed before reaching the dovetail. If the dovetail AS were present earlier in the egress segment it would be able to measure source identity with a cost lower than we intended. Note that this requirement does exclude a fraction of the source's path options, and this information may be used by an attacker to provide a very minor improvement in the effective source anonymity set.
2. *The ingress target AS must only appear on the approach segment once.* We set the minimum cost of the approach segment so the ingress target does not have good visibility of the destination identity. If the ingress target AS were present closer to the final destination it would be able to measure destination identity with a cost lower than we intended.

Any other AS that appears twice in a segment gains no additional information from its second inclusion. An attacker later in the segment cannot detect that a loop has occurred and so cannot use the information to her advantage.

Finally, we note that random selection of matchmaker is effective in isolating the anonymity properties of our system: A node before the matchmaker has complete knowledge of the matchmaker, but this is not helpful in identifying the destination because there is no correlation between the destination and the matchmaker selection. A node after the matchmaker might be able to trace the ingress segment back to learn the matchmaker identity, but again this is not helpful in identifying the source because there is no correlation between source and the matchmaker selection.

5.6 Response Timing Considerations

We have not yet considered the impact of response timing information on our system in detail. Notwithstanding this, we reason that the path diversity used to select each path segment has a strongly beneficial impact on an attacker's ability to identify participants from response timing data: Each potential source of a message segment normally has many thousand possible routes it could have used to reach the destination, and each of these routes has its own latency distribution. The superposition of these distributions blurs the range of possible response times for a source significantly when compared to a system that always uses shortest path routing, and therefore makes distinguishing between different sources harder.

We have also not considered the optimization of system performance based on latency and bandwidth differences between routes. It is very likely that an optimized version of Dovetail would consider geographical distance or latency in its selection of a matchmaker, which in turn would change the anonymity properties of the protocol. We consider this integration of performance and anonymity concerns in response to network latency information to be a rich avenue for further study.

CHAPTER 6

EVALUATION

6.1 Introduction

Our proposal is evaluated primarily by simulation, using a model of the complete Internet at the AS level. This chapter first describes the scope of our simulation and the derivation of our input data. The results are then discussed in two stages, beginning with the construction of a single path segment and then extending to the complete Dovetail protocol. Section 6.5 concludes by estimating a variety of resource requirements for our system.

6.2 Simulation Scope

Our simulation models a network composed of multiple autonomous systems, each containing a maximum of three routing vnodes representing its routing functionality, a single host vnode representing its ability to perform higher level functions such as matchmaking, and zero or more host vnodes each representing the customers of a single Internet service provider. Each pair of ASs is connected by at most one link that represents a data exchange agreement between the two. A pair of pathlets cross over each link, connected to routing vnodes that codify the contractual arrangement between the ASs; customer, provider, or peer. We consider all pathlets within an AS to have a cost of zero and all pathlets between different ASs to have a cost of one, such that the cost of each path is equal to the number of times it moves between ASs. We simulate the exchange of routing information between peer ASs at initialization, with each vnode exporting its entire shortest path tree plus 50% of the shortest path tree size in additional diversity pathlets. This routing exchange leads to a unique routing perspective for each AS, containing all routing vnodes but generally not all pathlets.

While developing our methods and tools, we initially worked with small scale artificial network topologies created using BRITE [44], before transitioning to a full Internet topology based on the CAIDA inferred AS relationship dataset [45]. The CAIDA dataset included an additional *sibling* relationship type, created when two ASs are owned by the same organization in the WHOIS database. When routing policies are extended to allow unrestricted communication over these sibling relationships, it becomes possible to construct arbitrarily long looping paths. Consider the case where a Tier 2 AS, X, has a sibling relationship with a Tier 4 AS, Y, in addition to an indirect provider-customer relationship. A path may be constructed that reaches a maximum tier at X, inferring that all subsequent ASs should be at lower tiers in order to comply with the valley-free routing policy. The path may then be extended down to Y using the customer relationship, followed by the sibling relationship to return to X. This loop may be repeated any number of times, before finally extending the path to a different customer of X. In some cases these infinite paths may be a realistic representation of routing policy, but in others they are merely an unintended consequence of the method by which the sibling relationship was inferred. Path length diversity is a valuable commodity for our protocol, and the presence of these sibling loops was a major contributor to the number of different paths each vnode was able to create. To avoid any optimistic bias, we replaced all sibling relationships with the more restrictive *peer* relationship.

To position the end hosts communicating via our protocol, we consider each AS without customer ASs to be a service provider for end users, and allocate a host vnode to represent these users. Once sibling relationships were reclassified, some portions of the network dropped to less than complete reachability. To address this problem, we removed all host vnodes from any ASs that could no longer reach the entire network. These removals totaled 5.5% of the network size. Ideally, we would model each end user as her own vnode, but accurate ISP customer size data are not available to drive this decomposition from ISPs to users. The available ISP size indicators, such as allocated IPv4 address space, are either coarse or incomplete. Rather than risk skewing our conclusions, we restrict ourselves to measuring anonymity as the number of possible ISPs a source or destination resides within, recognizing that some ISPs are far larger than others.

Each time a path segment must be selected, the source node compiles the set of available routes to the selected destination using a modified depth first search algorithm. Our implementation limits this set to a global maximum cost, and also a maximum number of options at each path cost. For our experiments, the maximum cost is set to 13 (based on the longest distance present in the network), and the maximum number of options per cost is set to 20,000. To explore the relationship between routing policy and path diversity, we consider a mixture of ASs following the strict and loose valley-free routing policies defined in Section 4.3. Figure 6.1 presents the number of different path options available from between randomly selected source and destination hosts, for a range of routing policy ratios.

This figure shows that when all ASs follow the strict valley-free policy the number of available routing options is limited, with a median of only 39 options and less than four options at the first decile. Under these assumptions the Dovetail protocol would still be able to provide a degree of anonymity, but we cannot heavily exercise our principle of anonymity by path diversity. Once even small fractions of loose valley-free ASs are introduced, the number of available options rises dramatically, with a median of 91,000 at 10% loose valley-free, and 112,000 at 20%.

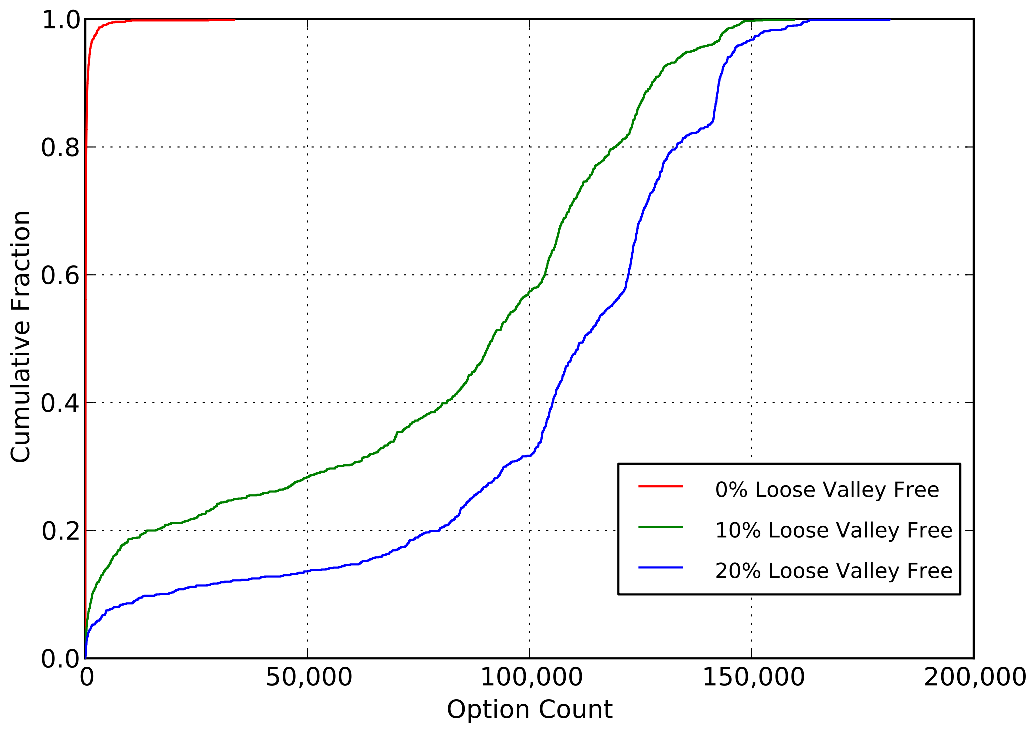


Figure 6.1. Option count distribution under different topology assumptions.

For the remainder of our experiments, we assume a topology in which 10% of the ASs use a loose valley-free routing policy. Previous studies show that strict valley-free routing is not universal in the Internet today [38]. Since any network routing proposal requires the support of Internet infrastructure operators, it is reasonable to assume that some operators would be willing to assist the protocol though a more diverse routing policy. We acknowledge that our selection of 10% as the appropriate proportion is somewhat arbitrary.

6.3 Single Path Segment Performance

Each segment of our complete path is constructed by first selecting a desired cost from the set of *available costs* (i.e. costs for which one or more route options exist), and then selecting a random route that meets this cost. We present results for the following five different path selection algorithms, expressed in terms of the probability $P(\lambda)$ that each will select a particular available cost λ. In all cases k is used to represent a constant specific to the algorithm:

Shortest The shortest possible path is selected in all cases, i.e. $P(\lambda) = 1.0$ if $\lambda = \lambda_{shortest}$, or 0.0 otherwise.

Uniform The path cost is selected uniformly from across all available costs, i.e. $P(\lambda) \propto 1$.

Weighted The probability of selecting an available cost is given by $P(\lambda) \propto 1 - k\lambda$.

Exponential The probability of selecting an available cost is given by $P(\lambda) \propto k^{\lambda}$.

Exponential4 The probability of selecting an available cost is given by $P(\lambda) = 0$ if $\lambda < 4$, and $P(\lambda) \propto k^{\lambda}$ otherwise.

Figure 6.2 illustrates the number of paths selected at each cost when using these algorithms for 2,000 random source destination pairs. The shortest path algorithm shows that the vast majority of host pairs within our model have a shortest path cost of between three and five. In contrast the uniform algorithm offers a near constant cost distribution for costs of four and above (costs below this are less common, because they were only available in a small fraction of the cases considered). The weighted and exponential algorithms both try and reduce the average cost of the algorithm by making shorter paths more common, while avoiding a fixed correlation between observed cost and shortest path cost. The Exponential4 algorithm additionally avoids ever selecting shorter than average paths, since these short paths always reveal that their source and destination are abnormally close.

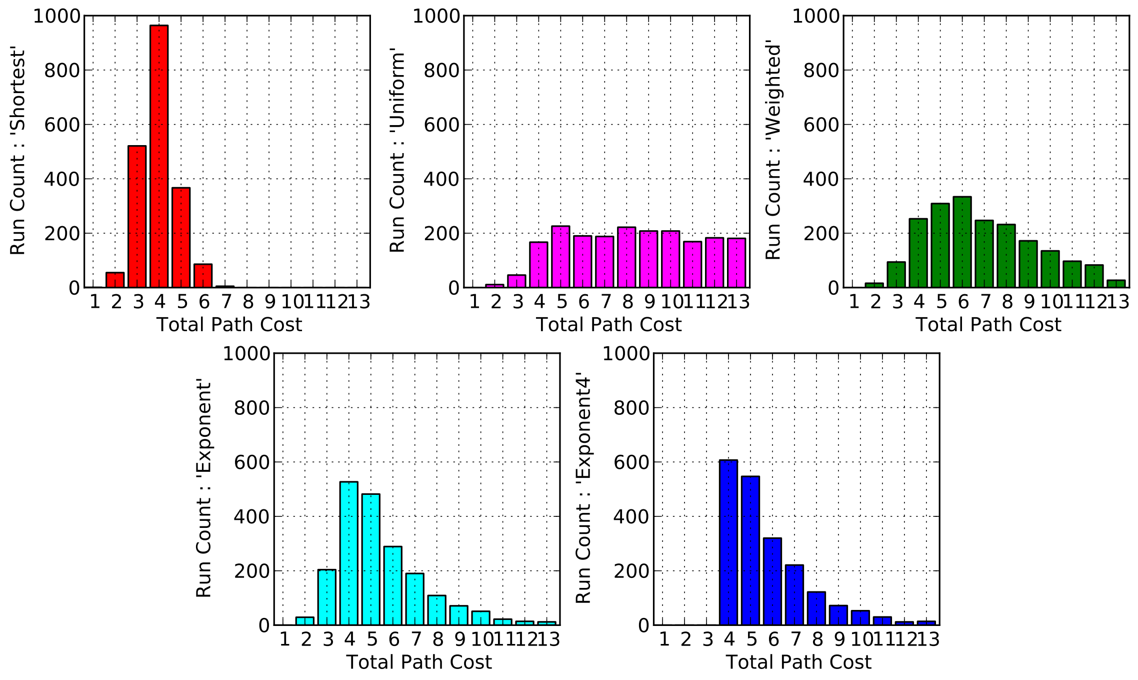


Figure 6.2. Cost distribution for each selection algorithm.

Figure 6.3a expresses this cost data cumulatively, while Figure 6.3b provides a cumulative distribution of the source anonymity set size as measured from the destination, calculated using the technique presented in Section 5.4.1.

These results demonstrate that all the algorithms producing a non-deterministic cost succeed in achieving a meaningful anonymity improvement over the shortest path case, by three bits in the best case, and over one bit for the majority of cases. As we would expect, algorithms that select long paths more frequently achieve a better anonymity, but result in a higher average cost. The move from Exponent to Exponent4 is particularly striking, showing a dramatic improvement in worst case anonymity from the exclusion of short paths, with only a very moderate increase in cost. This Exponent4 algorithm results in an average cost approximately 25% greater than shortest path routing, and yet achieves a complete source anonymity set containing all 2^{15} ISPs in 80% of the tests.

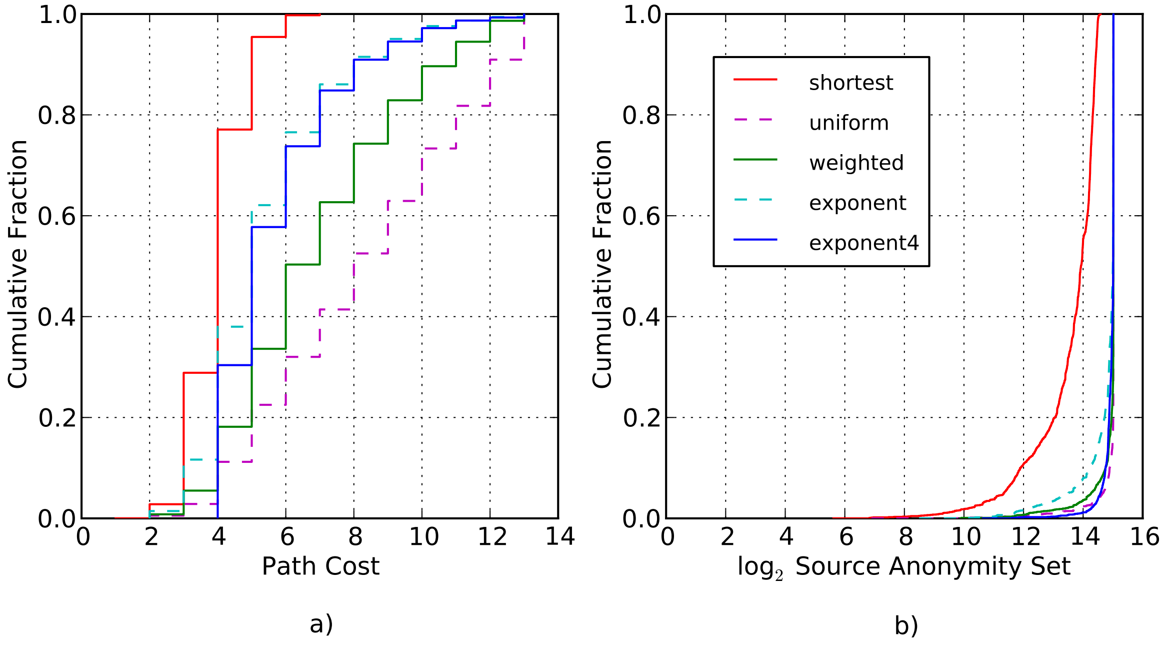


Figure 6.3. a) Cumulative cost distribution for each selection algorithm. b) Cumulative source anonymity distribution for each selection algorithm.

We developed logic within our simulation to also calculate the higher fidelity *effective* anonymity set size using the entropy method developed in Section 5.4.2. However, this method requires calculation of every path option from every potential source and, despite heavy optimization, proved impractical for our complete Internet model. An alternative approach would be to scale the network topology to some fraction of the complete network size, verify the scaled topology displays similar structural characteristics to the complete topology, and then execute the entropy calculations in this smaller environment. We would expect to see a small reduction in the absolute anonymity using the entropy method, with a stronger impact on the exponent and weighted methods, but not a change in the overall structure of our results.

6.4 Complete Path Performance

We now extend our evaluation to consider a complete Dovetail path. Our protocol includes a variety of configurable parameters that may be used to trade performance against privacy, allowing users to control their degree of anonymity in accordance with Requirement 5. These parameters provide a sliding scale, but our objective here is to demonstrate the maximum end of this scale, showing that the protocol can provide near complete anonymity against a local adversary. For many users this would be far stronger than necessary, and therefore in many cases the parameters would be tuned for more performance and less anonymity than we use here. Table 6.1 summarizes the configurable parameters and the values we select for our evaluation.

Table 6.1. Dovetail parameter selection

Parameter	**Description**	**Selection**
Source to Matchmaker Algorithm	Controls source anonymity measured at the dovetail.	**Exponent6.** After accounting for dovetail-matchmaker cost, effectively delivers Exponent4 at the Dovetail. Exponent4 is previously shown to provide near complete anonymity.
Dovetail to Matchmaker Cost	Low values control ability of matchmaker to probe for joins, high values decouple dovetail and matchmaker identities.	**Two AS hops.** Provides very strong limits on matchmaker capability without requiring that dovetail and matchmaker are immediate neighbors.
Dovetail to Ingress Target Algorithm	Controls destination anonymity measured at the source ISP.	**Exponent4.** Previously shown to provide near complete anonymity.
Ingress Target to Destination Cost	Controls linkability between destination and destination pseudonym.	**Five AS hops.** Optimized by experiment to give near complete unlinkability.

Our complete path experiments select a source and destination host at random, constructs a dovetail path between this pair, and then measure the source and destination anonymity sets observable by an attacker at each location in the path. The measurement is based on the analysis presented in Section 5.5, and the sets are expressed as a number of possible ISPs. To minimize the number of times the same AS appears on both the ingress and egress segments, we simulate the matchmaker returning up to eight randomly selected ingress paths that the source may select from. In cases where all eight options reuse an AS, we continue with the last option despite the duplication. This condition occurred in 23% of our experiments. In a practical implementation, we expect a heuristic could be developed to select dovetail vnodes without a strong probability of duplicate ASs. If all options contained duplicates even in this case, a source could select a different matchmaker rather than continue with a duplicated, and therefore unusually strong, AS.

Our random selection of matchmaker effectively decouples the source and destination anonymity sets, and therefore we can also consider the *source-destination unlinkability*, i.e. the number of potential source-destination pairs associated with an observed connection, to be the product of source anonymity set size and destination anonymity set size. Figure 6.4 presents the cumulative distribution of these three properties at a series of key locations for paths that do not use a destination pseudonym, Figure 6.5 presents the same information for paths that do use a destination pseudonym.

Overall these results support our assertion that Dovetail is able to provide near complete topological anonymity against a local attacker. From Figure 6.4, the source identity is known at the source and the source's ISP, but each successive step adds ambiguity, such that by the dovetail AS, the source anonymity set is nearly equal to network size in 80% of cases. By the final destination, source anonymity is near

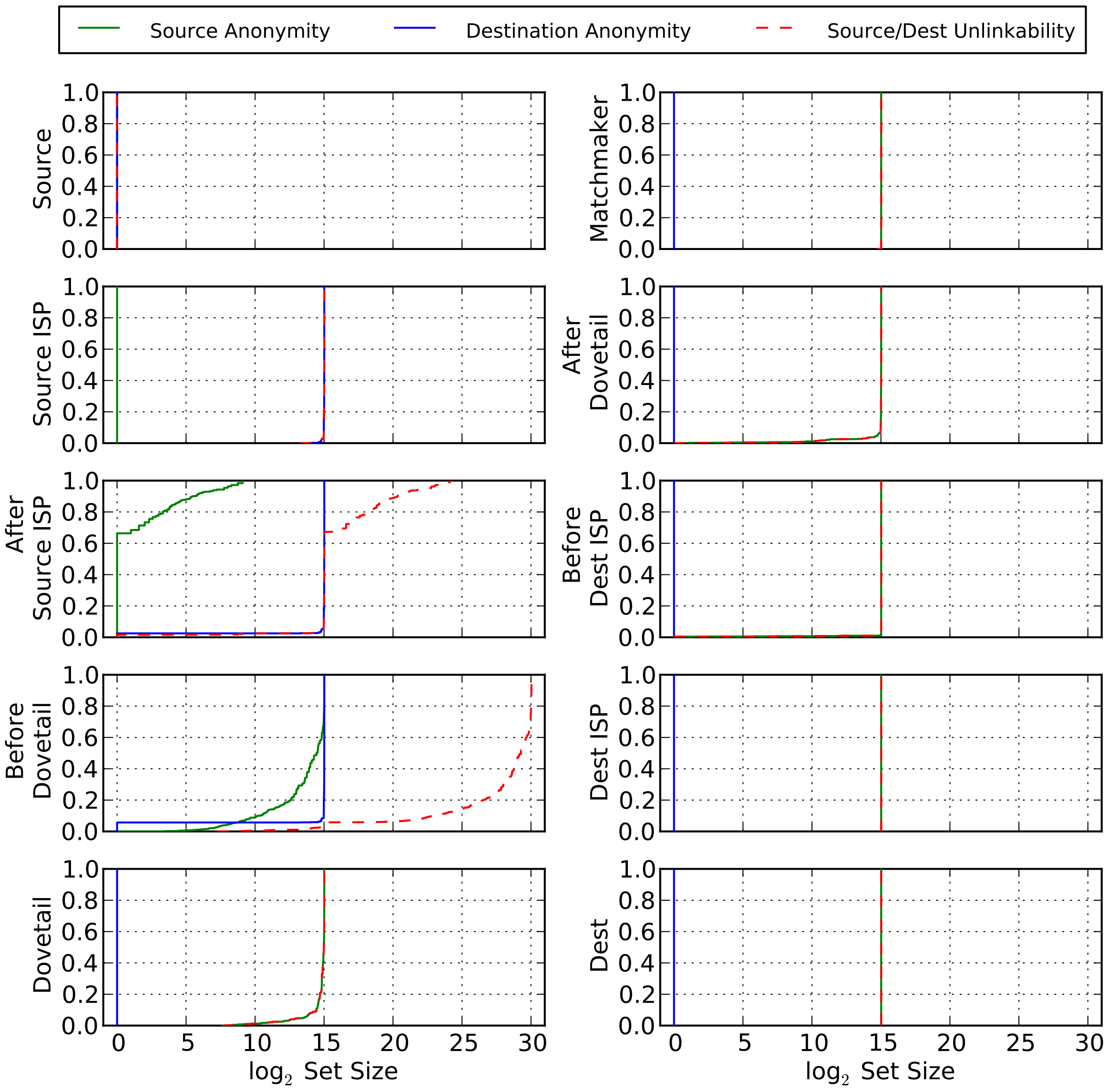


Figure 6.4. Source and destination anonymity without destination pseudonym.

perfect. Destination identity is known at the dovetail and all subsequent locations, but locations prior to the dovetail have only very limited knowledge of destination identity (a cost between the dovetail and destination, which our parameter settings ensure is at least four) and therefore are unable to calculate a meaningful destination identity. The AS immediately preceding the dovetail is most likely to be duplicated in

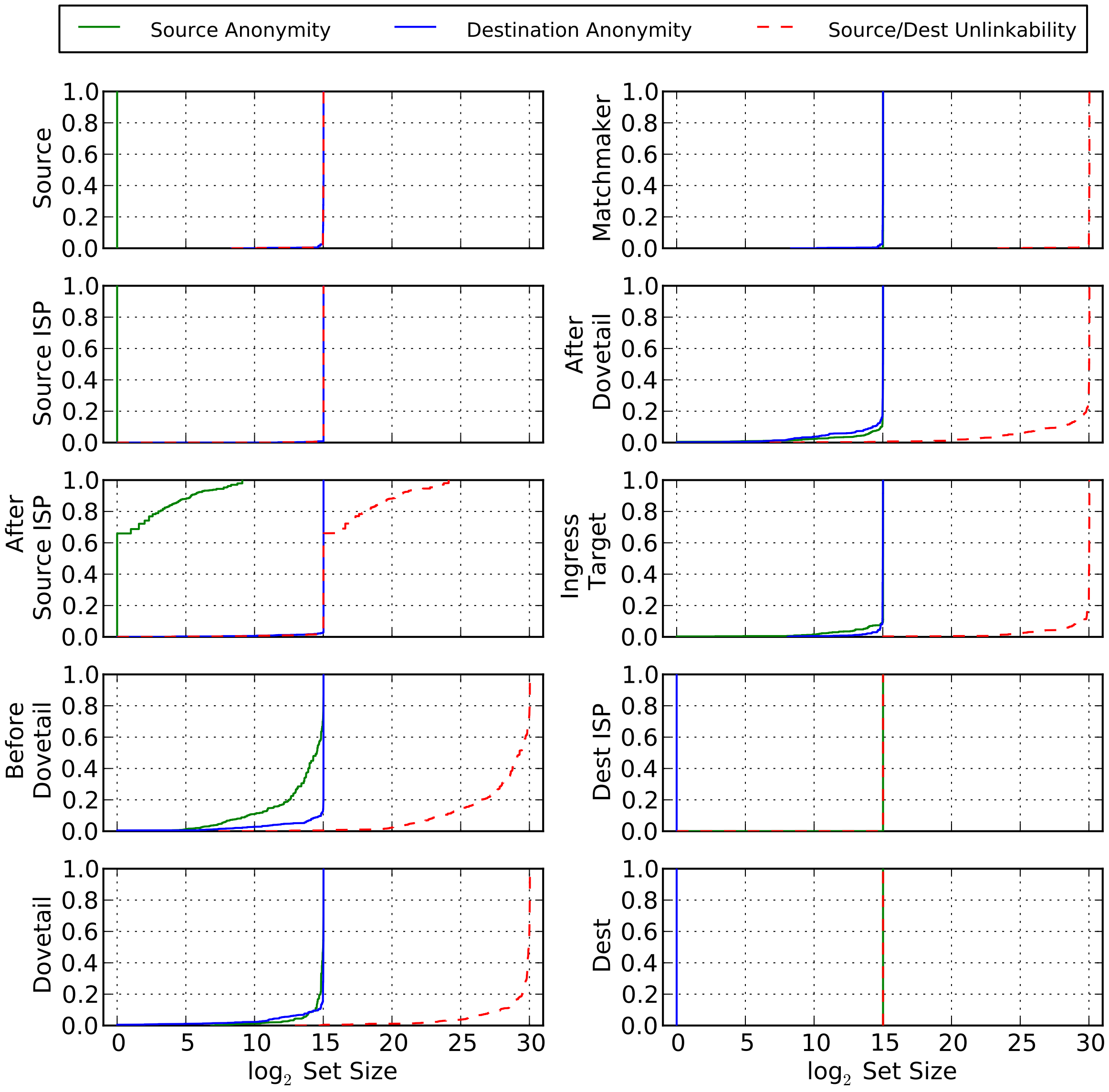


Figure 6.5. Source and destination anonymity with destination pseudonym.

ingress and egress segments, because it is adjacent to an AS that is always present in both. In our experiment, the AS before the dovetail was duplicated in approximately five percent of the cases, and therefore had complete knowledge of destination in these cases. In terms of source-destination linkability, we see that only the source is able to clearly link source and destination. For around 20% of cases the dovetail is able to calculate some information regarding source identity, but this is limited to around one thousand possible source ISPs, each containing many different users. This is sufficient to meet Requirement 1.

For the same number of destinations, linkability between source and destination pseudonym will be weaker than linkability between source and destination without a pseudonym. The path from source to ingress target is created using the same rules as the path from source to destination without an ingress target, and therefore accumulates ambiguity in the same way. However, locations prior to the dovetail are unable to measure a dovetail to ingress target cost in the same way they that can measure a dovetail to destination target cost. Through this argument, we also meet Requirement 2.

From Figure 6.5 we see that, when using a pseudonym, the destination identity is effectively concealed from all locations prior to the ingress target, including the source. When an AS is used on both the approach and the ingress segments, it may calculate a partial destination identity, but we previously accepted that a destination must explicitly trust the ASs it places on the approach path. These results are therefore sufficient to meet Requirement 3.

In Figure 6.6, we present the cumulative cost distribution for a dovetail connection, with and without a destination pseudonym, and with traditional shortest path routing included for reference.

This figure shows that a Dovetail path passes through approximately 3.5 times the number of ASs as a shortest path route when a destination pseudonym is used, and 2.5 times when a destination pseudonym is not used. Although these figures may initially seem high, when compared with the prevailing option for anonymity today they are modest: an anonymous circuit in Tor typically passes through three relays for a total of four IP paths. When relay locations are selected randomly we

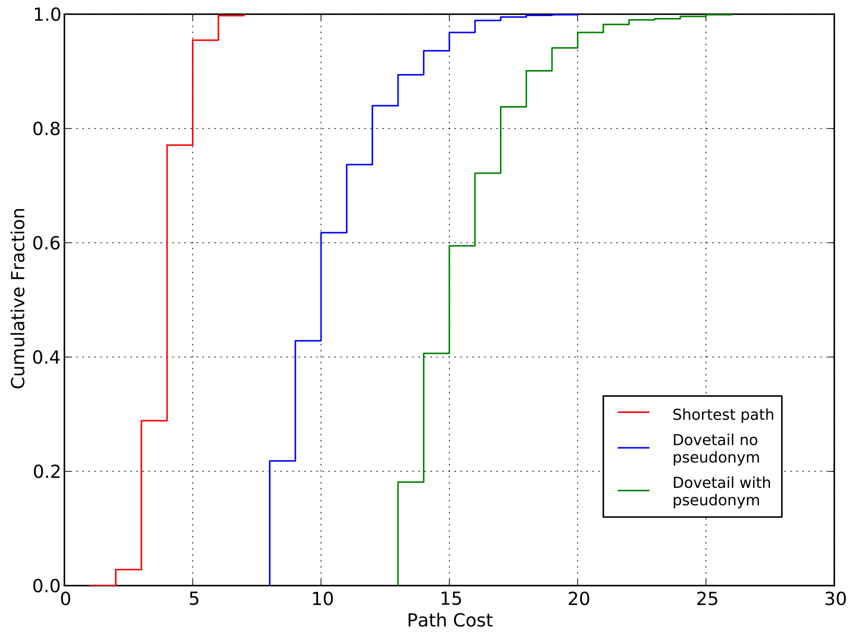


Figure 6.6. Cumulative cost distribution for complete path.

would expect an average of four times the shortest path cost, in addition to queueing latencies in each relay. We note that destination pseudonyms are likely to be the exception rather than the norm, since most organizations hosting Internet services have no motivation to conceal their location. Finally, our cost penalty can be traded for reduced anonymity, and the figures we present here are weighted heavily in favor of anonymity at the expense of latency; we expect a typical user to settle on a more balanced and therefore lower cost approach.

6.5 Resource Utilization

We have not created a network implementation of the Dovetail protocol, since our objective is to motivate inclusion of privacy in future protocol designs, rather than propose a mature protocol suitable for near-term implementation. In this section, we consider a variety of different resource requirements to demonstrate that implementation would not be impractical.

Host memory utilization With Dovetail, as with pathlet routing, each host is capable of generating routes and must therefore maintain a model of the Internet. We assume that the terminal pathlets leading to each end host would be requested on demand rather than distributed globally, and that pathlets managing the internal routing within an AS remain internal to that AS. In the 2012 dataset we use there are 252,666 externally visible pathlets, and on average each vnode knows 22% of these. The maximum size of a Forwarding ID is four bytes, leading to an average memory requirement of 680kB in each host.

Router memory utilization Routers need only store the vnode forwarding table in memory, which scales with the number of local peers and not the size of Internet as in BGP4. This reduction in forwarding memory is one of the most significant advantages of the pathlet routing proposal, and is one that we retain. All information required to forward a packet is carried by the packet itself, and therefore a router need not store any information per connection.

Router latency After a connection has been established, the only cryptographic operation required to forward a packet is decryption of a single word using a symmetric cipher. This is the same task performed by LAP, and Hsiao et al. measure an additional latency of under one microsecond in a software-based implementation of their system [21], noting that a dedicated hardware implementation would offer higher performance.

Transmission efficiency The message header in source-controlled routing must specify a complete path rather than only an endpoint, and this higher information content can lead to larger header sizes and a reduced efficiency. When Dovetail is used without a destination pseudonym, the average header length is 81 bytes. This is approximately double the 40 bytes required with IPv6. Packet headers scale with the number of transit ASs, and so tuning for lower latency would also improve transmission efficiency.

CHAPTER 7

CONCLUSION

Recent advances have provided both the mechanisms and the financial incentives for organizations to collect, correlate, and monetize Internet user information, making Internet privacy a pressing social issue. Anonymity research to date has logically focused on overlay networks that may be deployed without changes in the Internet infrastructure, but these solutions are inherently inefficient, requiring multiple layer 3 connections to conceal the network identity of participants. We prefer a more natural solution: control each form of identity at the level in the protocol stack where it is defined. This leads us to conclude that a layer 3 routing protocol should not expose a globally unique network identity in order to perform its function of routing data.

We do not advocate an Internet without identity. Rather, we propose that identity exposure be reserved for substantial relationships and avoided for ephemeral relationships. To use a physical analogue: One does not need to be identified to listen to a radio advertisement, to read a newspaper, or to drive between neighborhoods, but one usually does need to be identified when meeting a new friend or entering into a financial contract. As society moves online, IP is fast becoming the ether within which we live our digital lives, and it seems disproportionate that every interaction passing through this medium carries an unavoidable identity, no matter how the transaction was initiated or what purpose it serves.

In this paper we have presented Dovetail, a layer 3 routing protocol suitable for use in a next-generation Internet, and have demonstrated that it provides a workable solution for anonymity at the routing protocol layer. The overhead is approximately 2.5 times that of shortest path routing when configured to provide near complete anonymity against our chosen attacker, and we include mechanisms to exchange anonymity for performance. We have demonstrated key aspects of the feasibility and effectiveness of this direction, and hope this this motivates serious consideration of privacy as a requirement in the development of other next-generation routing protocols.

This research direction is still in its infancy, and much work remains to be done. We believe the most pressing matter is to develop diversity in the solution space by considering how privacy-preserving features might be integrated into other leading layer 3 proposals. Beyond this, there are many social and organizational issues to explore: If we wish to provide users with control over their anonymity, how can we explain the choices and their consequences in an accessible manner? How can user interaction be structured around policies rather than prompting for each transaction? Given freedom, what proportion of Internet users would select each level of anonymity, and for which transactions? These user-side interactions drive the network implications: How much additional load would be created on the network, and what characteristics would this additional traffic display? Could the current contractual arrangements between service providers naturally expand to allow greater transparency in Internet routing and sufficient routing diversity? What incentives would be necessary for service providers to adopt the new technologies and policies we propose?

Finally, further work remains with Dovetail itself: Our higher fidelity entropy-based anonymity analysis should be completed, and then we hope to integrate network timing information into both our path selection algorithms and anonymity assessments. Acquiring representative timing information to drive this integration is a challenge, but large scale network research testbeds, such as GENI, offer hope of meeting this need.

APPENDIX A

PACKET LEVEL SIMULATION

A.1 Introduction

The following tables illustrate a sequence of data packets used to create and then exchange data over a Dovetail connection, as produced by our simulation. Each table represents the packet in transit between a pair of vnodes, and the text between the tables define the alterations made by each vnode as it processes the packet. Each table row defines a portion of the packet, with the hexadecimal data in the left column, a description of the portion in the middle column, and the name of the header segment in the right column. Portions vary in length, and may therefore contain any number of hex digits.

In this particular example a destination pseudonym is not used, and the special roles in the protocol are played by the vnodes listed in the table below:

Source	H19114
Dovetail	N1250
Matchmaker	M6736
Destination	H9142

A.2 Construction Request

During connection construction, each new autonomous system (AS identifiers begin with the character 'E') constructs an encrypted version of the path through the AS in each direction and appends this to the Encrypted header segment. Each new autonomous system also constructs an encrypted join entry defining the incoming link and the location of its encrypted join segment, and appends this to the Join header segment. If an AS detects that it has previously created a join segment, it will use this information to modify the original encrypted segment to route from the original incoming link to the new outgoing link and then delete all encrypted segments and join segments after the original. This condition may only occur at the Dovetail, due to the use of a chained hash on N2.

Outgoing packet as initially constructed

02	Type	Type
15	Size	Unencrypted
0	a	
4	b	
C0EA	c	
C020	d	
C056	e	
C1EC	f	
0	g	
16	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
00	Tail	
12	Size	Join
D0D6C977FA651337	H(N2,0)	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Received packet at N37568 in E18621 over link 0

Adding new segment to traverse E18621(0x48BD) E(BD080002)

Stripping first ForwardingEntry: 0

02	Type	Type
14	Size	Unencrypted
4	b	
C0EA	c	
C020	d	
C056	e	
C1EC	f	
0	g	
1E	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
08	Tail	
12	Size	Join
D0D6C977FA651337	H(N2,0)	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Forwarding packet to N2578 over link 34059

Received packet at N2578 in E1288 over link 34059

Adding new join segment to join in E1288(0x0508) E(1C00850B)

Adding new segment to traverse E1288(0x0508) E(08080843)

Stripping first ForwardingEntry: 4

02	Type	Type
13	Size	Unencrypted
C0EA	c	
C020	d	
C056	e	
C1EC	f	
0	g	
26	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
08	Tail	
1A	Size	Join
D6BEB7A033CFA2C7	H(N2,1)	
E2B5F40F	E1288	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Forwarding packet to N2649 over link 34063

Received packet at N2649 in E1321 over link 34063

Adding new join segment to join in E1321(0x0529) E(2400850F)

Adding new segment to traverse E1321(0x0529) E(290E08C0EAC04D)

Stripping first ForwardingEntry: C0EA

02	Type	Type
0F	Size	Unencrypted
C020	d	
C056	e	
C1EC	f	
0	g	
34	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
0E	Tail	
22	Size	Join
533D43E9C11432DF	H(N2,2)	
E2B5F40F	E1288	
2374854D	E1321	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Forwarding packet to N60 over link 173

Received packet at N60 in E29 over link 173

Adding new join segment to join in E29(0x001D) E(320000AD)

Adding new segment to traverse E29(0x001D) E(1D0E0EC020C046)

Stripping first ForwardingEntry: C020

02	Type	Type
0B	Size	Unencrypted
C056	e	
C1EC	f	
0	g	
42	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
0E	Tail	
2A	Size	Join
C4F6E0A5FB5F5909	H(N2,3)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Forwarding packet to N1250 over link 135

Received packet at N1250 in E627 over link 135

Adding new join segment to join in E627(0x0273) E(40000087)

Adding new segment to traverse E627(0x0273) E(730E0EC056C03E)

Stripping first ForwardingEntry: C056

02	Type	Type
07	Size	Unencrypted
C1EC	f	
0	g	
50	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D4F595B	E627	
0E	Tail	
32	Size	Join
F51F00E3C1D3B26E	H(N2,4)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Forwarding packet to N3564 over link 14587

Adding new join segment to join in E1776(0x06F0) E(4E0038FB)

Adding new segment to traverse E1776(0x06F0) E(F00E0EC1ECC1F2)

Stripping first ForwardingEntry: C1EC

02	Type	Type
03	Size	Unencrypted
0	g	
5E	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D4F595B	E627	
E46694C05EA3AB	E1776	
0E	Tail	
3A	Size	Join
1D3FE24FCCF3A4AF	H(N2,5)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
FD173268	E1776	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Forwarding packet to N14930 over link 40378

Received packet at N14930 in E7400 over link 40378

Adding new join segment to join in E7400(0x1CE8) E(5C009DBA)

Adding new segment to traverse E7400(0x1CE8) E(E80A0E0085)

Stripping first ForwardingEntry: 0

02	Type	Type
02	Size	Unencrypted
68	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D4F595B	E627	
E46694C05EA3AB	E1776	
FCA47CC911	E7400	
0A	Tail	
42	Size	Join
1DDEDDEFE3C8BB03	H(N2,6)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
FD173268	E1776	
45F794F1	E7400	
1CBB45...27D979	E(M6736|N1250|H9142|H(N2,2)|2)	Payload

Received packet at M6736 in E7400 over link 0

Continuing path construction from matchmaker M6736 to target H9142

02	Type	Type
17	Size	Unencrypted
85	m	
C1F2	n	
C006	o	
83	p	
C088	q	
C29D	r	
1	s	
68	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D4F595B	E627	
E46694C05EA3AB	E1776	
FCA47CC911	E7400	
0A	Tail	
42	Size	Join
533D43E9C11432DF	H(N2,2)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
FD173268	E1776	
45F794F1	E7400	

Received packet at N14929 in E7400 over link 0

Stripping first ForwardingEntry: 85

02	Type	Type
15	Size	Unencrypted
C1F2	n	
C006	o	
83	p	
C088	q	
C29D	r	
1	s	
68	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D4F595B	E627	
E46694C05EA3AB	E1776	
FCA47CC911	E7400	
0A	Tail	
42	Size	Join
533D43E9C11432DF	H(N2,2)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
FD173268	E1776	
45F794F1	E7400	

Forwarding packet to N3564 over link 40378

Received packet at N3564 in E1776 over link 40378

Adding new join segment to join in E1776(0x06F0) E(66009DBA)

Adding new segment to traverse E1776(0x06F0) E(F00E0AC1F2C1EC)

Stripping first ForwardingEntry: C1F2

02	Type	Type
11	Size	Unencrypted
C006	o	
83	p	
C088	q	
C29D	r	
1	s	
76	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D4F595B	E627	
E46694C05EA3AB	E1776	
FCA47CC911	E7400	
3823FD1C61AB05	E1776	
0E	Tail	
4A	Size	Join
C4F6E0A5FB5F5909	H(N2,3)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
FD173268	E1776	
45F794F1	E7400	
83D68652	E1776	

Forwarding packet to N1250 over link 14587

Received packet at N1250 in E627 over link 14587

Found and replaced previous traversal through E627(0x0273)

New traversal and join entries are: E(730E0EC006C03E)/E(40000087)

Stripping first ForwardingEntry: C006

02	Type	Type
0D	Size	Unencrypted
83	p	
C088	q	
C29D	r	
1	s	
50	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
0E	Tail	
32	Size	Join
F51F00E3C1D3B26E	H(N2,4)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	

Forwarding packet to N578 over link 10568

Received packet at N578 in E291 over link 10568

Adding new join segment to join in E291(0x0123) E(4E002948)

Adding new segment to traverse E291(0x0123) E(230A0E8395)

Stripping first ForwardingEntry: 83

02	Type	Type
0B	Size	Unencrypted
C088	q	
C29D	r	
1	s	
5A	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A	Tail	
3A	Size	Join
1D3FE24FCCF3A4AF	H(N2,5)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
B3354312	E291	

Forwarding packet to N9957 over link 10551

Received packet at N9957 in E4930 over link 10551

Adding new join segment to join in E4930(0x1342) E(58002937)

Adding new segment to traverse E4930(0x1342) E(420E0AC088C05D)

Stripping first ForwardingEntry: C088

02	Type	Type
07	Size	Unencrypted
C29D	r	
1	s	
68	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
0E	Tail	
42	Size	Join
1DDEDDEFE3C8BB03	H(N2,6)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
B3354312	E291	
3784656B	E4930	

Forwarding packet to N4073 over link 48578

Received packet at N4073 in E2028 over link 48578

Adding new join segment to join in E2028(0x07EC) E(6600BDC2)

Adding new segment to traverse E2028(0x07EC) E(EC0E0EC29DC00B)

Stripping first ForwardingEntry: C29D

02	Type	Type
03	Size	Unencrypted
1	s	
76	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
0E	Tail	
4A	Size	Join
C9179E73C2CAA00F	H(N2,7)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
B3354312	E291	
3784656B	E4930	
76259D79	E2028	

Forwarding packet to N17896 over link 47537

Received packet at N17896 in E8874 over link 47537

Adding new join segment to join in E8874(0x22AA) E(7400B9B1)

Adding new segment to traverse E8874(0x22AA) E(AA080E13)

Stripping first ForwardingEntry: 1

02	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
00	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
52	Size	Join
204AEE79823F8E61	H(N2,8)	
E2B5F40F	E1288	
2374854D	E1321	
412A5E33	E29	
4E2C2094	E627	
B3354312	E291	
3784656B	E4930	
76259D79	E2028	
1E7678B4	E8874	

A.3 Construction Response

When the path construction packet reaches its destination, the destination vnode changes the packet type to ‘encrypted data response’, removes the join header segment, appends a payload indicating the request was successful, and returns the packet in the direction it arrived. At each step along the return path, a receiving AS uses the offset field to locate its encrypted offset, and decrypts the segment to learn the return path through the AS. This partial path is added to the unencrypted section and then the packet is processed as normal through any remaining vnodes in the AS.

```
Received packet at H9142 in E8874 over link 0
Removing previous join segment
Changing packet type to RETURN_ENCRYPTED
```

05	Type	Type
02	Size	Unencrypted
7E	Size	
74	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	Encrypted
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

Received packet at N17895 in E8874 over link 0

Decrypting FIDs for transit across E8874, and adding to unencrypted section

Stripping first ForwardingEntry: 3

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
66	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	'MADE'	Payload

Forwarding packet to N4072 over link 47537

Received packet at N4072 in E2028 over link 47537

Decrypting FIDs for transit across E2028, and adding to unencrypted section

Stripping first ForwardingEntry: C00B

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
58	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

Forwarding packet to N9957 over link 48578

Received packet at N9957 in E4930 over link 48578

Decrypting FIDs for transit across E4930, and adding to unencrypted section

Stripping first ForwardingEntry: C05D

05	Type	Type
02	Size	Unencrypted
7E	Size	
4E	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	Encrypted
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

Forwarding packet to N578 over link 10551

Received packet at N578 in E291 over link 10551

Decrypting FIDs for transit across E291, and adding to unencrypted section

Stripping first ForwardingEntry: 95

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
40	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

Forwarding packet to N1250 over link 10568

Received packet at N1250 in E627 over link 10568

Decrypting FIDs for transit across E627, and adding to unencrypted section

Stripping first ForwardingEntry: C03E

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
32	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

Forwarding packet to N60 over link 135

Received packet at N60 in E29 over link 135

Decrypting FIDs for transit across E29, and adding to unencrypted section

Stripping first ForwardingEntry: C046

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
24	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

Forwarding packet to N2650 over link 173

Received packet at N2650 in E1321 over link 173

Decrypting FIDs for transit across E1321, and adding to unencrypted section

Stripping first ForwardingEntry: C04D

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
1C	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	'MADE'	Payload

Forwarding packet to N2579 over link 34063

Received packet at N2579 in E1288 over link 34063

Decrypting FIDs for transit across E1288, and adding to unencrypted section

Stripping first ForwardingEntry: 3

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
14	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

Forwarding packet to N37569 over link 34059

Received packet at N37569 in E18621 over link 34059

Decrypting FIDs for transit across E18621, and adding to unencrypted section

Stripping first ForwardingEntry: 2

05	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
14	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4D414445	‘MADE’	Payload

A.4 Outbound Data

When the source is notified of a successful connection by way of the encrypted data response, it may use the received encrypted header segment to send data. At each step along the outbound path, a receiving AS uses the offset field to locate its encrypted offset, and decrypts the segment to learn the forward path through the AS. This partial path is added to the unencrypted section and then the packet is processed as normal through any remaining vnodes in the AS.

```
Received packet at H19114 in E18621 over link 0
Arriving at host node is end condition
Received data return packet of length: 138
Preparing an outgoing data packet
Changing packet type to OUTBOUND_ENCRYPTED
Removing 8 nibbles of previous payload
```

Received packet at N37568 in E18621 over link 0

Decrypting FIDs for transit across E18621, and adding to unencrypted section

Stripping first ForwardingEntry: 0

04	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
1C	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	‘OutgoingData’	Payload

Forwarding packet to N2578 over link 34059

Received packet at N2578 in E1288 over link 34059

Decrypting FIDs for transit across E1288, and adding to unencrypted section

Stripping first ForwardingEntry: 4

04	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
24	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	‘OutgoingData’	Payload

Forwarding packet to N2649 over link 34063

Received packet at N2649 in E1321 over link 34063

Decrypting FIDs for transit across E1321, and adding to unencrypted section

Stripping first ForwardingEntry: C0EA

04	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
32	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	‘OutgoingData’	Payload

Forwarding packet to N60 over link 173

Received packet at N60 in E29 over link 173

Decrypting FIDs for transit across E29, and adding to unencrypted section

Stripping first ForwardingEntry: C020

04	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
40	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	‘OutgoingData’	Payload

Forwarding packet to N1250 over link 135

Received packet at N1250 in E627 over link 135

Decrypting FIDs for transit across E627, and adding to unencrypted section

Stripping first ForwardingEntry: C006

04	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
4E	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	‘OutgoingData’	Payload

Forwarding packet to N578 over link 10568

Received packet at N578 in E291 over link 10568

Decrypting FIDs for transit across E291, and adding to unencrypted section

Stripping first ForwardingEntry: 83

04	Type	Type
02	Size	Unencrypted
7E	Size	
58	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	Encrypted
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	‘OutgoingData’	Payload

Forwarding packet to N9957 over link 10551

Received packet at N9957 in E4930 over link 10551

Decrypting FIDs for transit across E4930, and adding to unencrypted section

Stripping first ForwardingEntry: C088

04	Type	Type
02	Size	Unencrypted
7E	Size	Encrypted
66	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	'OutgoingData'	Payload

Forwarding packet to N4073 over link 48578

Received packet at N4073 in E2028 over link 48578

Decrypting FIDs for transit across E2028, and adding to unencrypted section

Stripping first ForwardingEntry: C29D

04	Type	Type
02	Size	Unencrypted
7E	Size	
74	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	Encrypted
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	'OutgoingData'	Payload

Forwarding packet to N17896 over link 47537

Received packet at N17896 in E8874 over link 47537

Decrypting FIDs for transit across E8874, and adding to unencrypted section

Stripping first ForwardingEntry: 1

04	Type	Type
02	Size	Unencrypted
7E	Size	
7C	Offset	
E1546A2C4791ABE9	N1	
3179F022	E18621	
801EA315	E1288	
87A55FA0B1F0ED	E1321	
1ADF8EB41F2F20	E29	
C79BEC1D1F771E	E627	
D2097B64B9	E291	Encrypted
0A1D0CA61CF443	E4930	
F353676F558E5B	E2028	
FAE69F19	E8874	
08	Tail	
4F7574676F696E6744617461	‘OutgoingData’	Payload

Received packet at H9142 in E8874 over link 0

REFERENCES

[1] J. R. Mayer and J. C. Mitchell, "Third-party web tracking: Policy and technology," in *Security and Privacy (SP), 2012 IEEE Symposium on.* IEEE, 2012, pp. 413–427.

[2] J. Mikians, L. Gyarmati, V. Erramilli, and N. Laoutaris, "Detecting price and search discrimination on the internet," in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks.* ACM, 2012, pp. 79–84.

[3] *Information technology – Open Systems Interconnection – Basic Reference Model: The Basic Model*, ISO Std. 7498-1, 1994.

[4] M. Reiter and A. Rubin, "Crowds: Anonymity for web transactions," *ACM Transactions on Information and System Security*, vol. 1, no. 1, June 1998.

[5] P. Boucher, A. Shostack, and I. Goldberg, "Freedom systems 2.0 architecture," Zero Knowledge Systems, Inc.," White Paper, December 2000.

[6] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[7] (2013) Tor metrics portal: Users. [Online]. Available: https://metrics.torproject.org/users.html

[8] S. Paul, J. Pan, and R. Jain, "Architectures for the future networks and the next generation internet: A survey," *Computer Communications*, vol. 34, no. 1, pp. 2–42, 2011.

[9] (2008) NSF NeTS FIND initiative. [Online]. Available: http://www.nets-find.net/index.php

[10] (2011) FIRE home page. [Online]. Available: http://cordis.europa.eu/fp7/ict/fire/home_en.html

[11] (2008) New generation network architecture AKARI conceptual design (ver1.1). [Online]. Available: http://akari-project.nict.go.jp/eng/concept-design/AKARI_fulltext_e_translated_version_1_1.pdf

[12] F. Papadopoulos, D. Krioukov, M. Bogua, and A. Vahdat, "Greedy forwarding in dynamic scale-free networks embedded in hyperbolic metric spaces," in *INFOCOM, 2010 Proceedings IEEE.* IEEE, 2010, pp. 1–9.

[13] B. Bhattacharjee, K. Calvert, J. Griffioen, N. Spring, and J. P. Sterbenz, "Post-modern internetwork architecture," *NSF Nets FIND Initiative*, 2006.

[14] P. B. Godfrey, I. Ganichev, S. Shenker, and I. Stoica, "Pathlet routing," in *ACM SIGCOMM*, 2009, pp. 111–122.

[15] D. Farinacci, D. Lewis, D. Meyer, and V. Fuller. (2013, January) The locator/ID separation protocol (LISP). RFC 6830. [Online]. Available: http://tools.ietf.org/html/rfc6830

[16] X. Yang and D. Wetherall, "Source selectable path diversity via routing deflections," in *ACM SIGCOMM Computer Communication Review*, vol. 36. ACM, 2006, pp. 159–170.

[17] X. Yang, "NIRA: A new internet routing architecture," in *Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture (FDNA '03).* ACM, 2003, pp. 301–312.

[18] X. Zhang, H.-C. Hsiao, G. Hasker, H. Chan, A. Perrig, and D. G. Andersen, "SCION: Scalability, control, and isolation on next-generation networks," in *Security and Privacy (SP), 2011 IEEE Symposium on.* IEEE, 2011, pp. 212–227.

[19] (2011) GENI at a glance. [Online]. Available: http://www.geni.net/wp-content/uploads/2011/06/GENI-at-a-Glance-1Jun2011.pdf

[20] A. Pfitzmann and M. Hansen, “A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management,” http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf, Aug. 2010, v0.34. [Online]. Available: http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.34.pdf

[21] H.-C. Hsiao, T.-J. Kim, A. Perrig, A. Yamada, S. C. Nelson, M. Gruteser, and W. Meng, “LAP: Lightweight anonymity and privacy,” in *Security and Privacy (SP), 2012 IEEE Symposium on.* IEEE, 2012, pp. 506–520.

[22] P. Eckersley, “How unique is your web browser?” in *Privacy Enhancing Technologies.* Springer, 2010, pp. 1–18.

[23] A. Soltani, S. Canty, Q. Mayo, L. Thomas, and C. J. Hoofnagle, “Flash cookies and privacy,” *SSRN eLibrary*, 2009.

[24] R. Dingledine and S. J. Murdoch. (2009) Performance improvements on Tor or, why Tor is slow and what we’re going to do about it. [Online]. Available: http://www.torproject.org/press/presskit/2009-03-11-performance.pdf

[25] R. Snader and N. Borisov, “A tune-up for Tor: Improving security and performance in the Tor network,” in *Proceedings of the Network and Distributed Security Symposium - NDSS ’08.* Internet Society, February 2008.

[26] R. Jansen, A. Johnson, and P. Syverson, “LIRA: Lightweight Incentivized Routing for Anonymity,” in *Proceedings of the Network and Distributed System Security Symposium - NDSS’13.* Internet Society, February 2013.

[27] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu, “Characterizing residential broadband networks,” in *Internet Measurement Conference: Proceedings of the 7 th ACM SIGCOMM conference on Internet measurement*, vol. 24, 2007, pp. 43–56.

[28] P. Syverson, "Why I'm not an entropist," in *Seventeenth International Workshop on Security Protocols.* Springer-Verlag, LNCS, 2009.

[29] M. Akhoondi, C. Yu, and H. V. Madhyastha, "LASTor: A Low-Latency AS-Aware Tor Client," in *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, May 2012.

[30] S. J. Murdoch and P. Zieliński, "Sampled traffic analysis by Internet-exchange-level adversaries," in *Proceedings of the Seventh Workshop on Privacy Enhancing Technologies (PET 2007)*, N. Borisov and P. Golle, Eds. Ottawa, Canada: Springer, June 2007.

[31] J. Postel. (1981) Internet protocol. RFC 791. [Online]. Available: http://www.ietf.org/rfc/rfc0791.txt

[32] Y. Rekhter, T. Li, and S. Hares. (2006, January) A border gateway protocol 4 (BGP-4). RFC 4271. [Online]. Available: http://tools.ietf.org/html/rfc4271

[33] J. Boyan, "The anonymizer," *Computer-Mediated Communication (CMC) Magazine*, 1997.

[34] A. Panchenko, L. Pimenidis, and J. Renner, "Performance analysis of anonymous communication channels provided by Tor," in *Proceedings of the The Third International Conference on Availability, Reliability and Security (ARES 2008).* Barcelona, Spain: IEEE Computer Society, March 2008, pp. 221–228.

[35] J. Reimer, "Your ISP may be selling your web clicks," 2007. [Online]. Available: http://arstechnica.com/tech-policy/2007/03/your-isp-may-be-selling-your-web-clicks/

[36] P. Dampier, "'Cable ONE spied on customers' alleges federal class action lawsuit," 2012. [Online]. Available: http://stopthecap.com/2010/02/08/cable-one-spied-on-customers-alleges-federal-class-action-lawsuit

[37] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Transactions on Networking (ToN)*, vol. 9, no. 6, pp. 733–745, 2001.

[38] V. Giotsas and S. Zhou, "Valley-free violation in internet routing-analysis based on BGP community data," in *Communications (ICC), 2012 IEEE International Conference on.* IEEE, 2012, pp. 1193–1197.

[39] L. Lamport, "Password authentication with insecure communication," *Communications of the ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[40] A. Hintz, "Fingerprinting websites using traffic analysis," in *Proceedings of Privacy Enhancing Technologies workshop (PET 2002)*, R. Dingledine and P. Syverson, Eds. Springer-Verlag, LNCS 2482, April 2002.

[41] S. J. Murdoch and G. Danezis, "Low-cost traffic analysis of Tor," in *Proceedings of the 2005 IEEE Symposium on Security and Privacy.* IEEE CS, May 2005.

[42] N. Evans, R. Dingledine, and C. Grothoff, "A practical congestion attack on Tor using long paths," in *Proceedings of the 18th USENIX Security Symposium*, August 2009.

[43] A. Serjantov and G. Danezis, "Towards an information theoretic metric for anonymity," in *Privacy Enhancing Technologies.* Springer, 2003, pp. 259–263.

[44] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: An approach to universal topology generation," in *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2001. Proceedings. Ninth International Symposium on.* IEEE, 2001, pp. 346–353.

[45] (2012) The CAIDA UCSD inferred AS relationships - 20120601. [Online]. Available: http://www.caida.org/data/active/as-relationships/index.xml

BIOGRAPHICAL STATEMENT

Jody M. Sankey was born in Lancaster, England, in 1974. He received his Bachelor of Engineering in Aeronautical Engineering from the University of Bath, England, in 1997. He is a Chartered Engineer through the Royal Aeronautical Society, and a student member of the ACM. Jody has been employed by BAE Systems and its subsiduary companies since 1995, and during that time has worked a variety of different airframe, systems, and software development roles at locations in the UK, Australia, and the US. Jody has been a part of iSec, The Information Security Lab at UT Arlington, since 2011. His current research interests include anonymity systems and in particular the impact of next-generation Internet protocols on these systems.