Routing Around Decoys

Max Schuchard[1]
schuch@cs.umn.edu

John Geddes[1]
geddes@cs.umn.edu

Christopher Thompson[2]
cthompson@cs.berkeley.edu

Nicholas Hopper[1]
hopper@cs.umn.edu

1: Department of Computer Science and Engineering, University of Minnesota, Twin Cities
2: Department of Electrical Engineering and Computer Science, University of California, Berkeley

ABSTRACT

Decoy Routing is a new approach to Internet censorship circumvention that was recently and independently proposed at FOCI'11, USENIX Security'11 and CCS'11. Decoy routing aims to hamper nation-state level Internet censorship by having routers, rather than end hosts, relay traffic to blocked destinations. We analyze the security of these schemes against a *routing capable adversary*, a censoring authority that is willing to make routing decisions in response to decoy routing systems.

We explore China, Syria, Iran, and Egypt as routing capable adversaries, and evaluate several attacks that defeat the security goals of existing decoy routing proposals. In particular, we show that a routing capable adversary can enumerate the participating routers implementing these protocols; can successfully avoid sending traffic along routes containing these routers with little or no adverse effects; can identify users of these schemes through active and passive attacks; and in some cases can probabilistically identify connections to targeted destinations.

Categories and Subject Descriptors: C.2.0 COMPUTER COMMUNICATION NETWORKS: Security and protection

General Terms: Security

Keywords: Decoy Routing, BGP, Telex, Cirripede, Censorship

1. INTRODUCTION

Decoy routing [19, 27, 18], as exemplified by Telex and Cirripede, is a new approach to building an anti-censorship tool. Instead of the traditional end-to-end based proxy solution, decoy routing instead places the proxies in the middle of paths, specifically at routers hidden throughout the Internet. Instead of explicitly connecting to these proxies, the user selects a destination whose path crosses a decoy router and signals to the router to act as a man-in-the-middle, proxying the connection to its real destination. This solves one of the main weaknesses of traditional proxies — enumeration and blocking by the censoring entity. Additionally, unlike traditional proxies, it is an explicit goal of decoy routing schemes to hide a client's usage of the system.

In this paper, we introduce the routing adversary, a new class of adversary against censorship circumvention schemes. The routing adversary is a censoring authority who is capable of controlling how packets originating from its network are routed. We describe new attacks that can be launched by a routing adversary, and allow the censoring authority to defeat each of the security goals of decoy routing schemes. In particular, we show that a censoring authority, or *warden*, that has this capability can detect the network locations of decoy routers; we demonstrate that a warden in control of how a user's packets are routed can prevent those packets from being seen by the decoy routing system; we show how an adversary that can predict the properties of paths to innocent destinations can detect the use of decoy routing through timing analysis; and we show how that same warden can launch confirmation attacks that, by exploiting the differences between a normal user and a decoy routing user, test if a host is utilizing a decoy routing system.

The majority of the attacks we present focus on wardens who are able to exert control on how a user's packets are routed. In particular, to launch our attacks the warden must be able to locate decoy routers and select from a diverse set of paths in reaction to this knowledge. We show that a restrictive nation-state — an entity decoy routing was explicitly intended to defeat — presents exactly such an adversary. Because of their history of interference with open Internet access and the diversity of their Internet connectivity, we use the examples of China, Syria, Iran, and Egypt to evaluate the effectiveness of these attacks.

Armed with both the knowledge of where decoy routers are located and a diverse collection of paths through the Internet, a warden is able to attack both the availability and deniablity of existing decoy routing schemes. In Section 4 we show how previous proposals for where to locate decoy routers allow a warden to find paths around them, preventing user traffic from being proxied. Worse, the warden can take advantage of the fact that while traditional hosts are not sensitive to the paths their packets take (a direct extension of the end-to-end principle), decoy routing users *are*. We will show a variety of ways a warden can detect this difference using active and passive means.

In addition to attacks focusing on manipulating the paths packets take, we also present a collection of attacks that exploit path properties, specifically latency. In Section 5 we consider passive timing attacks which can detect the usage of decoy routing. Even worse, we show that it is possible to fingerprint the covert website to which a user is connecting. The most troubling element of these attacks is that they are usable by weak wardens without the ability to control the path a user's packets take.

Finally, we show that there are fundamental difficulties based on the physical and economic architecture of the current Internet that limit the potential countermeasures to our attacks. We show that a deployment capable of denying these capabilities to a warden may be infeasible, requiring large fractions of the Internet to deploy decoy routers. Likewise, we discuss the limitations of traffic-shaping or other techniques in defeating timing analysis based on path properties. These limitations imply that while decoy routing may require a change in the tactics of censoring authorities, it is not an effective solution to the censorship circumvention arms race.

The remainder of the paper is organized as follows. In Section 2 we provide background information on decoy routing and Internet path selection. We then take a closer look at the implications of various countries as wardens and detail the relevant capabilities of such wardens in Section 3. In Section 4 we describe and evaluate attacks based on routing capabilities, under the deployment scenarios considered by previous work on decoy routing. Then in Section 5 we see how a warden can use fingerprinting to both detect when decoy routing is being used and, in some cases, with whom a client is actually communicating, evaluating our attack against the deployed Telex [27] station. Section 6 discusses the difficulties in countering our attacks, while Section 7 discusses related work.

2. BACKGROUND

Internet censorship circumvention tools aim to provide users with unrestricted connectivity to network resources, even when those users are located in networks controlled by the censor, henceforth referred to as the *warden*. The mostly widely deployed censorship resistance tools used today combine proxies and encrypted tunnels, examples of which include Tor [9], JAP [3], and Ultrasurf [7]. These systems provide an end-to-end approach to circumventing Internet censorship. The user makes a connection to one of these services and the service then acts as a proxy, relaying traffic between the user and the censored destination.

Unfortunately, censorship authorities have responded to these schemes with increasingly sophisticated mechanisms for identifying the hosts providing this service; for instance, there is documented evidence that both China and Iran have at times applied sophisticated Deep Packet Inspection (DPI) techniques and, in the case of China, active network probing, to every outgoing TLS connection in an effort to identify Tor Bridges [1, 4]. Once these hosts have been enumerated, these systems are easily defeated by blocking all connections to their IP addresses. To solve this issue, *decoy routing* systems were proposed. Decoy routing aims to fundamentally alter the way users communicate with the censorship resistance system.

2.1 Decoy Routing

Decoy routing systems [19, 27, 18], proposed concurrently by Karlin et al., Wustrow et al., and Houmansadr et al., use an end-to-middle approach to communication in an attempt to avoid being easily blocked. Instead of the censorship circumvention system being one of the endpoints in the communication, it is located amongst the routers used to forward packets on the Internet. Rather than making a direct connection to the proxy, the user instead selects an uncensored destination, called the *overt destination*, and initiates a TLS [8] connection to that host. The overt destination is selected such that the path from the user to the overt destination passes over a router participating in the decoy routing system, called a *decoy router*. The user signals the decoy router in a manner that the warden cannot observe, and the decoy router proceeds to act as a proxy, sending traffic not to the overt destination, but to the user's actual destination, called the *covert destination*. To the warden, it appears that the user has a functional TLS connection with the overt destination, when it actually has a connection with the covert destination.

The details of how this is done vary based on the exact system being used. Currently, two implementations of decoy routing exist: Telex [27] and Cirripede [18]. In both systems, users signal their intention to use decoy routing by selecting random fields in packets (the TLS nonce in the case of Telex and the initial sequence number in the case of Cirripede), in a predictable, but unobservable, manner. The clients then proceed to complete a TLS handshake with the overt destination, while the decoy router acts as a man-in-the-middle, eventually extracting the negotiated cryptographic key. At this point the decoy router switches to proxy mode for this connection, terminating the connection from the perspective of the overt destination with a TCP reset, and extracting the user's covert destination from packets sent by the user. For more details on how these systems function, we refer the reader to the original works.

2.2 Internet Routing

Of central importance to our work is how paths through the Internet are built. The Internet is composed of many autonomous systems (or ASes), sets of routers and IP addresses each under singular administrative control. Between ASes on the Internet, the Border Gateway Protocol [25] (BGP) is the de facto routing protocol. It allows the exchange of information between ASes about routes to blocks of IP addresses, allowing each AS to have knowledge of how to forward packets toward their destinations. BGP is a path-vector routing protocol with policies. This means that routes contain the path they traverse along with other qualities, and individual routers can define their own policies for which routes are considered "best" and used to forward packets.

These policies frequently extend beyond simply choosing the "fastest" or "shortest" routes: they allow complex and flexible decisions based on the relationships between ASes. In the Internet, there are three types of economic relationships between ASes: customer, provider, and peer. If A is a *customer* of B, then A pays B to carry traffic. Thus B is a *provider* of A. Two ASes can be *peers* of each other if they both agree to carry each others' traffic without charge. Because of these economic implications, a customer will not advertise routes to its providers other than those it or its customers originate. A provider will advertise all routes to all ASes to any of its (paying) customers. These basic policies constitute what is known as "valley-free routing" [13]—an AS never redistributes routes from one of its providers to another; if they violated this, they would end up paying for the privilege of carrying traffic for their providers. Valley-free routing is one example of routing decisions based on policy rather than path qualities. In principle, a BGP speaker can form a policy based on arbitrary criteria, a subtlety which is taken advantage of in Sections 3 and 4.

Due to the predictable routing behavior between ASes on the Internet, it is possible to infer the path along which traffic to a particular destination will be forwarded. Prior work by Qiu and Gao [24] and Mao, Qiu, Wang, and Zhang [20] detail methods for inferring the path between two endpoints on the Internet without requiring access to either.

The Internet's topology can be seen as a core of densely connected ASes, surrounded by a fringe of ASes that each have at most a handful of connections. The dense and widely geographically distributed core of the Internet means that there is a high amount of path diversity between any two ASes. This allows for operation to continue despite link failures, policy changes, and other potential issues. Each router maintains a routing table (the routing information base, or RIB), of all BGP routes it learns, and a forwarding table (the forwarding information base, or FIB), where the route chosen as "best" is stored and used to actually forward packets. But, at any given time, any of the routes in the routing table are

Country	ASNs	IP Addresses	PoC	External ASes
Australia	642	38,026,901	7	470
China	177	240,558,105	3	161
France	434	31,974,177	7	553
Iran	96	4,073,728	1	58
Syria	3	665,600	1	7
Venezuela	30	4,135,168	4	22

Table 1: The number of autonomous and IP addresses in each country, as well as the number of points of control (the smallest number of ASes that control 90% of IP addresses), and the number of external ASes directly connected to each country.

valid, and could be used in the forwarding table. Thus, an AS potentially has as many paths to each destination as it has outbound connections (peers and providers). Additionally, it can be possible to use the variety of additional route properties (such as the AS path or community attributes) to gain even more possible paths to a given destination.

3. ROUTING CAPABLE ADVERSARIES

The goal of any warden is to prevent users from accessing a set of "forbidden" websites. This could be accomplished through a variety of means, such as dropping inbound or outbound traffic, resetting TCP connections, or hijacking and middleboxing encrypted connections. A warden willing to make *routing decisions* in response to decoy routing systems can be considered a *routing capable adversary* (or simply a *routing adversary*).

Since an AS can simply change its policy configuration to alter which route it uses, and thus which path packets take, it is interesting to consider what tools this gives a warden. In addition to analyzing all traffic entering and leaving the network, a routing capable adversary is free to violate best practices and many assumptions about routing policy (e.g., those based on economic incentives, such as valley-free routing). As covered in Section 2.2, since routers store all currently valid routes, they can easily select between any of them for use in the forwarding table. Additionally, the warden could be selective about how it advertises routes to the rest of the Internet, to influence how traffic enters its network.

3.1 Wardens as Routing Adversaries

Since decoy routing was designed to defend against wardens as powerful as a nation-state, let us consider a variety of countries that have a history of monitoring Internet usage and censoring Internet access: Australia, China, France, Iran, Syria, and Venezuela. These countries also vary widely in the size and complexity of their network and their connectivity to the rest of the Internet.

Since a country can hold large amounts of political and economic control over the ASes operating within their borders, we can consider each to be not several individual ASes, but instead coalitions of ASes. While individual ASes within a warden country might have low degree in the Internet topology, collectively their connectivity to the rest of the Internet can be much higher. Using data from CAIDA [2] and the Berkman Center [6], we determined the size and connectedness of each country, as shown in Table 1. As an example, consider China with direct connections to 161 external ASes. This high degree of connectivity to the rest of the Internet means that China can select from up to 161 different paths *to any given destination on the Internet*. While other nations, for example Iran and Syria, are less well-connected, they still maintain a sufficient level of path diversity to perform routing attacks, as we will show in Section 4.

A wide variety of network engineering techniques can be used internally to allow a warden to take advantage of their path diversity. A warden could, for example, request that an ISP black-hole traffic (advertise a route that is highly preferable to existing ones) to a target destination so that they can forward it out one of their external connections. Another possible mechanism would be to have all ISPs share MPLS VPN tunnels [26], allowing them to tunnel traffic for particular destinations to the desired external connections. No matter the exact mechanism, a warden has access to a potentially large number of unique paths for the majority of destinations, allowing it to act as a powerful routing adversary.

4. ROUTING ATTACKS

Decoy routing schemes have viewed the problem of selecting where to deploy decoy routers as an issue of *availability*. It is obvious that if a user does not have even a single destination whose path crosses a decoy router, he can not utilize the system. Moreover, a user needs to be able to locate such a path quickly. Overcoming these two challenges are where authors have focused in the past. The flaw in prior work is that it approaches these issues assuming that the warden is not an active adversary. However, as discussed in Section 3, wardens are not passive entities. In this section, we show how a warden can identify which ASes are running decoy routers, even in extremely large deployments. We then show how a warden is able to launch both active attacks against the availability of decoy routers and attacks that confirm if a user is utilizing a decoy routing system, defeating both specific security goals of these systems.

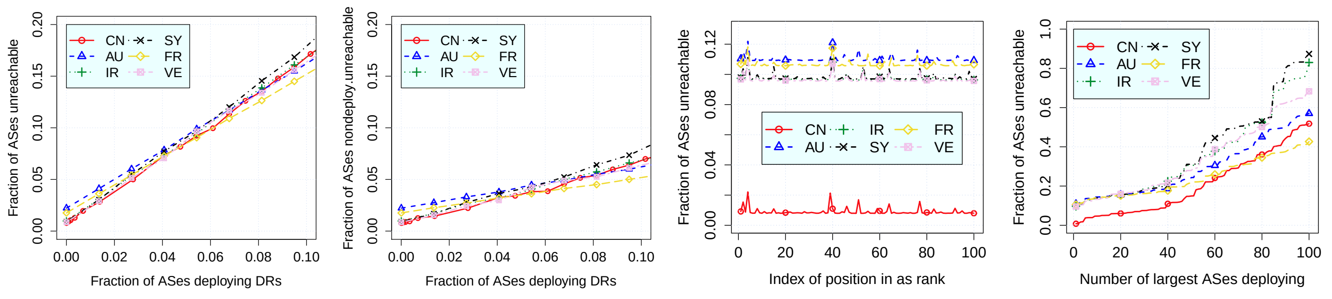
4.1 Detecting Decoy Routers

Some of our attacks require that the warden knows where decoy routers are deployed. In Telex [27], it is assumed that the directory of decoy routers is made publicly available, allowing clients to choose their overt destinations such that the usual path taken will cross a decoy router. While a public directory of decoy routers makes the use of decoy routing much simpler from the client's perspective, it also tells the warden which ASes are participating. Cirripede [18], however, instead relies on clients probing various destinations until they discover a path that crosses a decoy router. But even without such a public directory, the warden can still uncover which ASes are participating using an intersection-based discovery attack.

To determine which ASes are running decoy routers, the warden can probe a large number of paths to various destinations on the Internet using its own client. If the client does not connect to the decoy routing system using a path, the warden can add all ASes on that path to its list of "clean" ASes—the ASes that it knows are not running decoy routers. Using this list, the warden can proceed to look at all paths on which the client *was* able to connect. For each such path, the warden prunes out the known clean ASes, leaving only ASes which might be running decoy routers. If there is only a single AS left on such a path after pruning, then the warden knows that that AS must be running decoy routing (we refer to such ASes as being "tainted").

If more than one AS remains on a path after pruning, there are two possibilities. First, the warden can attempt to construct a new path for each AS remaining that otherwise only contains known clean ASes. As before, if the client fails to connect on these new paths, then that AS is also clean. If the client does connect, then that AS is tainted.

The second possibility is that the warden is unable to construct a new path. Note that the warden can always determine if the first AS on the pruned path is running decoy routing: they simply have the client attempt to connect to a destination inside that AS. From



(a) All ASes (b) Non Participating (c) Single Largest Deploy (d) Combined Largest Deploy

Figure 2: Fraction of all ASes unreachable for all wardens via at least one clean path when faced with deployments of decoy routers to random ASes. Both the fraction not reachable including those deploying decoy routers and the fraction of non-decoy router deploying ASes which are unreachable is shown.

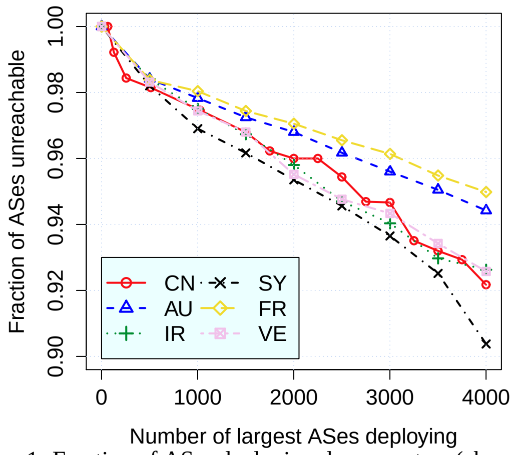


Figure 1: Fraction of ASes deploying decoy routers (chosen at random for various deployment sizes) that a warden can detect.

the perspective of the warden, this means that the later ASes on the pruned candidate path are "shadowed" by the first AS—any attempt to reach them goes through a tainted AS. To the warden, it then does not matter if they are clean or tainted.

To evaluate this and other attacks, we implemented a routing simulator based on CAIDA's [2] inferred 2011 AS level topology. We ran our experiments for Australia, China, France, Iran, Syria, and Venezuela, considering each as a warden consisting of a coalition of all their member ASes, as covered in Section 3. Paths between ASes were generated by running BGP using common routing practices, specifically valley-free routing [13]. After the routing topology converged, we then deployed decoy routers randomly to ASes for various deployment sizes, and measured what fraction of participating ASes each warden could detect using the method explained above. We found that all wardens had roughly equal success across all deployment sizes, and that they were able to detect over 90% of participating ASes for deployments as large as 4000 ASes. At such large random deployments, it is likely that most of the undetectable decoy routers were simply in the shadow of another decoy.

Since the warden must effectively mark all shadowed ASes as tainted, one goal of a decoy routing deployment would be to maximize the shadow produced by all participating ASes. However, as we explore in the following section, this is more difficult than it might appear.

4.2 Routing Around the Routers

As stated previously, the goal of decoy router deployment is to pick ASes such that all hosts in the warden's jurisdiction have at least one path that crosses a decoy router. Of all previous work, Cirripede covers how to select ASes for deployment of decoy routers in the most detail. Houmansadr et al. cover two deployment scenarios: *random* and *Tier-1*. In the random scenario, they claim that only a small fraction of randomly chosen ASes, roughly 0.4% to 1.0% of all ASes according to their results, need to be selected. Alternatively, in the Tier-1 scenario, they claim that as few as two or three Tier-1 ASes are needed, since these large transit ASes will have a vast number of paths that travel through them, including many to popular destinations, making these paths easy to locate and use.

The problem with these evaluations is that wardens, especially large ones such as China, have a large collection of diverse paths for the majority of destinations. This means that when decoy routers are deployed to a handful of large ASes, all a warden needs to do is select paths to destinations that do not utilize these ASes. Essentially, routing adversaries redefine the concept of availability for decoy routers. Instead of needing a *single* path to a destination with a decoy router on it, *all* paths to a destination need decoy routers deployed along them. The reason for this is simple. If the warden has a collection of paths to a destination (some with decoy routers and some without), then all the warden needs to do is alter its routing policy to prefer routes that do not contain decoy routers.

Of course, if all paths to a destination have decoy routers then the warden is left with several options: refuse to send information to that network, launch detection attacks against hosts sending data to those networks, or middlebox a subset of TLS connections bound for those networks. China, the most interesting example of a warden, has shown a willingness in the past to cut itself off from parts of the Internet that take actions counter to their policies, but conceivably would be unwilling to apply one of those solutions to a large portion of the Internet. Egypt, during the Arab Spring of 2011, fully disconnected itself from the rest of the Internet temporarily, and Iran has recently raised the threat of building homegrown versions of popular websites and doing the same. In essence, the decoy routing availability problem boils down to finding sufficient ASes to deploy decoy routers such that it will be too costly for the warden to handle.

Using our simulator and our reconstructed Internet topology, we explored how large of a deployment is needed to successfully disconnect a warden from a large fraction of the Internet. We deployed decoy routers using a variety of deployment strategies and measured the number of destinations to which each warden had at least one path that did not encounter a single decoy router, henceforth referred to as a *clean path*.

We start by considering Houmansadr et al.'s [18] "random ASes" scenario. Figure 2 shows the average fraction of destinations to which each warden fails to have a single clean path over 50 test deployments. This value represents the fraction of the Internet that each warden must cut itself off from in order to prevent use of the decoy routing system. We see that if deploying decoy routers to between 0.4% and 1.0% of all ASes, the wardens need only disconnect themselves from between 0.85% and 3.04% of the Internet. Essentially, these countries need only disconnect themselves from the ASes deploying decoy routers and an insignificantly sized "customer cone."[1] Figure 2 also shows exactly what fraction of non-participating ASes (i.e. those that are not deploying decoy routers) are disconnected. As can be seen there, even if 10% of the Internet deploys decoy routers, they only disconnect the wardens from a mere 7-9% of the rest of the Internet on average.

We also consider the "Tier-1 only" deployment scenario. Figure 2c shows the fraction of the Internet that is unreachable as a result of deploying individually to each of the 100 largest (by degree) ASes, excluding the ASes in each warden that fall within that set. It is clear that this strategy fails to work, as in only 2.3% of all ASes are cut off from China in the best case, while the Egypt, Iran and Syria will be cut off from 9.7% on average. Figure 2d shows the fraction of destinations each warden is cut off from as a function of deploying *simultaneously* to the top N largest ASes. As can be seen, eventually this strategy will disconnect each warden from a large fraction of the Internet, but the deployment cost is quite high. For example, in order to cut China off from at least half the Internet all of the 96 largest ISPs in the world would need to deploy decoy routers to all exit points in their network, while still needing 74-78 of them to cut off much smaller countries such as Syria. We note that such a deployment would incur high equipment costs and require incentivizing a large number of profitable companies in diverse political settings.

4.3 Detection Attacks

Attacking the availability of decoy routers is just one option open to the warden. Decoy routing systems also have the explicit goal of *unobservability*—hiding the fact that a host is using the system. However, wardens with path diversity are capable of launching attacks that unmask users of decoy routers. While the availability attack of Section 4.2 requires little in the way of real time actions by the warden (nothing more than a handful of lines in the configuration files of routers), the attacks of this section have a much more active element. In these attacks, the warden intentionally selects some paths to destinations that cross at least one decoy router, henceforth referred to as *tainted paths*. The warden then utilizes the state and topology of the network to identify a decoy routing user.

4.3.1 TCP Replay Attacks

Consider two hosts sending packets to a destination, one utilizing decoy routing, ostensibly sending traffic to the overt destination, the other a host legitimately communicating with that same destination. The most obvious difference between these two hosts is that the latter actually has a TCP connection with the destination while the former does not. The decoy routing user started a TCP connection with the overt destination, but in both existing decoy routing schemes that connection is torn down with assistance from the decoy router after TLS negotiation.

[1]AS X is in the customer cone of AS Y if AS Y is its only provider or all of its providers are in the customer cone of Y.

The challenge for the warden is to come up with a way to test if the destination thinks it actually has a TCP connection with the host. It turns out that the warden can do this quickly and cheaply if it also has a clean path to the destination, as shown in Figure 3. The warden need only replay a TCP packet sent by the host, but instead of forwarding it along the tainted path that the host is using, the warden forwards it along a clean path (Figure 3a). Because there are no decoy routers along the path to intercept the packet, it will reach the destination, and, by the end-to-end nature of the Internet, the destination is agnostic to the actual path taken by the packet. If the host was a legitimate host (Figure 3b), that is, not using decoy routing, then because there is an existing TCP stream, the destination will treat this packet as a duplicate, and, per the TCP RFC [23], send a duplicate acknowledgement. On the other hand, if the host was actually using decoy routing (Figure 3c) and the destination was simply the overt destination, no TCP connection will exist, and the destination will respond with a TCP reset packet.

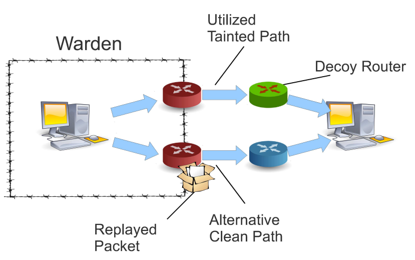
We note that if the return path of the packet crosses a decoy router, that decoy router could drop the packet.[2] However, the warden has multiple ways to force asymmetry of inbound and outbound paths.

4.3.2 Forced Asymmetry

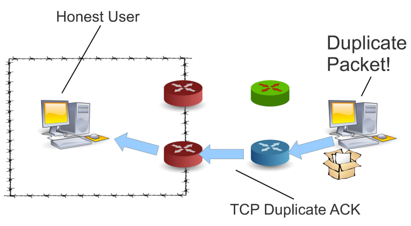
Asymmetry in the path taken by data going between two hosts on the Internet exists naturally [14]. However, a warden is able to artificially induce path asymmetry on a far larger scale. At the simplest level, all a warden needs to do is intuit which path a destination network is utilizing to send traffic to the warden, and then alter its routing policy to ensure that it picks a different path to the destination. The warden can utilize a variety of metrics including inferred AS relationships, incoming router/interface, TTLs, and packet timings in order to determine which route a destination is using.

Alternatively, a more active warden can utilize BGP's loop avoidance mechanism [25] in order to force both return path asymmetry and ensure that the return path is free of decoy routers. This attack relies on a traffic engineering technique known as hole punching. In hole punching, a router advertises both a block of IP addresses and a de-aggregation of that block, each with different path properties. Since these IP blocks are technically different, BGP will treat them as routes to different destinations, allowing for more specific policies for certain blocks of IP addresses. These more specific routes will automatically be used, as routers always forward on the *most specific matching IP block*. The warden then, for every block it wishes to advertise, hole punches a second set of routes covering the entirety of each block it would normally advertise. Since there is no currently deployed mechanism to prevent a router from falsifying route properties, an active warden can add every known decoy router deploying AS to these more specific routes. When a decoy router deploying AS receives these routes they will drop them, as it would appear like they would be creating a loop, but ASes which do not deploy decoy routers would not find themselves in the path already, and so would accept and forward these routes as normal. Since these routes are more specific, even if these non-decoy routing ASes also have the more general route that travels through decoy routing ASes, it will instead select the more specific clean route.

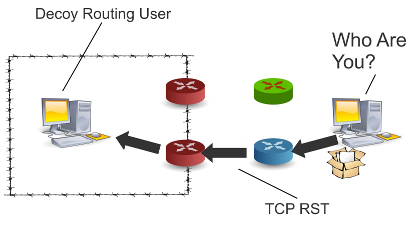
[2]In our understanding of the Cirripede design, the state of all client connections is replicated to all decoy routers, providing this functionality, while Telex does not currently explicitly provide this functionality.



(a)



(b)



(c)

Figure 3: Illustration of a simple confirmation attack launched using replayed TCP packets. In Figure 3a the warden has both a tainted path and clean path to a destination, and allows users to utilize the tainted path. The warden then replays an observed TCP packet using the clean path. If the user is honest (Figure 3b), a duplicate acknowledgement is seen. If the user is a decoy routing user (Figure 3c), a TCP reset is instead seen.

No matter how the warden achieves path asymmetry, the results are damaging to decoy routing systems. In the case of Telex, the decoy routing system simply ceases to function, as it requires path symmetry. Cirripede would function, but its use would become obvious. Packets returning from the decoy router will enter the warden at a different location in the network compared to packets returning from the overt destination. If all return paths are tainted, a decoy routing system could, in theory, shuffle packets between decoy routers to cause them to enter at the correct router and interface with the correct TTL, but this would further simplify timing attacks, which we will cover in Section 5.

4.3.3 *The "Crazy Ivan" Attack*

Another active attack for confirming if a user is utilizing a decoy routing system we call the "Crazy Ivan" Attack. A Crazy Ivan was a maneuver utilized by Soviet submarine commanders during the cold war. It consisted of a series of radical course changes in an effort to determine if an enemy submarine was hiding behind his submarine, in an area that is acoustically masked by engine noises, called a submarine's baffles. In an analogous manner, the warden can initiate a series of radical path changes and withdrawals in an attempt to unmask decoy routing users.

Again consider both a user who is utilizing decoy routing and a user who is not. Both are currently sending traffic down a tainted path. Now consider what happens if the warden flips the path utilized to this destination to a clean path. Any host not using decoy routing will not be impacted by this, and will continue with his session. Decoy routing users, however, will be impacted, as their functionality is sensitive to the path. In the worst case for the user, behavior similar to that discussed in Section 4.3.1 is seen—TCP reset packets sent from the destination. Even if the return path crosses decoy routers, which can drop the reset packets, the decoy routing user is presented with an issue. His decoy routing session no longer functions, and he can no longer pretend to communicate with the overt destination. While observed user behavior after the path to the destination is no longer tainted is not definitive proof of decoy router usage, this experiment can be repeated multiple times until the warden has a high enough confidence in its conclusions. A graphical representation of this attack can be seen in Figure 4.

Of course there is the question of what an adversary does when no clean paths are available. First, it is clear that destinations to which an alternate clean path can not be found are sub-optimal honey pots. If the warden is pushed into a scenario where such routes must be utilized another option still exists. The warden could, instead of changing the path to a destination, stop forwarding packets to the tainted destination all together. This will obviously disrupt both honest hosts and decoy routing users. The difference is that honest hosts will start new sessions with random destinations, while the decoy routing user will attempt to start new sessions down tainted paths. Again, repeated iterations of this experiment can be done to test if a user is utilizing decoy routing. Investigating the effectiveness of this last attack involves modeling user behavior and browsing habits, making it outside the scope of this work.

5. TIMING ATTACKS

One of the consequences of using decoy routing is that the path traversed to the covert destination will inevitably be different than the path that would have been used if the client was actually communicating with the overt destination. While the warden cannot explicitly notice that the paths are different, there are some unintended consequences of using different paths that might leak some information to a warden making careful observations. For instance, a warden might be able to fingerprint the flow that it would expect to see when a client communicates with the overt destination, and compare this to the flow of the actual connection made by the client. If these are significantly different, the warden can infer that the client is not actually connected to the overt destination.

One such common property of network flows that can be used in fingerprinting is network latency. Since the paths to the overt and covert destinations will diverge after the decoy router, there may be differences such as path length and bottlenecks which effect the latencies of packets traveling along these two paths. This enables a warden to be able to identify ground truth of what the range of latencies should be when communicating with an overt destination, and can compare this to the latencies they observe between a client and the overt destination. If these two distributions differ in a significant manner, the warden can infer that the client is in reality not communicating with the overt destination.

5.1 Experimental Setup

In order to validate the effectiveness of fingerprinting traffic using network latency, we took advantage of the publicly available Telex client version 0.0.2 in conjunction with the deployed Telex station. Due to the fact that connections to the overt destination must traverse the Telex station, the set of possible overt destinations was limited to `notblocked.telex.cc`, `jhalderm.com` and `notreallyblocked.telex.cc`. In our experiments, we used only `notblocked.telex.cc` for our overt destination, since all four possibilities are less than one millisecond away from the Telex station and all produce the same results.

In order to measure the latency of the client's connection through the decoy router to the covert destination, we wait until the TLS handshake is completed, during which time all communication is going through to the overt destination. We then wait until the

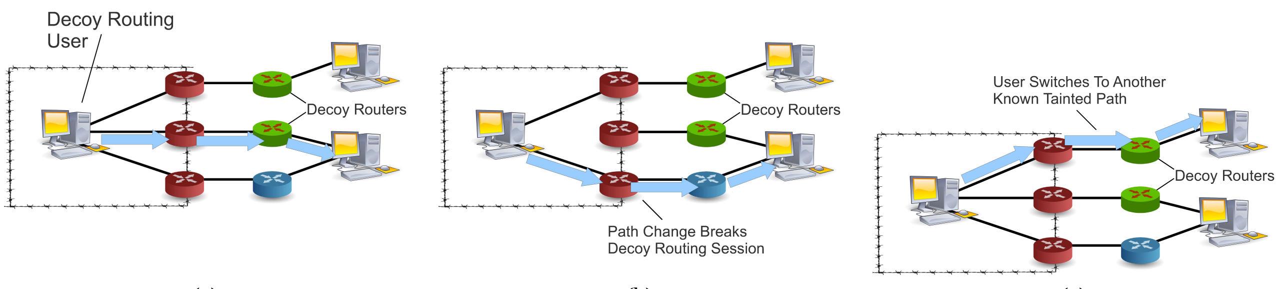

Decoy Routing User
Decoy Routers
Decoy Routers
Path Change Breaks Decoy Routing Session
User Switches To Another Known Tainted Path
Decoy Routers
(a)
(b)
(c)

Figure 4: An illustration of the Crazy Ivan attack. In Figure 4a, the warden allows users to utilize a tainted path. In Figure 4b, the warden switches to a clean path, breaking decoy routing user's session while leaving honest users unaffected. In Figure 4c, the user begins a new session, using another known tainted path, implying the users is looking for a tainted path. The warden repeats this tests several times to establish confidence in this assertion.

ChangeCipherSpec message is sent by the client, notifying us that the Telex key exchange protocol is completed and that all further traffic will be travelling to the covert destination. Once this is done, we then wait until an ApplicationData TLS packet is sent by the client and measure the time it takes to get a response ApplicationData TLS packet sent back from the server. While we are measuring the latencies of the connection from the client to the server through the decoy router, we simultaneously start up a separate direct connection to the overt destination, and similarly observe the time it takes for an ApplicationData TLS packet to be sent from the client until it receives a response from the server. This was repeated until we had 50 latency samples in our distributions.

5.2 Detecting Telex

In order to determine the feasibility of our plan of attack, we first ran some preliminary tests to see what sort of discrepancies in latency measurements could be seen when using Telex to connect to covert destinations. We first chose some arbitrary popular sites, Amazon, Gmail and Facebook, and ran our experiments to determine the latency distributions. Figures 5a-5c show the latency distributions measured to each of these covert destinations through Telex as compared to the measured latencies directly to the overt destination. As we can see, there is a significant difference in the distribution of latency measurements, implying a a warden would have no trouble at all distinguishing legitimate traffic from connections going over Telex.

While these results look promising for the warden, they are somewhat caused by the limitations in the choices we can make for the overt destination. Due to the fact that the only overt destinations available have a latency of less than one millisecond to the Telex station itself, while the selected covert destinations range anywhere from 10 to 60 milliseconds away, it is not surprising to see these large discrepancies. Because of this, we ran the same experiment using the covert destination also deployed with the overt destinations, `blocked.telex.cc`, getting rid of the large differences in latencies seen between the overt and covert destinations to the Telex station. As can be seen in Figure 5d, the distributions have much more overlap than seen previously, but there is still a significant difference in the distribution of latencies for connections going over Telex and for direct connections to the overt destination.

Given these promising results, we then moved to expand the analysis using larger sample sizes to determine exactly when a warden would be able to detect usage of the Telex system. In order to compare two latency distributions, we used the d-values returned by the Kolmogorov-Smirnov test, which quantifies the distance between two empirical distributions. For example, when comparing latency distributions for the overt destination against latency distributions for Amazon, Gmail and Facebook, we get Kolmogorov-Smirnov scores of 0.9901, 0.9536, and 1.0, respectively, and when comparing them to the latency distribution for `blocked.telex.cc` we get a score of 0.3665. To establish a baseline of what sort of scores should be expected when comparing samples from the same latency distribution, we randomly split in half the latencies that were observed to the overt destination and ran the Kolgmogorov-Smirnov test on the two samples. This was repeated 100 times to get an accurate representation of the range of scores that should be expected.

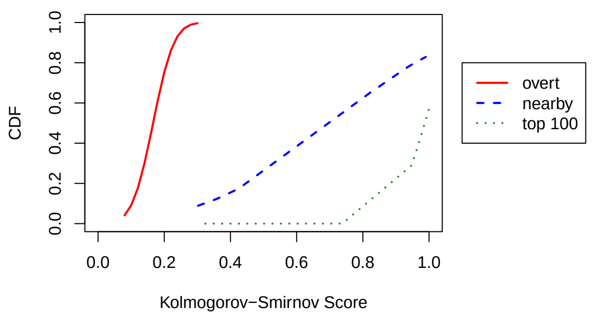
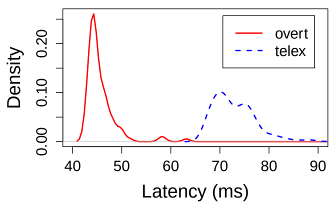
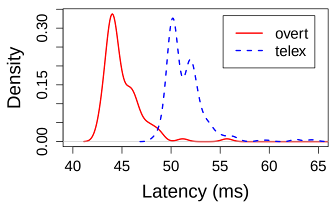


Figure 6: CDF of K-S scores when comparing an overt latency distribution to itself, to nearby servers within 10ms of the Telex station, and to the Alexa top 100 websites.

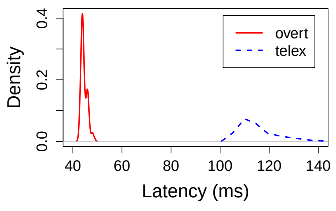
With a baseline set of scores gathered, we then wanted to see how well a warden would be able to distinguish connections going over Telex. We used two sets of covert destinations: one comprised of 10 nearby servers, all within 10 milliseconds of the Telex station; the other taken from the Alexa top 100. Figure 6 shows the CDF of Kolmogorov-Smirnov scores for the different sets of covert destinations. As can be seen, both the nearby servers and the Alexa top 100 all have significantly higher scores, ranging from 0.3 to 1.0 with median scores of 0.7 and 1.0, respectively. Compared to the set of scores seen when comparing latencies directly to the overt destination, where the maximum score is 0.26, the two sets of covert destinations are distinctly higher scoring, and would all be detectable by a warden. Furthermore, even looking at the distribution of latencies we saw earlier for `blocked.telex.cc` in Figure 5d, we see a score of 0.3665 which falls outside this range as well. This implies that a warden would be able to successfully detect a client using Telex to connect to `blocked.telex.cc`, which has a latency of approximately 0.5 milliseconds to the Telex station, which is the same as the overt destination `notblocked.telex.cc`. The large separation of latency distributions of servers so close to the Telex station suggests that the overhead of the man-in-the-middle



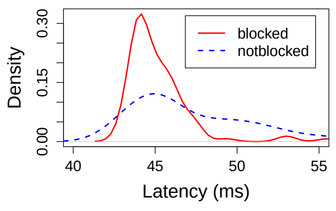
(a) Amazon



(b) Gmail

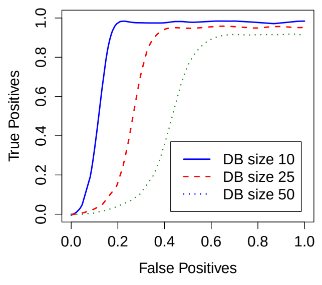


(c) Facebook

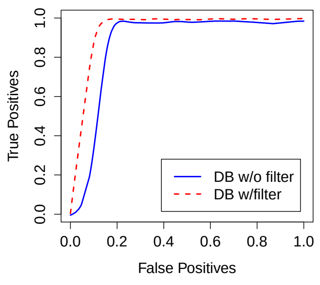


(d) blocked.telex.cc

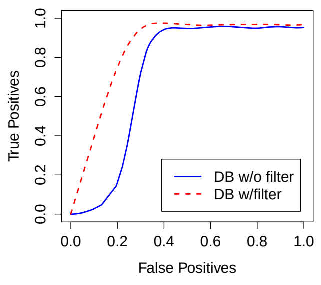
Figure 5: Comparing distribution of latencies from notblocked.telex.cc to (a) Amazon (b) Gmail (c) Facebook and (d) blocked.telex.cc



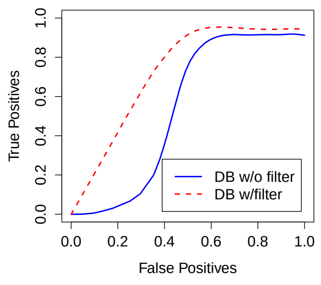
(a) ROC curves for all



(b) Database size 10



(c) Database size 25



(d) Database size 50

Figure 7: Comparing ROC curves for different database sizes in (a), and comparing ROC curves with and without filtering entries based on inter K-S scores in (b)-(d).

actions performed by the Telex station itself is causing some of the noticeable differences in latency measurements.

So far, all experiments have been run from a single machine which resides approximately 25 to 30 milliseconds away from the Telex station and servers. One possibility is that the further away the client is, the more noisy the connection will be, hiding any overhead or differences in path which are incurred by using Telex. Using PlanetLab, we selected 40 hosts, ranging from 50 to 250 milliseconds away from the Telex station and the overt destination, then ran the same previous experiments for each host, using the set of nearby servers from the previous experiments, along with `blocked.telex.cc`. These experiments were run sequentially instead of in parallel, in order to minimize any extra workload on the Telex station.

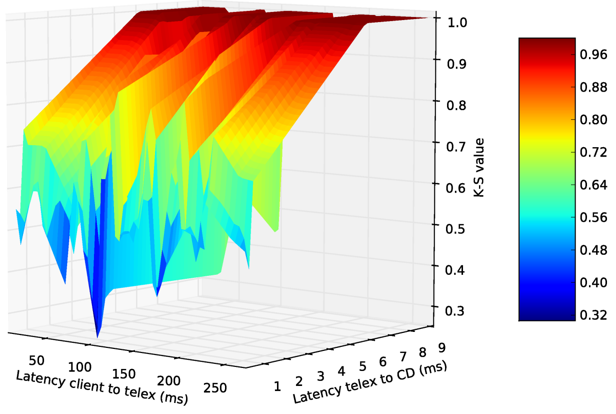


Figure 8: Surface plot of K-S score depending on client distance from Telex and distance between overt and covert destinations.

The results of these runs can be seen in Figure 8. Note that none of the Kolmogorov-Smirnov scores that were calculated were below 0.26, even when all the hosts were using `blocked.telex.cc` as the covert destination. In addition, we do not see any general trend of lower scores for hosts located further away as we had initially thought. Instead, we seem to only see background noise in the Kolmogorov-Sminov scores, with no relation to the distance of the host at all. Additionally, we looked at the latencies for each host when connecting to `blocked.telex.cc` and found that the range of scores returned was between 0.25 and 0.8. This still almost completely falls out of the range of values you would expect to see, as the CDF for the overt comparisons shown in Figure 6 show a range of 0.08 to 0.26.

5.3 Fingerprinting Covert Destinations

As we have seen, comparing distributions of latencies was an effective method for determining whether a client was either directly connecting to the overt destination or if they were using Telex and communicating with some unknown covert destination. In this section, we show how similar techniques can be used to fingerprint covert destinations, allowing a warden to identify with which sites a client is communicating.

The attack works as follows: first the warden selects a set of covert destinations to be included in the database. Then, since the warden has the ability to enumerate all decoy routers (see Section 4.1), they can build a database of latency distributions using each decoy router. When a client makes a connection, the warden uses any of the previously mentioned detection methods to determine if the client is using Telex, and then examines the path to identify the decoy router being used. After doing so, the warden compares the latency distributions for that decoy router against the observed latencies. As before, the Kolmogorov-Smirnov test is used to compare latency distributions, using a threshold on the d-value to decide when to accept or reject a sample. For our experiments, we used the latency distributions captured for the Alexa top 100 sites, and for each threshold value we would randomly select a fixed size of the samples to be in the database, using 50 of the 100 captured latencies to include in the database, while the other

50 were used to test for true positive rates. This was repeated 100 times for each threshold value to calculate the average true positive and false positive rates. Figure 7a contains the results from these experiment, showing the ROC curve for databases of size 10, 25 and 50, with AUC values of 0.868, 0.707 and 0.537 respectively.

As noted, these experiments randomly chose destinations to be included in the database. However, a warden can build a database in a more intelligent manner to improve the true positive rate while keeping the false positive rate low. By setting a lower bound threshold on the Kolmogorov-Smirnov score that any pair of entries can have, the database is built while ensuring that no two distributions are too similar. This way, the warden will be less likely to incorrectly classify an observed latency distribution. It should be noted that the larger the database is, the lower the threshold value will need to be, otherwise it will be impossible to find enough entries that are different enough from all the others. For our experiments, we used threshold values of 0.8, 0.7 and 0.35 for database sizes 10, 25 and 50. Figures 7b-7d show the results after applying a threshold on the database entries. We can see there is a significant improvement in the ROC curves, particularly for the larger database sizes.

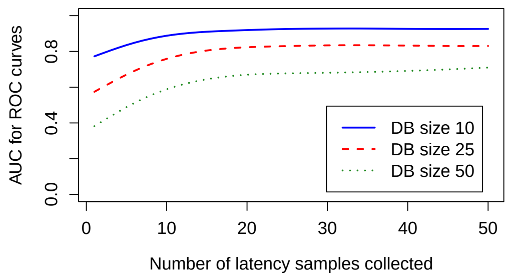


Figure 9: AUC of the ROC curve for all database sizes using different number of samples to compare to database entries.

So far, when comparing latency distributions, we have assumed that the warden has access to a somewhat large number of samples. This might not always be practical, so we tested the effect varying the number of samples had on the ROC curves. For the experiments, we restricted the size of the samples in the database to 50, while using the threshold method to ensure no two distributions in the database were too similar. We then repeated the previous experiments, creating an ROC curve while restricting the size of all samples used to compare to the database, then calculating the AUC for these ROC curves. Figure 9 shows the results from these experiments. We can see that having about 12 samples is enough to be able to consistently match distributions against the database. In fact, when restricting the size of the database to 10 distributions, even having just a few latency measurements was enough to generate ROC curves with AUC values above 0.8.

5.4 Timing Conclusions

As we have seen, a warden is able to infer a great deal of information by simply making latency measurements of connections it sees and comparing them to expected distributions. First, by comparing the distribution of latencies the warden would expect to see to the overt destination to those it observes from a client, a warden can definitively run a confirmation attack to tell if the client is using Telex or actually communicating with the overt destination. Even when a client is using Telex to communicate with a covert destination that is, for all practical purposes, running on the same machine, the overhead from the Telex station performing the man-in-the-middle actions is enough for a warden to be able to distinguish the latency distributions. Furthermore, we showed how a warden can construct databases of latency distributions of chosen covert destinations, which can be used by the warden to identify with which covert destination the client is communicating. By intelligently building the database and limiting the size, the warden is able to execute this with a remarkably high true positive rate while in many cases keeping the false positive rate under 10%.

6. COUNTERMEASURES AND THEIR LIMITATIONS

It is clear that a warden is able to launch attacks against decoy routing systems if the containment of the warden is incomplete. Sadly, achieving good containment for a warden is difficult, even for smaller, less well-connected ones, as discussed in Section 4.2. Path diversity provides far too many alternative routes to be slowed by small deployments of decoy routers. This raises an obvious question: what does a successful deployment look like? As discussed previously, a decoy routing system needs to cover *all* paths to a large enough set of destinations such that it is economically or functionally infeasible for the warden to block these destinations. But how would we best go about doing this? In a graph, a set of vertices that partition the remaining vertices into two disconnected sets is called a vertex separator. Finding an optimal vertex separator is NP-complete, with good approximations existing only for certain classes of graphs. We will instead focus on straightforward constructions of vertex separators that, while not optimal, will provide the best properties for decoy routing systems.

One immediate option is to surround the warden with a “ring” of decoy routers. The question is how many ASes would that encompass? Clearly the answer depends on how close to the warden this ring is built. If it is built close to the warden, the ring will be smaller than if it is built further out. For China, Syria, Iran, and Egypt, we consulted AS relationships from CAIDA to measure the size of this ring at various depths. We define an AS’s depth from a warden to be its minimum distance, in AS hops, from that warden. Hence, while there might be both a two hop and three hop path to a given AS, we consider it at a depth of two, not a depth of three. The sizes of the rings built by selecting all transit ASes at a given depth, are shown in Table 2, along with the fraction of the ASes external to each warden that are not reachable via at least one clean path. As can be seen, a ring at a depth of one is the smallest effective ring, with a size of 161 ASes. The following ring, at a depth of two, jumps in size by a factor of more than 23, becoming untenable in size. The ring at a depth of three is actually smaller, an artifact of defining ring membership by minimum depth, but as can be seen in the right-hand column, if containment is not achieved at a depth of two at the latest then the majority of the Internet is reachable. While the depth one ring might look promising, it is important to remember that it is comprised of ASes which have elected to directly conduct business with the warden. Providing sufficient economic incentives to take an action directly in opposition with their customer’s wishes may be difficult, considering that the warden can provide incentives to these entities to *not* deploy decoy routers.

Since a depth one ring is challenging for economic reasons and a depth three ring does not provide containment, clearly a depth two ring is the only workable option for a deployment in a ring around China. However, the depth two ring around China is 3,806 ASes large—far too large to see a successful deployment. The smallest depth two ring is around Syria, but even it contains 751 ASes. What about a fractional deployment to the depth two rings? We used our previous simulator to get some idea of the success of such a fractional deployment. The fraction of ASes that are unreachable via a clean path as a function of the fraction of the depth two ring

Country	Ring Depth	Ring Size	Size As Fraction of Remaining Transit ASes	Fraction of ASes Without Clean Paths
China	1	161	2.84%	100%
	2	3806	69.09%	91.43%
	3	1625	95.42%	2.25%
Australia	1	470	8.18%	100%
	2	3619	68.59%	78.04%
	3	1540	92.94%	3.13%
Iran	1	58	1.02%	100%
	2	1967	35.00%	98.44%
	3	3261	89.27%	16.67%
Syria	1	7	0.12%	100%
	2	751	13.26%	99.86%
	3	3969	80.79%	55.81%
France	1	553	9.50%	100%
	2	3841	75.88%	72.28%
	3	1344	94.05%	2.18%
Venezuela	1	22	0.39%	100%
	2	1993	35.29%	99.40%
	3	3176	86.92%	19.59%

Table 2: The size and containment of rings at various depths around the wardens.

receiving decoy routers can be seen in Figure 10. Again, in order to cut off Egypt, Iran and Syria from half of the Internet, more then 70% of the depth two ring needs decoy routers, while China would require more than 80%.

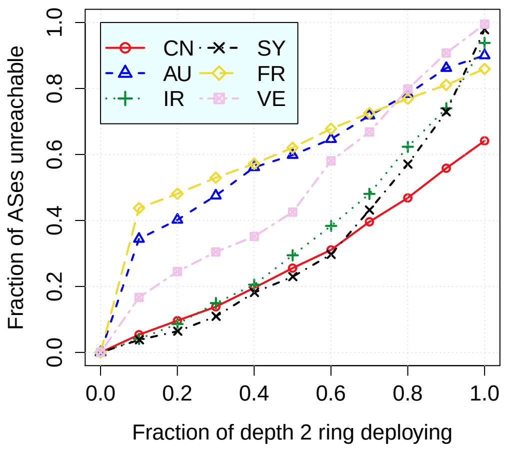


Figure 10: The fraction of all ASes unreachable from the wardens via at least one clean path for various fractional deployments to a depth two ring around the wardens.

Instead of ringing the source of traffic, an alternative strategy would be to ring popular destinations. For example, a ring could be built around the Alexa top 100. This strategy runs into a similar issue to that of the depth one ring around the warden: you must directly incentivize people to do things against their economic interests—in this case, the economic interests of the destinations. There is relatively little the destinations have to gain by being ringed with decoy routers, as they could lose customers in the warden's jurisdiction, and, consequently, revenue. This in turn would lead these content providers to select upstream ISPs that did not deploy decoy routers, making the deployment of decoy routers against the economic interests of ISPs as well. We leave a full investigation of these incentives to future work. There are two other complicating factors with a solution centered around ringing destinations. First, many popular destinations are not a single entity, but actually a broad collection of data centers, usually backed by some form of content distribution network, making containment of these destinations challenging. Second, wardens, particularly China and Iran, have shown a willingness in the past to disconnect themselves from content providers who do not agree to play by their rules and instead use homegrown solutions, meaning that the impact of such a deployment on these wardens would be limited.

An alternative would be to ring a geographic location with decoy routers. If connectivity to this region is deemed critical, we note that this can be defeated by tunneling TLS traffic. The warden rents or constructs a small data center inside the ringed location; once functional, all TLS connections bound for the region are placed in an IPsec tunnel bound for the data center, where they are unpacked and forwarded to the destination, using the correct source IP address of the client. The destination forwards packets normally to the client, but decoy routing systems are thwarted as the packets from the client are wrapped with an additional layer of encryption when they pass the decoy routers.

Timing. To prevent traffic analysis, Wustrow et al. [27] suggest having Telex perform traffic shaping, attempting to mimic network characteristics one would expect to see during a TLS connection. While this might prevent traditional traffic analysis from being done, it will do little to prevent the timing analysis we discuss in Section 5.2. The discrepancies that a warden is able to observe is due to the underlying differences in AS-level paths being taken, resulting in the network latencies being considerably higher then one would expect if the traffic was actually going to the covert destination. This is near impossible for the decoy routing to mask, because while the decoy router can increase latency by holding onto packets, there is no available method to *decrease* the latency which a warden observes. Therefore, the only way to hide this side channel is to try and make sure that the overt and covert destination have statistically similar latencies.

However, this raises some additional problems that would have to be fixed. First, since clients using the system need to broadcast to many different overt destinations, ensuring they traverse many distinct paths in order to increase their likelihood of crossing a decoy router, selecting specific overt destinations ahead of time could prove to be problematic. Furthermore, even if this was possible, by linking the choice of overt destination to the covert destination, this will reduce the anonymity of the covert destination that the user is attempting to communicate with. Finally, for many covert destinations there may not be any appropriate overt destination within the same distance from a decoy router; in this case such destinations are effectively unreachable, defeating the purpose of providing general Internet connectivity.

7. RELATED WORK

Several previous works have explored the impact of ISP-type adversaries on anonymity schemes. Feamster and Dingledine [12] analyzed the diversity of AS-level paths in anonymity networks, such as Tor and Mixmaster, and showed how path asymmetry could lead to poor location independence. Furthermore, Edman and Syverson [11] showed that even the large growth in the Tor network failed to dramatically improve AS path diversity and systems had to be aware of AS level adversaries and consciously make decisions with AS-level information in mind. Murdoch et al. [21] examined how even with high AS-level diversity in anonymity networks, many of the packets will travel through a single physical Internet exchange allowing a single entity to perform traffic analysis, negating the need for a global view. These types of studies highlight the importance of making sure anonymity systems take into account route diversity and underscores the dangers of sometimes treating the Internet as a black box.

As for the timing attacks, there has been much research conducted on how traffic analysis can be used on anonymity and other similar systems. Back et al. [5] showed how many traffic analysis techniques, and in particular latency measurements, can be used to fingerprint nodes in the network. Hopper et al. [17] expand on this and provide a formal framework on how an adversary can utilize latency measurements in the Tor network to reduce the anonymity of the client participating in the system. Several papers [22, 16, 10, 15] showed that by using more sophisticated fingerprinting methods, adversaries are able to perform website fingerprinting in the Tor network to identify the end server that a user is communicating with. These attacks are based on the size of downloaded files and could potentially be combined with our timing attacks to yield even more accurate identification of covert destinations.

8. CONCLUSION

In this paper, we have introduced a novel adversary model for decoy routing, the routing capable adversary, exploring the actual routing capabilities that a warden has and the implications that such an adversary has with respect to decoy routing. Specifically, we showed how wardens can easily enumerate all deployed decoy routers and use this information to successfully route around all such routers. We explored, in depth, the intricacies of deployment strategies and analyzed the effects they have with respect to the enumeration attacks. In addition, we showed how a warden can run multiple confirmation attacks to detect when a client is participating in the system and not actually communicating with their overt destination. Lastly, we showed that a warden can use fingerprinting techniques to expose the identity of the secret destination that a client is communicating with through the decoy routing system.

These results show that small deployments can be trivially defeated, requiring larger deployments for decoy routing to be successful. However, several of our confirmation attacks still work, even against very large deployments. This suggests that new ideas will be needed before decoy routing can be deployed in a secure and cost effective manner.

Acknowledgments This work was supported by NSF grant 0917154.

9. REFERENCES

[1] Knock Knock Knockin' on Bridges' Doors. `https://blog.torproject.org/blog/knock-knock-knockin-bridges-doors`.

[2] CAIDA AS relationship dataset. `http://www.caida.org/data/active/as-relationships/index.xml`.

[3] JAP: The JAP anonymity & privacy homepage. `http://www.anon-online.de`.

[4] New blocking activity from iran, Sep, 14, 2011. `https://blog.torproject.org/blog/iran-blocks-tor-tor-releases-same-day-fix`.

[5] A. Back, U. Möller, and A. Stiglic. Traffic analysis attacks and trade-offs in anonymity providing systems. In *Proceedings of the 4th International Workshop on Information Hiding*, IHW '01, pages 245–257. Springer-Verlag, 2001.

[6] Berkman Center for Internet & Society. Mapping local internet control. `http://cyber.law.harvard.edu/netmaps/geo_map_home.php`.

[7] U. I. Corporation. Ultrasurf - proxy-based internet privacy and security tools. `http://ultrasurf.us`.

[8] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.2. RFC 5246 (Proposed Standard), Aug. 2008. Updated by RFCs 5746, 5878, 6176.

[9] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th conference on USENIX Security Symposium*, pages 21–21. USENIX Association, 2004.

[10] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton. Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, May 2012.

[11] M. Edman and P. Syverson. As-awareness in tor path selection. In *Proceedings of the 16th ACM conference on Computer and communications security*, CCS '09. ACM, 2009.

[12] N. Feamster and R. Dingledine. Location diversity in anonymity networks. In *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, WPES '04, 2004.

[13] L. Gao and J. Rexford. Stable internet routing without global coordination. *IEEE/ACM Transactions on Networking (TON)*, 9(6):681–692, 2001.

[14] Y. He, M. Faloutsos, and S. Krishnamurthy. Quantifying routing asymmetry in the internet at the as level. In *Global Telecommunications Conference, 2004*, volume 3 of *GLOBECOM '04*, pages 1474–1479. IEEE, 2004.

[15] D. Herrmann, R. Wendolsky, and H. Federrath. Website fingerprinting: attacking popular privacy enhancing technologies with the multinomial naive-bayes classifier. In *Proceedings of the 2009 ACM workshop on Cloud computing security (CCSW '09)*, pages 31–42, New York, NY, USA, 2009. ACM.

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

[17] N. Hopper, E. Y. Vasserman, and E. Chan-tin. How much anonymity does network latency leak. In *Proceedings of the 14th ACM conference on Computer and communications security*, CCS '07, 2007.

[18] A. Houmansadr, G. T. Nguyen, M. Caesar, and N. Borisov. Cirripede: circumvention infrastructure using router redirection with plausible deniability. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS)*, 2011.

[19] J. Karlin, D. Ellard, A. W. Jackson, C. E. Jones, G. Lauer, D. P. Mankins, and W. T. Strayer. Decoy routing: Toward unblockable internet communication. In *Proceedings of the USENIX Workshop on Free and Open Communications on the Internet (FOCI)*, 2011.

[20] Z. Mao, L. Qiu, J. Wang, and Y. Zhang. On as-level path inference. In *ACM SIGMETRICS Performance Evaluation Review*, volume 33, pages 339–349. ACM, 2005.

[21] S. J. Murdoch and P. Zieliński. Sampled traffic analysis by internet-exchange-level adversaries. In *Proceedings of the 7th international conference on Privacy enhancing technologies*, PET'07, 2007.

[22] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel. Website fingerprinting in onion routing based anonymization networks. In *Proceedings of the 10th annual ACM workshop on Privacy in the electronic society*, WPES '11. ACM, 2011.

[23] J. Postel. Transmission Control Protocol. RFC 793 (Standard), Sept. 1981. Updated by RFCs 1122, 3168, 6093, 6528.

[24] J. Qiu and L. Gao. As path inference by exploiting known as paths. In *IEEE GLOBECOM*, 2006.

[25] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), Jan. 2006. Updated by RFC 6286.

[26] E. Rosen and Y. Rekhter. BGP/MPLS IP Virtual Private Networks (VPNs). RFC 4364 (Proposed Standard), Feb. 2006. Updated by RFCs 4577, 4684, 5462.

[27] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman. Telex: anticensorship in the network infrastructure. In *Proceedings of the 20th USENIX Conference on Security (SEC)*, 2011.