Stealth Probing: Efficient Data-Plane Security for IP Routing

Ioannis Avramopoulos and Jennifer Rexford
Princeton University
{iavramop,jrex}@princeton.edu

Abstract

IP routing is notoriously vulnerable to accidental misconfiguration and malicious attack. Although secure routing protocols are an important defense, the data plane must be part of any complete solution. Existing proposals for secure (link-level) forwarding are heavy-weight, requiring cryptographic operations at each hop in a path. Instead, we propose a light-weight data-plane mechanism (called *stealth probing*) that monitors the availability of paths in a secure fashion, while enabling the management plane to home in on the location of adversaries by combining the results of probes from different vantage points (called *Byzantine tomography*). We illustrate how stealth probing and Byzantine tomography can be applied in today's routing architecture, without requiring support from end hosts or internal routers.

1 Introduction

Most research and standards activity in secure IP routing has focused on the routing protocols, rather than the forwarding of data packets. In this paper, we propose an *operationally viable* approach to providing data-plane security. As an example of the threats we address, consider an attacker that breaks into one or more routers, or a disgruntled network operator with easy access to the routers. The adversary can easily create an unnoticed disruption by installing access control lists (ACLs) that selectively discard data traffic, while leaving the routing protocol intact and allowing probe traffic through. Since the routing protocol does not verify the operation of the data plane, and because the probes are delivered successfully, this attack is extremely difficult to diagnose.

Threat Model: We consider a network where a subset of the routers and links (unknown to the defending entity) are controlled by an adversary. Using these routers and links, the adversary can eavesdrop on host-to-host communications, tamper with the packet contents, impersonate host services, misdirect network traffic, or render the packet-delivery service unavailable by means of routing-protocol and data-plane attacks. A remote adversary may also deplete data-plane resources through denial-of-service attacks; because other techniques, such as fair queuing and packet filtering, can protect from these attacks, we do not consider them further.

Our primary goal is to secure routing through *data-plane countermeasures* that detect routing-protocol and data-plane attacks that disrupt packet delivery, assuming arbitrary (or *Byzantine*) behavior by the adversary. Ensuring the availability of the network despite the presence of adversaries prevents financial losses and other detrimental societal impacts that these adversaries could otherwise inflict. We do not prevent traffic misdirection attacks (and, thus, we avoid any associated overheads such as cryptographically enforcing a route), though we use network encryption of the data traffic to counter their consequences. Encryption, furthermore, safeguards against eavesdropping, the tampering of packet contents, and the impersonation of host services.

Layering of Countermeasures: In an operationally viable secured routing system, we argue that we should consider all three dimensions of IP routing, i.e., the *data*, *control*, and *management planes*: The data plane supports packet-forwarding functionality, such as destination-based forwarding, filtering, and tunneling. The control plane implements the routing protocols that discover the topology and select routes. The management plane monitors the network and configures the routers. The management and control planes have been the focus of most countermeasures to Byzantine failures.

Management Plane: As a management-plane countermeasure, operators can apply Best Common Practices (BCPs) for securing their infrastructure and filtering suspicious route announcements. These BCPs reduce the likelihood of attacks but cannot prevent them entirely; in addition, an end-to-end path often traverses multiple networks, including some that do not apply BCPs.

Control Plane: Secure routing protocols [8, 10, 14, 18] ensure that valid routing advertisements correctly identify the links between non-faulty routers (or Autonomous Systems). However, these protocols do not prevent false announcements that faulty routers are connected to other faulty routers, as in collusion (or wormhole) attacks [8, 18]. Wormholes along with malicious ACLs can create invisible black holes for the data traffic. Because routing protocols do not verify forwarding behavior, even if perfectly secure routing protocols were deployed, availability could still be compromised. This risk can be mitigated by incorporating secure forwarding functionality in the routing system.

Data Plane: Secure forwarding protocols such as [2, 4] and the Π_2 protocol in [13] provide availability monitoring and secure fault localization at the *link level*. The fine granularity leads to high overhead and complexity (e.g., path-specific authentication and, in certain cases [2, 4], the distribution of pairwise router keys) inappropriate for a generic forwarding paradigm. Although useful for failure recovery, fault localization should not overburden the data plane. We advocate that availability monitoring and fault localization should be cleanly separated, into the data plane (*stealth probing*) and management plane (*Byzantine tomography*), respectively.

Stealth Probing: The stealth-probing protocol we propose in this paper is a data-plane availability monitor. It determines whether a router-to-router path is operational, even if an adversary controls intermediate routers and tries to evade detection. Stealth probing creates an encrypted tunnel between two end-routers and diverts both the data and probe traffic into the tunnel. Since the data and probe packets are indistinguishable, the adversary cannot drop data packets without dropping the probes as well, making it difficult to evade detection. Rather than requiring ubiquitous deployment, stealth probing could be deployed "as needed" to protect critical traffic between selected edge networks.

Stealth probing offers several key practical advantages. First, stealth probing is *incrementally deployable*. Because of its end-router-to-end-router design, stealth probing does not require support from legacy routers in the core of an ISP network or intermediate ASes in an interdomain path. Networks that adopt stealth probing will see immediate benefits even in limited deployment scenarios. Second, stealth probing is *backward compatible* with the existing infrastructure, since the tunnels do not require any support from the internal routers. Finally, stealth probing is *incentive compatible*. Service providers can use the encrypted tunnels to provide other value-added services, such as secure Virtual Private Networks (VPNs), to customers. Encrypted tunnels also protect users from a broader range of attacks such as eavesdropping, tampering, traffic analysis, and misdirection.

2 Stealth Probing

Stealth probing is a secure data-plane monitoring tool that relies on the efficient symmetric cryptographic protection of the IPsec protocol suite, applied in an end-router-to-end-router fashion. In this section, we first discuss the limitations of other approaches to secure data-plane monitoring, followed by an overview of stealth probing. Then, we describe how stealth probing works in greater detail.

2.1 Limitations of Strawman Designs

Stealth probing addresses the problem of securely deciding whether a node-to-node path correctly delivers data packets from one end of the path to the other. An adversary that is present at one or more intermediate nodes of the path must not be able to coerce a false decision. Furthermore, the overhead of the decision process must be practical for deployment in operational networks.

Consider two routers u and v. Lets assume for simplicity that u is a source of data traffic for which v is a sink. We want to verify that this traffic is flowing properly in the forward $u \rightarrow v$ direction.

Probing One approach to meet our objective is for router u to send to v one or more ICMP echo requests and infer the fate of data traffic based on the receipt of ICMP echo replies. This method is non-intrusive since it reaches a decision with a small number of probes. However, if an adversary is present in the path between u and v, he can selectively drop data packets and avoid detection by selectively forwarding echo requests and replies.

Cumulative network-layer ACKs In a second approach, v explicitly acknowledges receipt of a bundle of data packets from u by a cumulative ACK that contains a count of the received packets. However, if an adversary is present in the path between u and v, he can drop data packets and avoid detection by forging destination ACKs. So, let's further assume that u and v share a secret key. Using this key, we can prevent this attack by requiring u to authenticate data packets by means of a message authentication code (MAC) and v to authenticate ACKs in the same way. However, packet counts are insufficient to determine the timeliness of data delivery and, therefore, this scheme is vulnerable to an adversary that *delays* packets. Furthermore, packet counts are insufficient to detect *selective* attacks that target individual IP addresses (or prefixes).

Transport-layer ACKs A third approach is to use a secure (host-to-host) transport layer protocol such as TLS (Transport Layer Security) [5]. However, because this scheme cannot differentiate between host and router failures, it would suffer from "false alarms" due to host failures that would complicate fault localization by the

management plane.

Traceroute A fourth approach is that adopted by traceroute that uses ICMP "time exceeded" and "port unreachable" messages to either determine the full path from a source to a destination or identify the last router before a black hole. Traceroute has fine *link-level* detection granularity but cannot prevent the preferential treatment of its packets by an adversary who can in this way avoid detection. In addition, many ISPs disable their routers from sending ICMP response messages.

2.2 Minimal Secure Data Plane Monitor

Stealth probing has a "minimalist" design: It enables recovery from routing attacks and misconfigurations by offering secure *path-level* failure detection capability, keeping the data-plane support to a minimum. The idea in stealth probing is to use probes to reach a secure decision on the fate of data traffic by establishing an encrypted and authenticated *tunnel* between two routers in the traffic's path and diverting both the data traffic and the probes into this tunnel. Encryption conceals probing traffic so that it is indistinguishable from data traffic, and authentication makes the tampering of data traffic detectable. Probing can be either *active* or *passive*:

- Active probing uses ICMP echo requests and replies. The size of echo requests is concealed using padding to decrease the number of distinct data-packet sizes. Echo request sizes are then chosen to match data-packet sizes, and inter-probing intervals are randomly jittered.
- In passive probing, the tunnel entry and exit points agree on an efficient (non-cryptographic) hash function to be applied on the immutable fields of each packet—before encryption at the entry point and after decryption at the exit point. If the image of the hash is less than an agreed-upon value, the corresponding data packet serves as an implicit probe that the tunnel exit point must acknowledge. This method is akin to *trajectory sampling* [6]; a Bloom filter may be used to compress the ACKs, similar to how hashes are compressed in [17].

Stealth probing has the following primary benefits:

- Because stealth probing is an *end-router-to-end-router failure detection mechanism*, intermediate routers of a monitored path do not need to explicitly support the stealth probe. Therefore, stealth probing can be deployed across legacy routers and over interdomain paths.
- Stealth probing is *non-intrusive*. The processing requirements at tunnel endpoints (outlined in Section 2.3) are simple and the probing overhead is minimal. Intermediate routers do not process tunneled packets as they are tunnel agnostic.
- By measuring the round-trip-times of probing traffic, attacks that delay packets are detectable.
- By hiding the source and destination IP addresses of the data traffic, encryption prevents attacks that target individual IP addresses.
- By making the TCP mechanism *opaque*, encryption mitigates attacks that exploit the TCP mechanism.
- The use of tunnels permits *selectivity* in the traffic that is protected. The management plane can configure packet classifiers that identify the critical traffic and direct only the matching packets into the encrypted tunnels.

Stealth probing has the following secondary benefits:

- Encryption at the edge routers of a network infrastructure (even if selectively applied) (a) prevents the eavesdropping of unencrypted host-to-host communications, (b) prevents traffic-analysis attacks that host-to-host encryption does not prevent, for example, by hiding the source and destination addresses of data traffic, (c) precludes the adversary from impersonating the services of the receiving host, (d) renders misdirection attacks that divert traffic to adversarially controlled locations for eavesdropping and traffic analysis ineffective, and (e) enables ISPs to offer value-added services like VPNs.
- Stealth probing enforces fate sharing between data traffic and probes, which is broadly useful for troubleshooting network problems. For example, simple ICMP echo requests and replies may be treated differently from data packets either because of MTU size limits or packet filters that discard traffic based on the protocol or port numbers. Stealth probing avoids this problem by tunneling all traffic and matching the packet sizes of data and probe traffic (e.g., due to the padding step in active probing or the random packet sampling in passive probing).
- Tunnels are broadly useful for controlling the flow of traffic in an AS (e.g., for traffic engineering).

2.3 Mechanics

Stealth probing requires the endpoints of a path to share a secret and use this secret to create an *IPsec tunnel*. This section charts the workings of the IPsec protocol suite and the process that directs packets into tunnels.

IPsec protocol suite: IPsec provides end-to-end cryptographic protection at the IP layer. The communicating parties—the tunnel end-points—use the Internet Key

Exchange (IKE) protocol [7] to negotiate the establishment of a Security Association (SA). IKE relies on preshared secret keys or the public keys of an associated Public Key Infrastructure (PKI). In intradomain routing, key exchange can be assumed by a domain's authority; in interdomain routing, key exchange should not depend on a single central authority. Due to its end-to-end design, stealth probing does not depend on such authority.

Following the SA establishment, IP packets are protected using an Encapsulating Security Payload (ESP) module [9]. Using tunnel-mode ESP, the tunnel entry point adds an outer IP header to each packet, followed by the ESP header and trailer. ESP provides encryption using a standard encryption algorithm and ensures authenticity and integrity using a standard MAC. The tunnel exit point removes the outer IP header and restores the inner IP packet by inverting the encryption. Stealth probing, therefore, relies only on efficient symmetric cryptographic primitives. Thus, packet processing can proceed at the line speeds of core routers. Commercial routers increasingly offer such encryption capabilities.

Directing packets into tunnels: The management plane configures packet classifiers to specify which traffic should enter the tunnel, based on the five-tuples of source and destination address prefixes, port numbers, and protocol numbers. Tunnels are deployed across the network to match this specification (see Section 3). For protected packets, a longest-prefix-match table lookup will determine the tunnel exit point, based on a packet's destination address. A simple table lookup will then retrieve the associated encryption key needed to encapsulate the packet.

3 Deployment Scenarios

In this section, we present two deployment scenarios for stealth probing. First, we describe how an ISP can deploy stealth probing to secure its own infrastructure. Then, we discuss how a pair of edge networks can deploy stealth probing to secure the path through untrusted ASes in the Internet.

3.1 Intradomain Routing

Identifying tunnel endpoints: An ISP network typically has a *periphery* (i.e., edge routers that aggregate customer, transit, and peering traffic) and a *core* that interconnects the edge routers. The edge routers are an apt location to deploy stealth probing to leverage the benefits of an end-to-end design. First, core routers can be tunnel-agnostic and need only support simple destination-based forwarding and, second, processing requirements are distributed over a large number of edge routers. The management plane can configure five-tuples to identify the protected traffic, as discussed in Section 2.3.

The tunnel exit point corresponds to the next-hop attribute of the chosen BGP route for the destination prefix. A longest prefix match on a packet's destination address will determine the tunnel exit point (i.e., the egress router), and a simple table lookup returns the appropriate encryption key. In terms of scale, a large ISP network might have a few hundred edge routers, resulting in a few hundred keys and a few hundred tunnels per ingress point (i.e., one per egress router). Compared to the standard forwarding-table lookup that must be performed for each IP packet, the overhead of retrieving the keys is low; in fact, the forwarding table could store a pointer to the appropriate key for each prefix.

Byzantine tomography: In a network under attack, stealth probes detect the dysfunctional paths. Armed with this knowledge, the management plane can identify the compromised routers and recover from the attack. In the simplest case, the management plane can reconfigure or reboot the compromised routers, or reinstall the routers' operating system. Fine-grained detection of the compromised routers is useful to avoid the unnecessary downtime caused by false alarms. Byzantine tomography estimates the compromised routers by combining stealth probing output from multiple vantage points.

Byzantine tomography generalizes the notion of network tomography, which identifies the loss rates of network links using end-to-end probing traffic, by assuming that (the unknown) malicious routers may lie about their collected measurements. Byzantine tomography minimizes, over all possible faulty compositions, the number of faulty routers that explain the faulty paths observed in stealth probing. Algorithmically, this is an instance of the Minimum Hitting Set (MHS) problem: If S is the set of routers in the network and C is the collection of paths (subsets of S) that are faulty, a *hitting set* for C is a subset S' of S such that S' contains at least one element from each path in C. MHS can be solved using one of the algorithms presented in [3, 11].

The adversary's goal is to disorient the management plane into false detections. For example, the adversary can instruct the compromised routers to spuriously report certain paths as dysfunctional. If we require the routers to cryptographically sign their stealth-probing reports, a compromised router could not forge a bogus report for another router. As such, these reports could only identify paths that include the faulty router, making these reports accurate because the path does indeed contain a compromised router!

An adversary could also try to thwart the management system by selectively discarding packets traversing a small number of paths, making it difficult for Byzantine tomography to have fault reports from enough vantage

points to identify the compromised routers. However, in doing so, the adversary also confines the scope of attacks. The more selective the adversary is in dropping packets (to evade detection), the less extensive the damage of the attack. In addition, even if Byzantine tomography cannot uniquely identify the faulty routers, the network operators could easily take corrective action based on a *set* of suspected routers. For example, the operators could reconfigure the remaining routers to select paths that circumvent the suspected routers, or reboot each of the suspected routers.

3.2 Interdomain Routing

Securing interdomain routing is arguably harder than securing intradomain routing for two reasons. First, without a trusted central authority, key distribution is more challenging. Second, the compromised routers might reside in a remote AS outside the control of the communicating edge networks, making fault localization and fault recovery more challenging. An interdomain deployment can address these challenges through a small-scale key distribution (between selected edge networks) and coarse-grain rerouting (through techniques commonly used for intelligent route control).

Incremental deployability: ASes willing to deploy stealth probing over interdomain paths can engage in bilateral or small-scale multilateral agreements, and exchange pairwise keys either manually or by small-scale PKIs. ISPs have an incentive to join small groups, both to provide value-added services (such as multi-site VPNs) and to securely detect connectivity problems (to ensure higher availability for their services). Because stealth probing can be deployed across tunnel-agnostic legacy routers, early adopters will see an immediate benefit without requiring the participation of intermediate ASes. In fact, stealth probing enables the participating ASes to provide secure service, despite the presence of untrusted ASes in the rest of the Internet. The economic return to the early adopters can provide an incentive for other ASes to join these groups. As more ASes join these groups, scalable key distribution could be addressed through a larger PKI or a distributed trust model.

Circumventing the compromised routers: Although securely detecting routing failures is an important capability in its own right, the ability to bypass the affected routers is important as well. However, in interdomain routing, the communicating edge networks might not be able to identify the specific routers (or ASes) that have been compromised. Instead, the tunnel end points can adapt by directing the tunneled traffic on a different path, in the hope of circumventing the compromised routers, following techniques used in intelligent route control [1]. For example, consider two stub ASes, AS_1 and AS_2, and assume that AS_1 is m_1-multihomed and AS_2 is m_2-multihomed. (For simplicity, also assume that each of AS_1 and AS_2 has a single border router.) AS_1 can choose among $m_1 \times m_2$ different BGP paths to forward traffic from AS_1 to AS_2. Choosing any of the m_1 outgoing links is straightforward for AS_1. Furthermore, any of the m_2 incoming links to AS_2 can be chosen as follows: AS_2 advertises a different primary prefix to each of its m_2 providers, and destination addresses from each of these prefixes are used to terminate m_2 tunnels between the border routers of AS_1 and AS_2. AS_1 can, thus, direct traffic via any of the m_2 incoming links to AS_2 by choosing the remote tunnel end-point address accordingly. AS_2 selects the reverse path to AS_1 in the same manner. In this setting, stealth probing can detect which of the $m_1 \times m_2$ paths contain compromised routers, and the edge networks can switch to a working path.

4 Related Work

Encryption to make data and control traffic indistinguishable was first suggested by Perlman [16], who proposed hop-by-hop encryption between neighboring routers to hide beaconing traffic and prevent "man-in-the-middle" attacks on the topology-discovery process. The novelty of stealth probing is in applying this general idea to the paths between end-routers to identify data-plane problems in a secure fashion. Perlman also proposed recovery from routing attacks using multipath routing and disjoint paths. Stealth probing is well-suited for monitoring the quality of active paths in order to dynamically recompute the active path set.

The Fatih [13] secure data-plane monitor can adjust detection granularity from link-level to path-level for lower overhead. However, [13] does not propose a fault-localization mechanism to compensate for the reduced detection level, and the scheme also requires synchronized clocks. In our proposal, fault localization is attained using Byzantine tomography and we do not rely on clock synchronization.

Secure traceroute [15] is a link-level detection scheme that could conceivably be applied at the path level. Secure traceroute is based on secret identifiers embedded in packets that single out those packets as probes, which elicit responses for detecting reachability. However, this scheme may fail to detect attacks that target low-rate components of the aggregate traffic in a path or attacks that exploit the TCP mechanism. By encrypting traffic, stealth probing prevents those attacks. Secure traceroute could conceivably be extended into a hybrid scheme where the sender initiates link-level detection only after path-level probing suggests a problem. However, an adversary could easily thwart the on-demand link-level detection by limiting the duration of attacks; in addi-

tion, such a hybrid scheme would still require pairwise keys between the routers. In contrast, stealth probing, combined with Byzantine tomography, can pinpoint even short-lived attacks and does not require per-hop keys.

Other recent proposals, such as Listen [18] and Feedback-Based Routing [19], detect data-plane attacks by monitoring traffic at the TCP level. However, these techniques would falsely detect an unavailable path to a prefix as workable, if an adversary impersonates hosts in the monitored prefix.

5 Conclusion

In this paper, we presented stealth probing and Byzantine tomography as effective ways to protect against network-availability attacks without overburdening the data plane. We also showed how these techniques can be applied in the Internet's existing routing system, without changing the end hosts or the internal routers. In the future, we will explore "clean-slate" secure routing system designs. In particular, we will study whether more flexible path-selection schemes, such as source routing, are necessary, or whether coarse-grained path selection is sufficient for secure routing. We will also explore the many security benefits of encrypting the data traffic between edge networks, and study how to balance the trade-offs between host-based and network-based encryption for providing secure Internet services.

Acknowledgments

The authors would like to thank Constantinos Dovrolis, Nick Feamster, Karthik Lakshminarayanan, Barath Raghavan, Alex Snoeren, and the anonymous reviewers for their invaluable feedback. Ioannis Avramopoulos has been supported by a grant from the New Jersey Center for Wireless and Internet Security and a wireless testbed project (ORBIT) grant from the National Science Foundation. Jennifer Rexford was supported by Homeland Security Advanced Research Project Agency grant 1756303.

References

[1] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman. A measurement-based analysis of multihoming. In *Proc. ACM SIGCOMM*, Aug. 2003.

[2] I. Avramopoulos, H. Kobayashi, R. Wang, and A. Krishnamurthy. Highly secure and efficient routing. In *Proc. IEEE Infocom*, Mar. 2004.

[3] I. Avramopoulos, A. Krishnamurthy, H. Kobayashi, and R. Wang. Nicephorus: Striking a balance between the recovery capability and the overhead of Byzantine detection. Technical Report TR-710-04, Dept. of Computer Science, Princeton University, Oct. 2004.

[4] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An on-demand secure routing protocol resilient to byzantine failures. In *Proc. ACM Workshop on Wireless Security*, Sep. 2002.

[5] T. Dierks and C. Allen. The TLS protocol version 1.0. RFC 2246, IETF, Jan. 1999.

[6] N. Duffield and M. Grossglauser. Trajectory sampling for direct traffic observation. *IEEE/ACM Trans. Networking*, 9(3):280–292, Jun. 2001.

[7] D. Harkins and D. Carrel. The Internet Key Exchange (IKE). RFC 2409, IETF, Nov. 1998.

[8] Y.-C. Hu, A. Perrig, and M. Sirbu. SPV: A secure path vector routing scheme for securing BGP. In *Proc. ACM SIGCOMM*, Sep. 2004.

[9] S. Kent and R. Atkinson. IP Encapsulating Security Payload (ESP). RFC 2406, IETF, Nov. 1998.

[10] S. Kent, C. Lynn, and K. Seo. Secure Border Gateway Protocol (Secure-BGP). *IEEE Journal on Selected Areas in Communications*, 18(4):582–592, Apr. 2000.

[11] R. Kompella, J. Yates, A. Greenberg, and A. Snoeren. IP fault localization via risk modeling. In *Proc. Symposium on Networked System Design and Implementation*, May 2005.

[12] G. Mathur, V. Padmanabhan, and D. Simon. Securing routing in open networks using secure traceroute. Technical Report MSR-TR-2004-66, Microsoft Research, Jul. 2004.

[13] A. Mizrak, Y.-C. Cheng, K. Marzullo, and S. Savage. Fatih: Detecting and isolating malicious routers. In *Proc. International Conference on Dependable Systems and Networks*, Jun. 2005.

[14] S. Murphy, M. Badger, and B. Wellington. OSPF with digital signatures. RFC 2154, IETF, Jun. 1997.

[15] V. Padmanabhan and D. Simon. Secure traceroute to detect faulty or malicious routing. In *Proc. ACM SIGCOMM HotNets Workshop*, Oct. 2002.

[16] R. Perlman. *Network Layer Protocols with Byzantine Robustness*. PhD thesis, Massachusetts Institute of Technology, Aug. 1988.

[17] A. Snoeren, C. Partridge, L. Sanchez, C. Jones, F. Tchakountio, S. Kent, and T. Strayer. Hash-based IP traceback. In *Proc. ACM SIGCOMM*, Aug. 2001.

[18] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. Katz. Listen and Whisper: Security mechanisms for BGP. In *Proc. Symposium on Networked System Design and Implementation*, Mar. 2004.

[19] D. Zhu, M. Gritter, and D. Cheriton. Feedback based routing. In *Proc. ACM SIGCOMM HotNets Workshop*, Oct. 2002.