Vulnerabilities in distance-indexed IP traceback schemes

Jianping Pan*

Department of Computer Science,
University of Victoria,
Victoria, BC, Canada
E-mail: pan@uvic.ca
*Corresponding author

Lin Cai

Department of Electrical and Computer Engineering,
University of Victoria,
Victoria, BC, Canada
E-mail: cai@uvic.ca

Xuemin Sherman Shen

Department of Electrical and Computer Engineering,
University of Waterloo,
Waterloo, Ontario, Canada
E-mail: xshen@bbcr.uwaterloo.ca

Abstract: In order to counter Denial-of-Service (DoS) attacks using spoofed source addresses, many IP traceback schemes have been proposed in the last few years. Among them, distance-indexed probabilistic packet marking schemes appear to be very attractive. In this paper, we first discover two intrinsic vulnerabilities in these schemes. Substantiated by efficacy analysis and numerical results, several exploits are designed to take advantage of these vulnerabilities in an efficient manner when compared with the traceback effort attempted by victims. Consequently, we show that the design goal of these schemes can be compromised in practice. Further, we discuss these vulnerabilities in a general context relevant to network protocols and examine a few possible alternatives.

Keywords: IP traceback; probabilistic packet marking; Denial-of-Service (DoS) attacks; TCP/IP vulnerabilities; internet.

Reference to this paper should be made as follows: Pan, J., Cai, L. and Shen, X.S. (2007) 'Vulnerabilities in distance-indexed IP traceback schemes', *Int. J. Security and Networks*, Vol. 2, Nos. 1/2, pp.81–94.

Biographical notes: Jianping Pan is currently an Assistant Professor of Computer Science at the University of Victoria, British Columbia, Canada. He received a Bachelor's and a PhD in Computer Science from Southeast University, Nanjing, China in 1994 and 1998, respectively. From 1999 to 2001, he was a Postdoctoral fellow and then a Research Associate at the University of Waterloo, Ontario, Canada; from 2001 to 2005, he was a member of research staff at Fujitsu Labs and then a Research Scientist at NTT MCL in Silicon Valley, CA, USA. His area of specialisation is distributed systems and computer networks and his recent research interests include protocols for advanced networking, performance analysis of networked systems and applied network security. He is a member of ACM and the IEEE.

Lin Cai received the MASc and PhD with Outstanding Achievement in Graduate Studies Award) in Electrical and Computer Engineering from the University of Waterloo, Waterloo, Canada, in 2002 and 2005, respectively. Since July 2005, she has been an Assistant Professor in the Department of Electrical and Computer Engineering, University of Victoria, Victoria, Canada. She serves as an Associate Editor of the *ERUASIP Journal on Wireless Communications and Networking* and *The International Journal of Sensor Networks*. Her research interest focus on network protocol and architecture design supporting of multimedia traffic over wireless, mobile, ad hoc and sensor networks.

Xuemin Sherman Shen received a BSc (1982) from Dalian Maritime University (China) and an MSc (1987) and a PhD (1990) from Rutgers University, New Jersey (USA), all in electrical engineering. Since October 1993, he has been with the Department of electrical and Computer

Engineering, University of Waterloo, Canada where he is a Professor and the Associate Chair for Graduate Studies'. His research focuses on mobility and resource management in interconnected wireless/internet interworking, UWB, WiFi/WiMA sensor and ad hoc wireless networks. He is a co-author of three books and has over 200 publications in wireless communications and networks control and filtering. He serves as the Technical Program Chair for many IEEE and International conferences. He also serves as an Associate Editor for *IEEE Transactions Wireless Communications, IEEE Transactions Vehicle Technology*, etc. He is a senior member of the IEEE and a registered professional Engineer of Ontario, Canada.

1 Introduction

Denial-of-Service (DoS) attacks and their distributed variants (DDoS) have become a threat to the internet. Many internet service and content providers have suffered DoS/DDoS attacks, losing service capabilities, customers, and revenues. Besides other factors, the lack of *source accountability* in the TCP/IP protocol stack *enables* such attacks: attackers can forge their identities (i.e. the source IP address, protocol identifier and port number of their outgoing packets) when they have no intention to obtain services from DoS/DDoS victims, but just want to prevent legitimate users from doing so. The so-called *source spoofing* does not affect the destination-oriented internet routing fabric that can transport both attack and legitimate packets to victims. Ingress filtering (Ferguson and Senie, 2000) is a counter-spoofing measure, but its effectiveness does not appear until being considerably deployed and its efficiency with noncontinuous IP address blocks as well as compatibility with other schemes such as Mobile IP are still questionable.

On the other hand, *source traceability* is a victim-oriented approach towards accountability. With the assistance of anomaly and intrusion detection tools, victims or their agents first identify attack packets and then initiate a request that traces back towards the real sources of these packets. Traceback can occur in upper protocol layers (e.g. by correlating SMTP server signatures in e-mail headers), but the traceability of IP packets is essential, due to the fact that many DoS attacks do not exchange application-layer data at all. However, IP-level traceback is very challenging, since every IP packet is self-contained and can carry different source identities even from the same attack source. An *ideal* traceback scheme should correlate attack packets efficiently, identify or isolate attack sources effectively and more importantly, allow an incremental deployment over the internet. In addition, the scheme should be lightweight and only impose minimal changes to existing internet infrastructures (e.g. in core routers).

Many IP traceback schemes (Gao and Ansari, 2005) have been proposed in the last few years. In terms of how traceback characteristics are extracted and where the information is stored, most schemes follow these approaches: *router stamping* or *packet stamping*. In router-stamping schemes, a router identity (or its fraction), that is, *router stamp*, is stored in packets when they travel through routers. Victims collect packets carrying router stamps and recover a reverse path towards attackers, which is identified by the stamps of traversal routers. In packet-stamping schemes, routers keep a copy (or a digest) of forwarded packets, that is, *packet stamp*, for a while. Victims should initiate a traceback request within a certain time window, which is facilitated by a traceback authority consulting routers still having the matching packet stamps. In general, packet stamping incurs higher computation and storage overhead in routers. Thus, router stamping appears to be more attractive.

In this paper, we focus on Probabilistic Packet Marking with Compressed Edge Fragment Sampling (PPM/CEFS), a distance-indexed router-stamping scheme that has become a template for many follow-on traceback schemes. In a nutshell, PPM/CEFS overloads the 16-bit Identification (ID) field in the IP packet header by a PPM/CEFS router stamp consisting of offset index, distance field and mark fragment (see Section 2 for more design details). A stamping router, identified by a 64-bit node mark (i.e. its 32-bit IPv4 address and a 32-bit address hash), only has one of the following two choices when forwarding packets: with a given probability p, an 8-bit node mark fragment is independently inscribed in router stamps with an initialised distance field; otherwise, the distance field is deterministically incremented. Further, if the stamping router receives packets with an initialised distance field, it will incorporate its own node mark fragment with existing router stamps (i.e. by XOR) to create an edge mark fragment. Victims consequently reconstruct reverse paths from themselves towards attackers in a hop-by-hop, independent and possibly *postmortem* manner, by recovering and validating edge/node marks with the assistance of distance field, offset index and address hash.

PPM/CEFS is attractive due to its stateless, low-overhead and incrementally deployable design. Although PPM/CEFS becomes a template for other schemes, its own safety has not been thoroughly understood yet, especially when both the scheme and its parameters are known to the public (including attackers). The first of three possible router stamping operations (i.e. inscribe, increment and XOR) is considered safe. Although PPM/CEFS overwrites the IP ID field used by IP fragments for reassembly, not many internet flows are fragmented nowadays. The second one can become vulnerable when the incremented distance value is greater than what the allocated data structure can hold (i.e. buffer overflow). Also, a stamping router has no way to determine whether it indeed increments a valid distance field, other than a regular IP ID field or something staged by attackers. The third operation, instructed by the information in forwarded packets, is vulnerable, since the stamping router has no way to verify whether there is a node mark fragment of its upstream stamping router. Therefore, instead of adding its node mark to existing stamps by XOR, the stamping router can remove its node mark if such information is already there. Although PPM/CEFS tries to avoid the distance vulnerability by using 'saturating addition', its implication

and effectiveness have not been rigorously analysed and fully aware of yet. Further, the combination of the distance and XOR vulnerabilities is lethal: together with other PPM/CEFS weaknesses such as explosive reassembly space and weak hash protection, the design goals of these schemes can be compromised in practice and these unexpected consequences *cannot* be eliminated by just dropping or flagging overflowed packets without interfering with the design criteria and end-to-end communication guarantees. For example, with the extension, split, branch and synthesise exploits designed in this paper, a single attacker can easily emulate multiple attackers and render these schemes ineffective even in single-attacker scenarios.

Our contributions in this paper are threefold. Firstly, we investigate PPM/CEFS in a critical but practical environment and discover two possible vulnerabilities. Secondly, we design several exploits that can take advantage of these vulnerabilities in an effective and efficient manner when compared with the traceback effort attempted by victims. We further substantiate the feasibility of these exploits with efficacy analysis and numerical results. Thirdly, we examine the causes of these vulnerabilities and possible remedies, and discuss distance-related buffer overflow in a general context relevant to network protocols. Our goal in this paper is *not* to find vulnerabilities in some schemes, but to understand their consequences and how to avoid such implications in future design and implementation. Our results serve as a critical revisit to distance-indexed schemes, and are complementary to other traceback efforts.

The rest of this paper is organised as follows. In Section 2, we present the DDoS attack and IP traceback models, as well as an overview of probabilistic packet marking schemes. Related work, including other router-stamping and packet-stamping schemes, is also reviewed and compared. In Section 3, we reveal the vulnerabilities in PPM/CEFS-like schemes and their consequences. We then design several exploits that can take advantage of these vulnerabilities by creating different types of forged reverse paths. Section 4 gives an efficacy analysis of the designed exploits, substantiated by numerical results in Section 5. In Section 6, we offer further discussions and examine a few alternatives. Section 7 concludes this paper.

2 Background

2.1 Attack and traceback models

Figure 1 shows a conceptual model of regular and reflective DDoS attacks. An attack *master* (or a group of conspiring attackers) first compromises some third-party computers, and then converts them to attack *slaves*. During a certain time period, under the coordination of the master attacker, slave attackers flood a designated victim with a large number of attack packets that consume virtually all server and (near-server) network resources, which prevents legitimate users from obtaining desired services from the victim. Most attack packets carry spoofed source addresses to conceal the identities of slave attackers and to reduce the chance of exposing the master attacker.

Compared with Figure 1(a), reflective DDoS attacks shown in Figure 1(b) introduce a new layer of entities, *reflectors* (also known as *amplifiers*), between slave attackers and the victim. Unlike regular attacks in which a victim is identified by the destination address of packets sent by slave attackers, in reflective attacks, the victim is identified by the source address of these packets, while their destination addresses identify the chosen reflectors. On reception of these packets, a reply or an error message is automatically generated by reflectors and then sent *back* to the spoofed source of trigger packets, the victim. The introduction of reflectors further complicates the process of identifying or isolating slave attackers, while the goal of victim-initiated traceback is to accomplish this process without relying on the source address of attack packets.

Figure 1 Distributed denial of service attacks

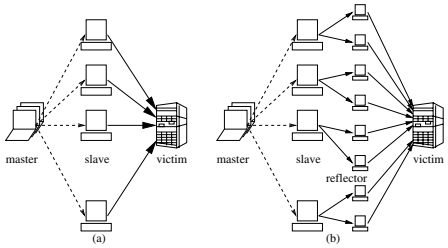
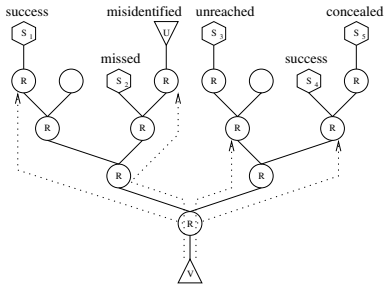


Figure 2 shows a model of reverse-path-based traceback, where a victim (V, i.e. the victim in Figure 1(a) or a reflector in Figure 1(b)) is the root of an attack tree and attackers (S, i.e. slave attackers in Figure 1) are the leaves of this tree. Sometimes, legitimate users (U) are also in the tree when they are *misidentified* as attackers. The inner nodes of the tree are traversal routers (R). When a traceback reaches an R that is close enough to an attacker (S_1 in Figure 2), it is considered a *success*. When the traceback follows a trail leaving away from an attacker (S_2), the attacker is bypassed. When the traceback stops before getting close enough to an attacker (S_3), the attacker is *unreached*. When the traceback successfully identifies an attacker (S_4) but omits another one (S_5) along the same attack path, the latter is *concealed* by the former.

Figure 2 Reverse-path-based IP traceback



For a traceback attempt, bypassed, unreached and concealed attackers are false negative outcomes; misidentified users are false positive ones. The objective of a traceback scheme is to eliminate false negatives and to minimise false positives. If attackers can increase false negative and false positive probabilities beyond certain thresholds, the traceback scheme becomes practically unusable. As we shall see, many proposed schemes suffer intrinsic vulnerabilities that allow an attacker to create a relatively large number of bypassed and misidentified outcomes, which can compromise the design goal of these schemes.

2.2 Probabilistic packet marking

Probabilistic Packet Marking with Compressed Edge Fragment Sampling (PPM/CEFS) (Savage et al., 2000; Savage et al., 2001) is a representative router-stamping scheme. There are several key techniques in this scheme. Firstly, PPM/CEFS overloads the mandatory 16-bit ID field in the IP header with the proposed router stamps. In its chosen design, a router stamp consists of an 8-bit mark fragment, a 5-bit distance field and a 3-bit offset index. A 64-bit node mark, identifying a stamping router, consists of a 32-bit IPv4 router address interleaved by a 32-bit hash of the address. A 64-bit edge mark, identifying one edge of two consecutive *stamping* routers along a forward path, is the XOR value of the node marks for these two routers. In each stamp, only one of the eight 8-bit edge (or node) mark fragments is inscribed and indexed by the 3-bit offset field. Secondly, with a given probability p, each router independently stamps packets with a fragment of its node mark and initialises the distance field to 0. Otherwise, with the remaining probability $1-p$, the distance field is incremented. In addition, if the updated distance field equals 1 the router XORs a fragment of its node mark (instructed by the existing offset index) with the stamp to create an edge mark fragment. Finally, with the assistance of offset index, distance field, and address hash in collected stamps, victims reconstruct and validate edge/node marks to recover a reverse path towards the first stamping router.

PPM/CEFS has a lightweight design: a stamping router may keep the existing mark intact; alternatively, it initialises or updates a stamp with its own mark. Consequently, it only incurs very little per-packet overhead and is scalable to large-capacity routers. Although this scheme overloads the IP ID field, traffic measurement shows that only a small fraction of internet flows requires packet fragmentation and PPM/CEFS can be in force only when victims request so (Savage et al., 2000). There are concerns for this scheme with IPSec and IPv6 in which the IP ID field is either protected or non-existent. However, such concerns also occur in other router-assisted schemes. Another router stamping scheme, similar to PPM/CEFS, is independently proposed in Doeppner et al. (2000). Modelled as a constrained minimax optimisation problem, (Park and Lee, 2001) gives a comprehensive analysis on the effectiveness of PPM/CEFS with regard to marking probability, path length and traffic volume. In addition, Adler (2002) presents a theoretical framework on the trade-offs between the size of router stamps and the number of attack packets required for path recovery in PPM/CEFS-like schemes.

PPM/CEFS is known to be less effective with multiple attackers. Due to mark fragmentation, a victim has to explore a combinatorial explosion space when reconstructing edge marks along different paths but at the same distance away from itself. A solution is proposed in Advanced and Authenticated Marking Schemes (AMS) (Song and Perrig, 2001): instead of edge fragments indexed by PPM/CEFS offset, an 8-bit hash of the source address of forwarded packets and the identity of forwarding routers is treated as a node mark in AMS, with a 3-bit flag ID indicating one of eight independent hash functions designed to reduce hash collisions. AMS requires victims to have a forward path graph of *stamping* routers (not just *regular* routers) to verify (not identify) whether a router has its mark in the collected stamps. A further performance improvement appears in Yaar et al. (2005). Adjusted PPM (APPM) (Peng et al., 2002) gives routers far away from victims a higher stamping probability to reduce the number of attack packets required for path recovery. However, the router location awareness assumed in APPM may not be always feasible with spoofed packets.

Another concern for PPM/CEFS is due to the limited length (i.e. 32-bit) of address hashes. Since not all attack packets are stamped by legitimate routers before reaching victims, attackers can seed them with some specially-crafted stamps, hoping that fake stamps will reach victims intact. Although PPM/CEFS marks are protected by hashes, each mark contains an address fragment and a hash fragment. Attackers have the chance to create fake stamps that can reconstruct *valid* (but fake) edge/node marks with genuine stamps, by applying the concept of *Groups of Strongly Similar Birthdays* (GOSSIB) (Waldvogel, 2002). Such fake edges appear only beyond the perimeter of stamping routers, but as we shall see soon, fake edges within the perimeter can also be created by attackers.

2.3 Other traceback schemes

Many IP traceback schemes have appeared in the literature. Instead of node/edge marks, an algebraic approach is attempted by Dean et al. (2002) to populate router stamps with a partial accumulator calculated at each stamping router according to Horner's rule. Victims then need to solve an array of equations to reconstruct the original polynomial representing the forward path. However, in practice, this process is far-from-trivial when there are random full or partial paths. Chen and Lee (2003) gives an extension of this scheme to handle reflective attacks. In a deterministic marking scheme (Belenky and Ansari, 2003), if a stamping router *knows* that it is the first router for a packet along its forward path, a unique router stamp is stored in this packet and will not be replaced by follow-on routers. However, unless a complete perimeter is established beforehand, individual routers are unlikely to know their location for packets with spoofed addresses.

Source Path Isolation Engine (SPIE) (Sanchez et al., 2001; Snoeren et al., 2001, 2002) is a representative packet-stamping scheme. Instead of logging packets, SPIE-capable routers digest the first 28 bytes of IP packets, excluding some hop-by-hop header fields. To conserve the log space, SPIE adopts Bloom filters to store these packet stamps by using a set of hash functions. With a controllable false positive probability, SPIE can verify whether a packet has been digested in a filter by using the same set of hash functions again. Victims can submit even a single packet to a traceback manager that has access to these filters and verifies whether the packet has been forwarded by a particular SPIE-capable router. In addition, SPIE introduces a lookup table to correlate packet digests before and after packet transformation. With such a space-saving design, SPIE roughly consumes 0.5% of link capacity per unit time in storage to have an acceptable performance measure; this overhead can be a concern for large-capacity routers. In addition, traceback requests

should be submitted within a certain time window before SPIE-capable routers purge accumulated packet stamps. A lower probability to digest packets among consecutive routers, which lessens the storage requirement, is attempted by Li et al. (2003); but this approach further increases the complexity during the digest and recovery processes. A layer-2 extension to hash-based IP traceback is proposed by Hazeyama et al. (2003), which tries to identify attackers within a subnet by their link layer identities (e.g. MAC addresses).

IETF has initiated an ICMP-based traceback scheme with newly introduced iTrace messages (Bellovin, 2000). In contrast to router stamping in which in-band stamps may be replaced or updated by downstream routers, in iTrace, with a very low probability, a router generates an out-of-band ICMP message sent directly to the destination of a trigger packet with the identity of the router and a partial copy of the packet. Messages are cryptographically protected and bypass all downstream routers. Victims verify message integrity and router authenticity and then identify the traversal router *directly* (instead of in a hop-by-hop manner). Although it is robust against DDoS attacks (Kuznetsov et al., 2002), iTrace requires more attack packets to identify stamping routers due to its ultra-low probability of generating iTrace messages, introduces considerable network and router overhead and relies on an authentication key infrastructure not available yet. An intention bit (Wu et al., 2003) is introduced in routing table for iTrace and is enabled only when victims request so; requested routers then can generate iTrace messages with a higher probability. These intention requests should be authenticated; otherwise, they can become a form of DoS attacks towards iTrace-capable routers. New ICMPv6 messages are proposed by Lee et al. (2003) to support similar traceback activities in IPv6 networks.

There are path-based traceback schemes as well. In Pi (Yaar et al., 2003) and StackPi (Perrig et al., 2003), packets carry a bunch of tiny router stamps (e.g. a 2-bit mark per router) in the IP ID field, which collectively identifies a forward path. Victims then trace back towards attackers with this path identifier and a forward path graph. However, even with these tiny stamps, there is still insufficient space in a single packet that can accommodate all necessary stamps for a relatively long forward path (e.g. more than 8 hops of stamping routers with 2-bit marks and 16-bit total space).

Among so many proposed schemes, PPM/CEFS and follow-on schemes appear attractive due to their stateless, low-overhead and incrementally deployable design. These schemes allow victims to independently trace back towards attackers in a *postmortem* (i.e. after attacks) and hop-by-hop manner without much ISP involvement and are considered promising in countering DoS attacks and holding attackers accountable. As a result, it is of utmost importance to ensure their own 'safety'.

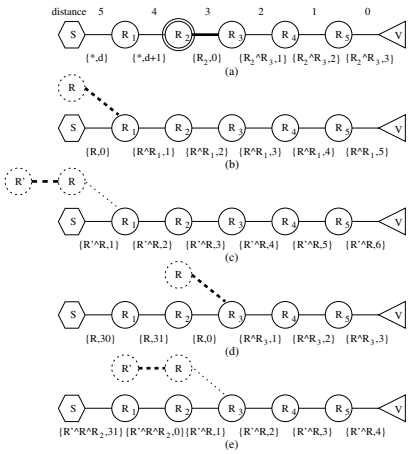
3 Vulnerabilities and exploits

In this section, we first reveal vulnerabilities that turn out to be common to many distance-indexed marking schemes (including PPM/CEFS, AMS, APPM and others). We then design several exploits that can actually take advantage of these vulnerabilities by creating a relatively large number of forged paths in a very efficient manner. In doing so, attackers can defeat any traceback efforts attempted by victims employing these schemes.

3.1 Overflow and XOR vulnerabilities

In distance-indexed traceback schemes, a stamping router (e.g. R_1, R_3, R_4, or R_5 in Figure 3(a)) only increments the distance field of router stamps (denoted as {node/edge mark, distance} in Figure 3, and Δ for XOR) in forwarded packets, unless a stamping router (R_2) initialises a stamp and the subsequent router (R_3) updates the stamp. Therefore, victims can distinguish stamps created by routers at different distances away from themselves. This property is employed by victims to recover a reverse path in a hop-by-hop manner towards the first stamping router. Another property is that attackers *cannot* spoof stamps produced by two consecutive stamping routers closer to victims than the stamping router closest to attackers. This property is referred to as *perimeter control* or *line-of-defense* in distance-indexed traceback schemes.

Figure 3 Overflow and XOR vulnerabilities



Unfortunately, the second property is intrinsically vulnerable to buffer overflow and XOR exploitation. Without loss of generality, we use PPM/CEFS as an example in this paper; the results are applicable to other distance-indexed traceback schemes as well. In PPM/CEFS, the distance field has a 5-bit counter, which is supposed to index router stamps for a maximum 31-hop path. With a given probability p, a router (R_2 in Figure 3(a)) overloads the router stamp of a forwarded packet with its own node mark and initialises the distance field to 0. With the remaining probability $1 - p$, the router increments the distance field in the forwarded packet. Also, if the post-updated distance field equals 1 (or the preupdated distance field equals 0), the router (R_3) updates the existing stamp with its own node mark by a simple bitwise XOR operation.

This marking scheme should follow open protocols and adopt well-known parameters to promote an incremental deployment. Therefore, an attacker, with the knowledge of such a scheme and its parameters, can also initialise a router stamp with an arbitrary node mark (as R in Figure 3(b)). The

downstream stamping router (R_1) will introduce a fake edge (R, R_1) by incorporating its own node mark with probability $1-p$. This fake edge mark (5 hops away from the victim) can survive its journey to the victim with probability $(1-p)^5$, if none of the total five stamping routers between the attacker and the victim in this example reinitialises the router stamp. In addition, the attacker can introduce more fake edges at the same or farther-away distances by further utilising this fake edge (see Figure 3(c)). PPM/CEFS is aware of this vulnerability and argues if the first stamping router (R_1) closest to the attacker is in a stub domain, an edge leading to another stub domain or a transit domain can be excluded manually. However, when the first stamping router is in a transit domain, which is very likely in an incremental deployment, the victim completely loses its capability to exclude fake edges, since in general it has no idea about the distance from the attacker to itself; otherwise, the attacker has been already isolated.

Another buffer overflow and XOR exploitation is even more lethal. Instead of initialising the distance field, an attacker can arrange a relatively large distance value with an arbitrary node mark (as R in Figure 3(d)). For example, the attacker creates a stamp of initial distance 30 in a 5-bit distance field. With probability $(1-p)^3$, the stamp survives its journey beyond a stamping router R_3 that is 3 hops away from the attacker. Since R_3 has an updated (and overflowed) distance of value 1 for this stamp, it will also introduce another fake edge (R, R_3) by incorporating its own node mark. Such an edge mark (2 hops away from the victim) has probability $(1-p)^2$ to complete its journey to the victim. Overall, the attacker can successfully deliver this fake edge to the victim with probability $(1-p)^5$. In addition, more fake edges that are farther away to the victim than R_3 but are closer than the attacker can be delivered with the same probability $(1-p)^5$, that is, none of the all 5 stamping routers reinitialises the router stamp. In Figure 3(e), the router stamp specially-crafted by the attacker contains the node marks of two fake routers (R' and R) not along the attack path and the node mark of R_2 along the attack path. When R_2 incorporates its own node mark using XOR, R_2 essentially removes its node mark from the router stamp.

One may propose stamping routers (such as R_2 in Figure 3(d) and R_1 in Figure 3(b)) to manually drop packets (similar to the procedure when IP TTL expires) or mark stamps when their distance field is about to overflow. However, such an attempt creates more problems in an incremental deployment. Although non-fragmented IP packets do not rely on the IP ID field for reassembly, the IP ID field does contain a meaningful (usually sequential within a flow) identifier and may incur a *normal* overflow if stamping routers increment the *imaginary* distance field. Dropping such overflowed packets can cause undesired packet losses of legitimate flows and break existing applications. If stamping routers mark overflowed stamps and instruct downstream routers and victims not to react to these stamps, an attacker can emulate the same procedure and mislead victims to ignore all attack packets.

The only way to handle these vulnerabilities correctly is to change the stamping behaviour in PPM/CEFS. Instead of the *deterministic* read-increment-write operation, routers should follow a new, *conditional* read-increment-test-overwrite procedure. In other words, if a router encounters an overflowed distance field when incrementing it, in addition to writing back the overflowed distance value, it should inscribe its own identity as well, which is the *only* behaviour expected by the immediate downstream router. Alternatively, routers can follow a read-increment-test-nowrite procedure; that is, if an overflow is encountered, no write back is performed, which is similar to 'saturating addition' in Savage et al. (2000). Here, we recommend the *overwrite* approach, since it fully emulates the behaviour expected by downstream routers, guarantees the sanity of stamps received by victims from routers that follow the newly-proposed distance increment procedure and does not bring extra combinatorial explosion. While with the *nowrite* approach, all overflowed stamps, even having different router identities, appear to be inscribed by stamping routers at the same distance away from victims. However, these remedies also have other implication. The overwrite or nowrite approach alters the expected distribution of stamps received from different routers. Without sanity check, forged stamps delivered to routers directly through IP-in-IP tunnels or other perimeter-breaking means can still exploit the XOR vulnerability.

3.2 *Overflow and XOR exploits*

After demonstrating how attackers inject fake edges, we show how attackers can systematically create different types of forged reverse paths in PPM-like traceback schemes, with or without genuine edges. To facilitate our discussions, we make the following assumptions:

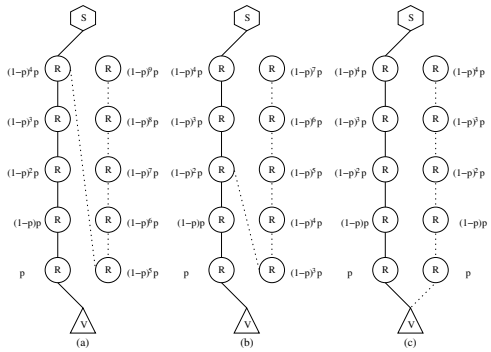
- The traceback scheme and its control parameters are publicly known to victims and attackers.
- Stamping routers either independently initialise router stamps by their node marks, or increment the distance field in stamps (or regular ID fields) and update stamps with their node marks if necessary, but they are unaware of their locations in the attack tree and cannot verify upstream stamps without introducing considerable overhead.
- Attackers know their locations in the attack tree with regard to a particular victim, can create many packets with fake stamps and have obtained a copy of genuine stamps produced by legitimate routers in advance.

The first assumption should be endorsed by all traceback schemes, since only an open protocol and unified parameters can promote an incremental deployment over the internet. If stamping routers follow different protocols and have proprietary parameters, the incompatibility among them will prevent these schemes from being widely adopted in practice. The second assumption is explicit in distance-indexed traceback schemes; it essentially describes the router procedures defined by these schemes, no matter how stamps are actually specified. The last assumption is crucial but also practical and acceptable. It is always possible for attackers to know their positions and attack paths when choosing victims. Attackers have no intention to obtain services from victims and can create any kind of packets and stamps. Stamping routers cannot distinguish between attack and legitimate

packets without incurring considerable overhead, so attackers can easily arrange sampling packets to go through routers and obtain a copy of their genuine stamps. Later, we will have more discussions on these assumptions.

Figure 4 shows three types of exploits. In Figure 4(a), an *extension* of reverse path is made from the stamping router closest to an attacker. This exploit allows the attacker to be bypassed from a reverse-path-based traceback. To achieve this goal, the attacker first creates a fake edge to another router from the closest router (as shown in Figure 3(b)) and then generates more fake edges from this extended router (as shown in Figure 3(c)). In doing so, the attacker has created an extended reverse path bypassing the attacker. To protect fake stamps against statistical analysis employed by the victim, the number of fake stamps should follow a certain distribution. Figure 4 shows the probability of received stamps initialised by routers (including fake ones) at different distances away from the victim, where p is the stamping probability of individual routers. In order to make sure that fake edges are indistinguishable from genuine ones, the length of this forged extension of reverse path *may* depend on p.

Figure 4 Basic exploits

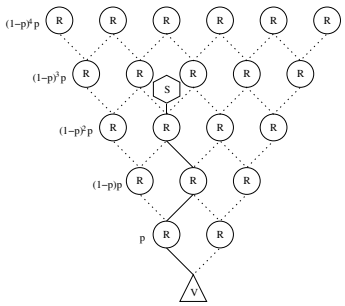


As argued by PPM/CEFS, if the closest router is in a stub domain, it is very unlikely that more genuine edges could be extended from this router. Therefore, an attacker should identify a stamping router in a transit domain and *split* the reverse path as early as possible. Figure 4(b) shows such an exploit. By applying the technique shown in Figure 3(d), the attacker can first create a fake edge to another router from any stamping router. Then, the attacker can repeat the technique shown in Figure 3(e) to generate more fake edges from the extended router. In doing so, two reverse paths, one genuine and the other one partially-forged, are created by the attacker. In addition to ensuring the number of fake edges follows the intended distribution, the attacker can also control the number of total attack packets and the number of splits, again depending on p, in order to conceal its location (i.e. close to a stamping router) along the genuine path as much as possible.

Branch, the last exploit shown in Figure 4(c), is an extreme case of split. Here, two reverse paths are created, one genuine and the other one fully forged. Similar to the procedures in Figure 4(b), an attacker can first create a fake edge to another router directly from the victim and then generate more fake edges from the extended router. Depending on the attacker location and the chosen p, the attacker can introduce a number of branches of certain hops, which are indistinguishable from the viewpoint of the victim. Both branch and split will increase the number of bypassed and misidentified traceback outcomes; if the attacker can control the number of total attack packets properly, they will also increase the number of unreached outcomes.

More exploits can be built upon these basic ones. In addition to introducing extensions, splits and branches, attackers can intentionally choose stamping routers between victims and themselves as the extended routers, so they can create *loop* and *multi-path* in the reverse-path-based traceback tree, which further confuses the victim. For example, in Figure 5, an attacker successfully creates multiple overlapped reverse paths from the victim and appears to be close to any of the 20 stamping routers.

Figure 5 Synthesised exploits



4 Efficacy analysis

We have shown that attackers can effectively introduce forged paths in distance-indexed traceback schemes. We now substantiate these exploits with efficacy analysis. Numerical results are presented in the next section.

4.1 Original traceback schemes

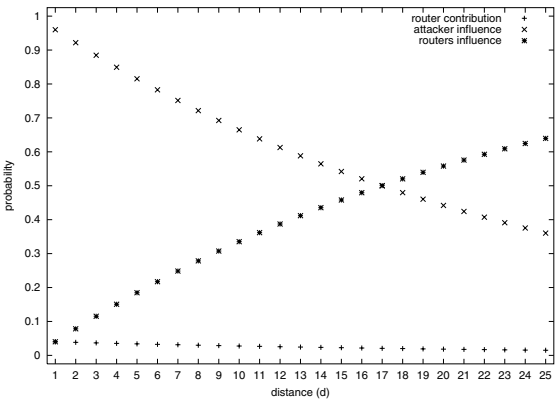
For an attacker that is D hops away, in terms of the number of stamping routers between a victim and itself, the probability of delivering its fake stamps to the victim is $(1-p)^(D)$; that is, none of these D stamping routers initialises router stamps with their marks. The stamping router closest to the attacker has probability $(1-p)^{D-1}p$ to deliver its stamp to the victim; that is, it initialises router stamps with its mark and none of the remaining $D-1$ stamping routers reinitialises these stamps again. For clarity, we do not consider mark fragmentation here, but our analysis can be extended accordingly when mark fragmentation is used. It takes at least $[(1-p)^{D-1}p]^{-1}$ packets on average for the victim to receive the mark of the stamping router closest to the attacker. Since edge sampling is adopted, to fully decode the identity of this router, the victim also needs to have the identity of the stamping router next closest to the attacker. Therefore, the total number (n) of packets required for the victim to recover a path to the stamping router closest to the attacker is bounded by

$$n \geq \frac{K(D)}{(1-p)^{D-1}p} \tag{1}$$

where $K(D)$ is a function of path length D. By conservatively assuming that all stamping routers have the same probability $(1-p)^{D-1}p$ to deliver their marks, PPM/CEFS gives an estimate on $K(D) \sim \mathcal{O}(\ln(D))$, which is suggested by the coupon collector's problem. The traceback scheme has the freedom to choose p and $p = D^{-1}$ if victims want to minimize the number of attack packets needed to identify the stamping router closest to attackers. However, D is unknown before the attacker is identified, so $p = D_{\max}^{-1}$ is chosen with an empirical estimate ($D_{\max}$) on D. Once p is determined by a traceback scheme, it remains relatively stable to be adopted by all stamping routers.

In their chosen design, PPM/CEFS and many follow-on schemes have $p = 0.04$, given the fact that measurement shows few paths over the internet exceed a length of $D_{\max} = 25$ hops. Hence, we adopt the same $D_{\max}$ and assume $D \leq 12$, since in most occasions attackers are much closer to victims and only some traversal routers are stamping-capable. In Figure 6, we show the influence of an attacker that is d hops away from a victim to deliver fake stamps, compared with the cumulative influence of all d stamping routers to deliver genuine stamps. When $d = 12$, the attacker has a probability of $(1-0.04)^{12} = 0.61$ to deliver fake stamps to the victim, while the closest router only has a probability of $(1-0.04)^{11} \times 0.04 = 0.03$ to deliver a genuine stamp containing its mark to the victim. Even when $d = 25$, the attacker still has a considerable influence of 0.36 to deliver fake stamps to the victim. Only when $d > 17$, stamping routers have greater *cumulative* influence on stamp delivery, but the contribution from individual routers is still much less when compared with the influence that the attacker has.

Figure 6 Routers vs attacker influence ($p = 0.04$)



4.2 *Efficacy of constrained exploits*

To extend the reverse path to a farther away distance $d > D$ and to defeat statistical analysis employed by victims, an attacker should arrange fake stamps from the router $D+1$ hops away from a victim being delivered to the victim with probability $(1-p)^D p$, from the router $D+2$ hops away with $(1-p)^{D+1}p$, and from the router d hops away with $(1-p)^{d-1}p$. Given the influence $(1-p)^D$ that the attacker of D hops away from the victim has, the possible path extension (d) is bounded by

$$(1-p)^D p + (1-p)^{D+1}p + \cdots + (1-p)^{d-1}p \leq (1-p)^D$$

which can be rewritten as

$$\frac{(1-p)^D p[1-(1-p)^{d-D}]}{1-(1-p)} \leq (1-p)^D$$

or

$$(1-p)^{d-D} \geq 0 \tag{2}$$

where (2) holds whenever $d > D$ and $0 < p < 1$. This result implies that the reverse path can be extended by the attacker, regardless of its location, without an upper bound. We will have some practical discussions later.

To branch a reverse path from a victim and to keep the forged path of d hops statistically indistinguishable, an attacker should arrange fake stamps from the router 1 hop away from the victim being delivered with probability p, from the router 2 hops away with $(1-p)p$ and from the router d hops away with $(1-p)^{d-1}p$. Similarly, given the influence $(1-p)^D$ that the attacker has, the possible length (d) of the forged path is bounded by

$$p + (1-p)p + \cdots + (1-p)^{d-1}p \leq (1-p)^D$$

which can be rewritten as

$$\frac{p[1-(1-p)^d]}{1-(1-p)} \leq (1-p)^D$$

or

$$(1-p)^d \geq 1-(1-p)^D$$

Since $0 < 1-p < 1$,

$$d \leq \frac{\log[1-(1-p)^D]}{\log(1-p)} \tag{3}$$

that is, branch is bounded after p and D are determined. From the viewpoint of the attacker, having $d = D$ is sufficient in most occasions. Therefore, when

$$D \leq \frac{-\log 2}{\log(1-p)} \tag{4}$$

it is always possible for the attacker D hops away to create a forged path of the same or greater length from the victim. When $p = 0.04$, the attacker 17 hops away from the victim still has enough influence to create a forged path of the same length, as indicated in Figure 6. When $D < 10$, the attacker can even create more than one forged path, which further confuses the victim attempting a reverse-path-based traceback.

Furthermore, assume that an attacker wants to introduce m splits from the genuine path at a stamping router that is d hops away from a victim and that these forged splits have the same length D as the genuine path. Given the influence $(1-p)^D$ that the attacker has, the number (m) of splits is bounded by

$$m[(1-p)^d p + (1-p)^{d+1}p + \cdots + (1-p)^{D-1}p] \leq (1-p)^D$$

or

$$m \leq \frac{(1-p)^{D-d}}{1-(1-p)^{D-d}} \tag{5}$$

When $d = D-1$, m is maximised as $m_{\max} = 1-p/p$. In other words, when $p = 0.04$, the attacker can create about

24 fake paths, besides the genuine one, splitting at the next closest stamping router. However, the attacker always wants d to be as small as possible when compared with D in the split case (i.e. far away from the attacker). With a given m, the location of the split point (d) is bounded by

$$(1-p)^{D-d} \geq \frac{m}{1+m}$$

or

$$d \geq D - \frac{\log m - \log(m+1)}{\log(1-p)} \tag{6}$$

When $p = 0.04$ and $m = 7$, $d \approx D - 3$; that is, the attacker can create 7 splits at a stamping router that is 3 hops away from the attacker. When $m = 3$, $d \approx D - 7$.

To apply synthesised exploits as shown in Figure 5 (i.e. extending the genuine path to d hops and introducing forged paths of d hops branching at the victim or splitting from the genuine path), an attacker that is D hops away from a victim should properly arrange the distribution of router stamps delivered to the victim from genuine and fake routers. Given the influence $(1-p)^D$ that the attacker has, d is bounded by

$$\begin{aligned} & p + 3p(1-p) + \cdots + (2D-1)p(1-p)^{D-1} \\ + \quad & 2(D+1)p(1-p)^D + \cdots + 2dp(1-p)^{d-1} \\ \leq \quad & (1-p)^D \end{aligned}$$

which can be rewritten as

$$\sum_{i=0}^{d-1} 2(i+1)p(1-p)^i \leq 1 \tag{7}$$

(7) is equivalent to

$$\frac{2 - 2(1+dp)(1-p)^d}{p} \leq 1$$

or

$$(dp+1)(1-p)^d \geq \frac{2-p}{2} \tag{8}$$

When $p = 0.04$, $d_{\max} \approx 5$; that is, a single attacker can inject at least 25 fake edges in Figure 5 to conceal its proximity location close to any of these 20 stamping routers.

4.3 Efficacy of unconstrained exploits

So far, our analysis assumes that fake edges follow certain distributions similar to their genuine counterparts, in order to let victims believe forged paths are genuine. Also, attackers can inject fake edges with arbitrary distributions after they have destroyed the statistical characteristic that the genuine path should have. In doing so, all paths, including the genuine one, appear to be forged from the viewpoint of victims if statistical analysis is applied, which forces victims to give up any traceback attempts.

Without edge distribution constraints, attackers are only concerned about how to deliver fake stamps and how to corrupt the distribution of genuine stamps. For an attacker that is D hops away from a victim, its fake stamps can survive their journey to the victim with probability $(1-p)^D$; that is, none of the D stamping routers between the victim and itself initialises router stamps. This is the total capability that the attacker has to inject fake stamps. For example, the attacker can inject genuine stamps in a forged manner to create a uniform distribution of received stamps. When $p = 0.04$ and $D = 12$, the cumulative routers influence is only 0.39, while the attacker influence is 0.61; therefore, the attacker has enough influence to change the distribution of genuine stamps. In practice, the attacker can even achieve this goal with much less effort.

While delivering fake stamps, an attacker also needs to calibrate the risk of exposing itself. With p, the router closest to the attacker of d hops away from a victim only has a probability of $p(1-p)^{d-1}$ to deliver its stamps. Given the attacker influence of $(1-p)^d$, a fake stamp can be reinitialised by the closest router with probability $p(1-p)^{d-1}$, or reinitialised by other stamping routers with $1-(1-p)^d - p(1-p)^{d-1}$. Assume that n is the number of fake stamps that the attacker is willing to send before being exposed by the closest router. The probability of n follows

$$P(x=n) = [1 - p(1-p)^{d-1}]^n p(1-p)^{d-1} \tag{9}$$

When $d = 12$ and $p = 0.04$, the expectation of n is

$$E(n) = \sum_{n=1}^{\infty} nP(x=n) \approx 32.3$$

that is, the attacker can safely inject 32 fake stamps on average without being exposed by the closest stamping router, when it is 12 hops away from the victim. If mark fragmentation is adopted in the traceback scheme (e.g. a PPM stamp only contains one of eight 8-bit mark fragments), $E(n)$ can be considerably increased.

However, most DoS attacks require many more than 32 attack packets to be effective, so an attacker needs to relax its concern and to allow more attack packets with fake stamps. Actually, the attacker can easily eliminate its concern of being exposed by the closest router, if the reverse path has been extended or splits and branches have been introduced. Even if the closest stamping router has its stamp successfully delivered to a victim, the victim-initiated traceback is neither conclusive nor effective, since there are many forged reverse paths (one of them bypassing the attacker) having similar statistical characteristics, which prevent the victim from identifying the attacker and its proximity location in the attack tree.

5 Numerical results

We now substantiate our analysis with numerical results to demonstrate how effective and efficient these exploits are in practice. Unless otherwise explicitly stated, we assume that an attacker is $D = 12$ hops away from a victim adopting a distance-indexed traceback scheme with $D_{\max} = 25$ (or $p = 0.04$). For each setting, our experiment is repeated 1000 times; in each run, 1000 packets are sent by the attacker. All router stamps are collected by the victim for path recovery and offline analysis. As mentioned in Section 4, we do not consider mark fragmentation here and similar results can be obtained accordingly in the cases when router marks are fragmented.

5.1 Constrained extension exploit

First, we show how effectively an attacker can create a partially-forged reverse path bypassing the attacker. Figure 7 shows the number and its statistical moments (including mean, median and standard deviation values in vertical axis) of attack packets required for a victim to recover a reverse path of d hops (in horizontal axis). The rectangle in the figure represents the range of 25–75% quantiles.

In Figure 7(a), the attacker is indeed 24 hops away from the victim (i.e. a long path). It normally takes about 128–201 packets (with mean 172.855, median 156 and standard deviation 69.368) for the victim to recover a genuine long path of 24 hops to the attacker. When router stamps only store one of 8 mark fragments such as those in PPM/CEFS, it takes more than 2000 packets on average for the victim to recover a path of the same length. On the other hand, in Figure 7(b), the attacker is actually 12 hops away. As we can see, the attacker can create a partially-forged path of 24 hops, of which the first half (hops 1–12) are genuine and the second half (hops 13–24) are forged by fake router stamps. With fake stamps, it normally takes 123–202 packets (with mean 172.223, median 159.5 and standard deviation 69.524) for the victim to recover the partially-forged path of 24 hops, which is almost identical to the effort to recover a genuine path of 24 hops.

When compared, Figure 7(a) and (b) appear to be indistinguishable from the viewpoint of the victim. We try to confirm this proposition by applying statistical tests on the datasets representing two path recovery processes shown in Figure 7. These datasets fail to pass Komogorov-Smirnov (KS) Lilliefors tests, which implies that they do not follow a normal distribution; indeed, they are related to a geometric distribution as suggested by (1), and we cannot apply the usual t-Test on their mean values. Therefore, we resort to Mann-Whitney U test, which is the most powerful and sensitive (in terms of rejecting the null hypothesis) nonparametric alternative to the t-test for independent samples. The z-value for the U test and the probability with which H_0 (the null hypothesis) holds, are listed in Table 1. In our U-test, H_0 suggests that there is no significant difference in median values between these two datasets. None of P_{H_0} in Table 1 is significant enough to reject the H_0 hypothesis with a confidence level of 0.05 (H_0 is rejected when $z > 1.960$), which confirms that these two path recovery processes are statistically indistinguishable from the viewpoint of the victim.

Therefore, even when a victim has collected enough router stamps and recovered a reverse path of 24 hops, the victim still has no information on the proximity location (i.e. close to a particular stamping router) of an attacker among this path, unless the attacker has its closest stamping router in a stub domain.

Figure 8 further plots the distribution of router stamps, received by the victim, from a stamping router d hops away. The victim can apply this statistical analysis, so it *should* know whether the collected stamps have been poisoned by fake ones. Unfortunately, an attacker can easily bypass this analysis by intentionally faking stamps that follow the expected distribution. Figure 8(a) shows the percentage, its statistical moments, and the expected distribution of router stamps for a genuine long path of 24 hops. The average stamp percentage fits quite well with the expected distribution. However, as shown in Figure 8(b), the attacker 12 hops away from the victim can also have the average stamp percentage satisfy the expected distribution of a genuine path very well,

Figure 7 Number of packets required for the victim to recover a reverse path (extension): (a) a genuine long path ($D = 24$) and (b) a partially-forged path ($D = 12$)

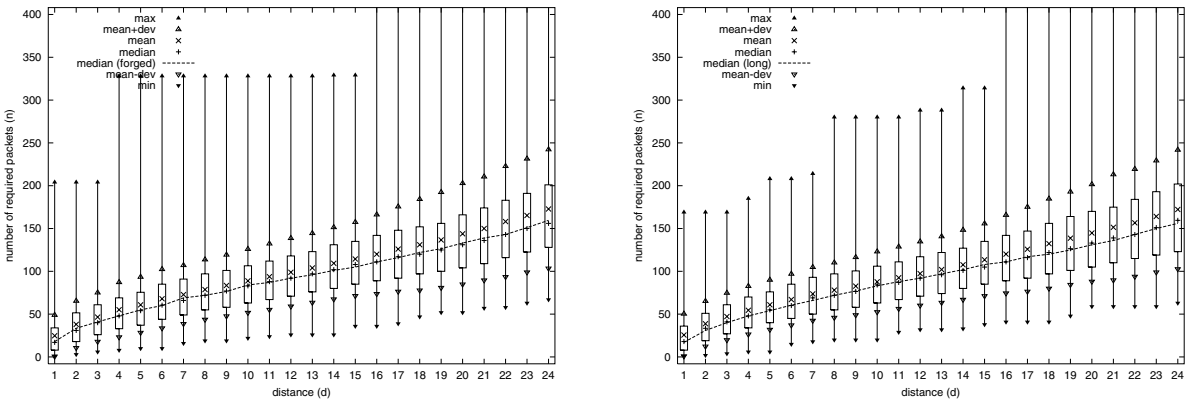
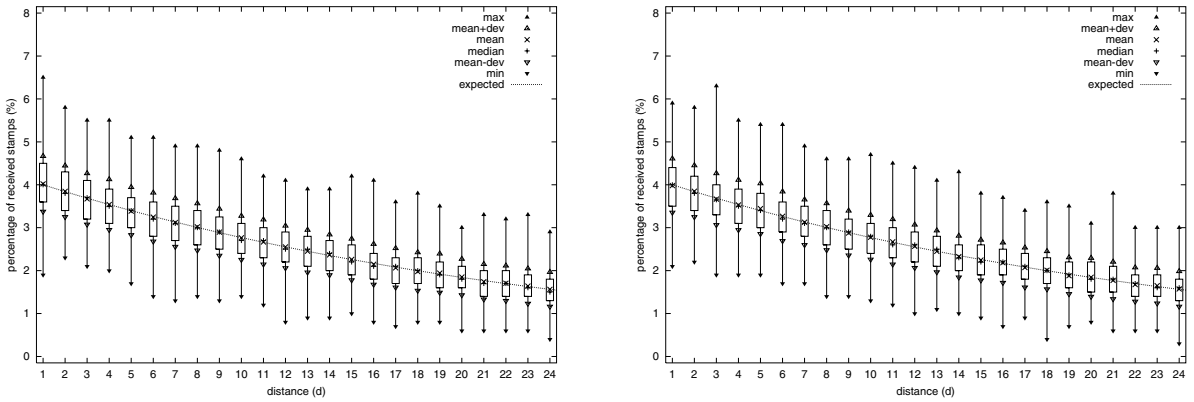


Table 1 Statistical indistinguishableness between two reverse paths: U-test

hops	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
z-Value	0.759	1.180	1.076	0.511	1.059	0.279	1.370	0.182	0.128	0.223	0.260	0.538
P_{H_0}	0.454	0.238	0.282	0.609	0.290	0.780	0.171	0.855	0.898	0.824	0.795	0.591
hops	13	14	15	16	17	18	19	20	21	22	23	24
z-Value	0.993	0.946	0.459	0.011	0.243	0.666	1.101	0.946	0.485	0.365	0.362	0.185
P_{H_0}	0.321	0.344	0.646	0.991	0.808	0.505	0.271	0.344	0.627	0.715	0.717	0.853

Figure 8 Percentage of stamps received by the victim (extension): (a) a genuine long path ($D = 24$) and (b) a partially-forged path ($D = 12$)



(a) (b)

no matter for the genuine portion (hops 1–12) or the forged portion (hops 13–24) of the extended path, which makes these stamps statistically indistinguishable from those for the genuine long path.

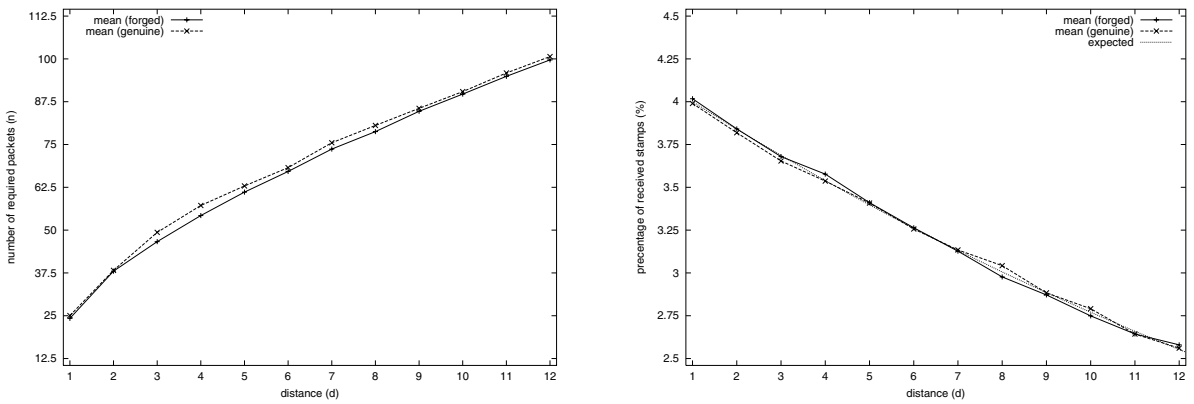
As analysed in Section 4.2, attackers always have enough influence to extend a genuine path bypassing themselves without an upper bound, no matter where attackers are located. In practice, there are other issues for which attackers to trade-off. The farther away the path is extended, the more packets (which also contain more genuine stamps) are required for victims to collect, in order to actually recover the path to that extent. There is a risk for attackers that the victim-initiated path recovery may prematurely stop at a location that is closer to attackers, due to the limited number of attack packets (and fake stamps). Generally, it is sufficient for attackers to extend a reverse path to a router $2D$ hops away from victims.

5.2 Constrained branch exploit

Next, we show that attackers also have the advantage when creating a fully forged path by applying the branch exploit designed in Section 3.2. To save space, Figure 9(a) only plots the average number of packets required for a victim to recover a reverse path of 12 hops, when an attacker is 12 hops away from the victim. The attacker also injects fake router stamps that identify a fully forged path branching directly at the victim (see Figure 4(c)). On average, it takes almost the same amount of effort for the victim to recover the genuine path and the forged path (this proposition is also confirmed by statistical tests). Thus, when the victim has collected enough router stamps to recover the genuine path, it has also collected enough stamps to recover the forged path. There is no way for the victim to tell which one, if any, of these identified paths is genuine.

Figure 9(b) further validates the feasibility of a fully forged path branching at the victim. In this figure, the average percentage and the expected distribution of router stamps from a stamping router that is d hops away from the victim are plotted for both genuine and forged paths. An attacker can arrange fake stamps properly so that these stamps can pass the statistical analysis at the victim. From the viewpoint of the victim, these two paths are identical. According to the analysis in Section 4.2, when $D < 10$, the attacker even has enough influence to create more forged paths of the same length as the genuine one, with the expected stamp distribution. As we mentioned, split exploit is a special case of branch and extension exploits. To conserve space, we omit presenting numerical results for the case with split forged paths in this section.

Figure 9 Constrained branch exploit: (a) number of required packets and (b) percentage of received stamps



(a) (b)

Easily forging a branch also raises another concern with mark fragmentation. Basically, branch represents a very efficient way for a single attacker to emulate multiple attackers. Recall that PPM/CEFS faces a combinatorial explosion where there are multiple attackers. With attacker emulation, even when there are no multiple attackers, a smart attacker can always inject many fragmented router stamps that belong to different paths but are the same distance away from a victim, which creates a combinatorial explosion space and forces the victim to explore.

5.3 Unconstrained exploits

With the constraint of the expected stamp distribution, attackers can create forged paths that appear genuine. On the other hand, attackers can destroy the expected distribution that the genuine path should have, making the genuine path appear as a forged one from the viewpoint of victims. Here, we show that such a goal is highly achievable with very little effort from attackers.

Figure 10(a) shows a poisoned genuine path that is identified by *genuine* stamps forcedly injected by an attacker. In this experiment, the attacker injects the router stamp of a stamping router d hops away from a victim with probability $p_d = (1-p)^{D-d}/P$, where $P = \sum_{d=1}^{D} p_d$. Therefore, the victim has a probability of $p_d\, p(1-p)^{d-1} = p(1-p)^{D-1}/P$ (i.e. a constant) to receive router stamps *from* any stamping routers. Basically, the attacker creates more stamps for routers farther away from the victim, since their stamps are more likely to be overwritten by routers closer to the victim. We can see a considerable reduction in the number of packets required for the victim to recover a reverse path in Figure 10(a). Compared with Figure 9(a), in which the victim collects more than 100 packets on average to recover a path of 12 hops, with forcedly injected genuine stamps, it only takes less than 40 packets to achieve the same result, which is actually staged by the attacker.

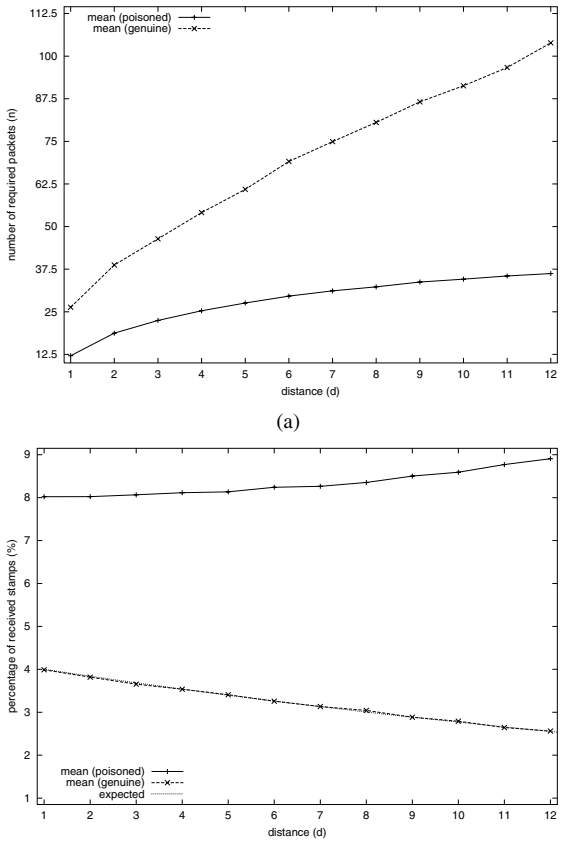
Figure 10(b) shows the distribution of stamps collected by the victim. With forcedly injected genuine stamps, the average percentage of received stamps *from* individual routers no longer satisfies the expected distribution. If the victim applies statistical analysis against the collected stamps, the analysis will reject these stamps, since they do not follow the required distribution, even though they still identify a genuine path.

Certainly, attackers have no incentive to inject stamps only identifying the genuine path, which accelerates the process for victims to trace back toward themselves. However, attackers can combine this technique together with other exploits, so that it becomes more efficient for attackers to inject fake stamps that identify forged paths, especially when neither fake stamps nor genuine stamps follow the expected distribution.

6 Further discussions

We have identified the vulnerabilities shared by many distance-indexed traceback schemes and demonstrated that several developed exploits can effectively and efficiently compromise the design goal of these schemes. We now discuss the distance-related vulnerability in a general context and examine some possible alternatives that *may* be used to strengthen these vulnerable traceback schemes.

Figure 10 Unconstrained exploit (poisoned genuine path): (a) number of required packets and (b) percentage of received stamps



(a)

(b)

6.1 Practical considerations

Firstly, we discuss the feasibility of these exploits in a practical environment. To create fake edges, attackers should know their distance to victims and obtain a copy of router stamps from legitimate routers. Also, attackers should know the network topology between victims and themselves, as well as the identity and location of the entities misidentified in forged reverse paths. There are many topology discovery tools available for attackers to obtain a comprehensive knowledge about victims and their surroundings. According to these traceback schemes, stamping routers are identified by their IP address (or a known transformation of IP address) to enable an independent path recovery by victims; therefore, attackers can easily obtain and forge router stamps. Even if a secret is involved in generating genuine stamps, attackers can arrange sampling packets going through stamping routers, since their stamps are supposed to be valid for a while.

Secondly, we consider a scenario when there are multiple conspiring attackers. We mentioned that a single attacker can emulate multiple attackers and force victims to explore a combinatorial explosion when mark fragmentation is adopted. When there are multiple attackers, in addition to

the combinatorial explosion, these attackers can share the knowledge about forged paths. Collectively, they have much greater influence on injecting fake stamps and creating forged paths, when compared with the contribution to path recovery from individual routers. For example, when there are m conspiring attackers, the extent (d) of the synthesized exploits shown in Figure 5 are bounded by

$$\sum_{i=0}^{d-1} 2(i+1)p(1-p)^i \leq m$$

where d is the longest distance between any stamping routers and victims. When $m = 2$, $d_{\max} \approx 8$ and at least 48 fake edges are injected by two attackers in a structure similar to that shown in Figure 5. When $m = 10$, $d_{\max} \approx 20$ and at least 200 fake edges are injected. Therefore, attackers can easily conceal their locations by creating a large number of forged paths with similar characteristics and destroying the expected distribution of genuine paths.

Finally, we show that distance overflow is undetectable and unpreventable by the existing mechanisms in these traceback schemes. Overflow happens when a variable is assigned to a value greater than its allocated structure allows; similar buffer overflow exists in many software packages. Obviously, stamping routers have no way to verify the already-embedded stamps themselves. When a router increments the distance field of a stamp and encounters an overflow, it has no idea whether this is a genuine one (or a regular IP ID) or a fake one staged by attackers; therefore, the router should not drop overflowed packets. If routers flag overflows in forwarded stamps, victims can exclude these stamps from statistical analysis and path recovery. However, attackers can flag every stamp (including genuine ones) and victims completely lose their capability to collect any information about the genuine path if all flagged stamps are excluded. In either scenario, attackers have achieved their goal of defeating traceback attempts. Given the fact that distance-indexed operations occur in many network protocols and distributed applications, some of them may also be susceptible to the distance overflow exploitation presented in this paper.

6.2 Alternatives and limitations

Distance overflow is the intrinsic vulnerability that leads to the aforementioned exploits in distance-indexed traceback schemes. There might be some approaches to circumventing these exploits. First, if the distance field can be totally eliminated from router stamps (e.g. iTrace with its own strengths and weaknesses), attackers have no chance to exploit this vulnerability, assuming that other stamp fields are invulnerable to buffer overflow exploitation. However, if the path has to be recovered in a hop-by-hop manner or the node/edge marks are fragmented, victims have to rely on other means to recover the reverse path. Goodrich (2002) proposes a *randomize-and-link* scheme employing large checksum *cords* (instead of the distance field that is also vulnerable to combinatorial explosion) to link and verify mark fragments from a stamping router when the router delivers a message (e.g. its identity) to victims. Assuming that the checksum has sufficient randomness and is unpredictable in advance, attackers cannot interfere with the process of recovering this message at victims. However, such an approach still does not eliminate false but verifiable messages injected by attackers. The checksum verification requires considerable computation and storage resources and it can become a form of DoS attacks if not well-protected.

When the distance field is required in a hop-by-hop path recovery scheme, another approach is to reduce the influence that attackers can have; in this case, even if attackers inject false information, the attacker influence should be negligible when compared with the contribution from individual routers. Park and Lee (2001) and Adler (2002) discussed the trade-offs among the marking probability, the mark size, the length of attack path, and the number of attack packets required to conclusively identify an attacker. An attacker D hops away from a victim has the influence $(1-p)^D$ on the victim, which can be reduced by the victim increasing p or can be increased by the victim reducing D. However, when increasing p, the victim also increases its effort to trace back toward a faraway attacker. Peng et al. (2002) tries to increase the contribution from individual routers with a variable p, especially for those far away from victims, in order to reduce the number of required attack packets. However, with this approach, unless a perimeter is established beforehand or attackers have been roughly isolated, it is still very difficult, if not impossible, to reduce the attacker influence (Rizvi and Fernandez-Gaucherand, 2003) without sacrificing the contribution from routers that are actually close to attackers.

Finally, attackers can freely inject false information in a legitimate format without much effort. To reduce the amount of false information, one can increase the cost for attackers to produce it. For example, if router stamps contain randomness related to the secret that only the legitimate stamping routers have, victims can exclude fake stamps that do not have the desired characteristics. However, to enable victims to recover a reverse path independently, they should learn the router secret appropriately (e.g. by a time-released key chain (Song and Perrig, 2001)). Also, routers should avoid attackers sampling their stamps (e.g. stamps being specific to a particular destination). These techniques undoubtedly increase the complexity of stamping routers and may reduce their chance of being widely adopted; thus, the trade-off should be balanced properly.

In summary, PPM/CEFS was designed under many realistic constraints (e.g. backward compatibility) and is indeed very attractive and promising due to its stateless, low-overhead, and incrementally-deployable design. On the other hand, these constraints unavoidably introduce some vulnerabilities in its design and implementation. There are several remedies that can alleviate these vulnerabilities to certain extent, but none of them can completely eliminate these implications without interfering with the design goal and criteria of these schemes.

7 Conclusions

In this paper, we discovered a few vulnerabilities that are common to many distance-indexed probabilistic packet marking schemes proposed to counter DoS/DDoS attacks and to achieve source accountability. To reveal the consequences,

we designed several effective exploits that can take advantage of these vulnerabilities in a very efficient manner, especially when compared with the traceback efforts attempted by victims employing these schemes. Practical considerations and limitations of possible alternatives were also discussed, as well as distance-related buffer overflow in a general context relevant to network protocols.

In addition, our work offers two more guidelines to the design and implementation of any future traceback schemes:

1 incrementing the value of fix-sized fields should always be protected by the boundary check of such fields and

2 operations instructed by the information that already exists in forwarded packets (e.g. XOR in PPM/CEFS) should be avoided as much as possible if the sanity check of the existing information is infeasible or unrealistic.

References

Adler, M. (2002) 'Tradeoffs in probabilistic packet marking for IP traceback', *Proceedings of 34th ACM Symposium on Theory of Computing (STOC'02)*, pp.407–418.

Belenky, A. and Ansari, N. (2003) 'IP traceback with deterministic packet marking', *IEEE Communications Letters*, Vol. 7, No. 4, pp.162–164.

Bellovin, S. (2000) 'ICMP traceback messages', *IETF Internet Draft*.

Dean, D., Franklin, M. and Stubblefield, A. (2002) 'An algebraic approach to IP traceback', *ACM Transactions on Information and System Security*, Vol. 5, No. 2, pp.119–137.

Chen, Z. and Lee, M. (2003) 'An IP traceback technique against denial-of-service attacks', *Proceedings of ACSA Annual Computer Security Application Conference (ACSAC'03)*, pp.96–105.

Doeppner, T., Klein, P. and Koyfman, A. (2000) 'Using routing stamping to identify the source of IP packets', *Proceedings of Seventh ACM Computer and Communications Security (CCS'2000)*, pp.184–189.

Ferguson, P. and Senie, D. (2000) 'Network ingress filtering: defeating denial of service attacks which employ IP source address spoofing', *IETF RFC 2827*.

Gao, Z. and Ansari, N. (2005) 'Tracing cyber attacks from the practical perspective', *IEEE Communications Magazine*, Vol. 43, No. 5, pp.123–131.

Goodrich, M. (2002) 'Efficient packet marking for large-scale IP traceback', *Proceedings of Ninth ACM Computer and Communications Security (CCS'02)*, pp.117–126.

Hazeyama, H., Oe, M. and Kadobayashi, Y. (2003) 'Layer-2 extension to hash-based IP traceback', *IEICE Transactions on Information and System*, E86-D(11), pp.2325–2333.

Kuznetsov, V., Simkin, A. and Sandstroem, H. (2002) 'An evaluation of different IP traceback approaches', *Proceedings of Fourth International Conference on Information and Communications Security (ICICS'02)*, pp.38–48.

Lee, H., Thing, V., Ma, M. and Xu, Y. (2003) 'On the issues of IP traceback for IPv6 and mobile IPv6', *Proceedings of Eighth IEEE Int'l Symposium on Computers and Communications (ISCC'03)*, pp.582–587.

Li, J., Sung, M., Xu, J., Li, L. and Zhao, Q. (2003) 'Large-scale IP traceback in high-speed internet: practical techniques and theoretical foundation', *Georgia Tech Technical Report*, GIT-CC-03-32.

Park, K. and Lee, H. (2001) 'On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack', *Proceedings of 20th IEEE INFOCOM (INFOCOM'01)*, pp.338–347.

Peng, T., Leckie, C. and Ramamohanarao, K. (2002) 'Adjusted probabilistic packet marking for IP traceback', *Proceedings of Second IFIP-TC6 Conference on Networking (Networking'02)*, pp.697–708.

Perrig, A., Song, D. and Yaar, A. (2003) 'StackPi: a new defense mechanism against IP spoofing and DDoS attacks', *Carnegie Mellon University Technical Report*, CMU-CS-02-208.

Rizvi, B. and Fernandez-Gaucherand, E. (2003) 'Analysis of adjusted probabilistic packet marking', *Proceedings of Third IEEE IP Operations and Management (IPOM'03)*, pp.9–13.

Sanchez, L., Milliken, W., Snoeren, A., Tchakountio, F., Jones, C., Kent, S., Partridge, C. and Strayer, W. (2001) 'Hardware support for a hash-based IP traceback', *Proceedings of Second DARPA Information Survivability Conference and Exposition (DISCEX-II)*, pp.146–152.

Savage, S., Wetherall, D., Karlin, A. and Anderson, T. (2000) 'Practical network support for IP traceback', *Proceedings of 16th ACM SIGCOMM (SIGCOMM'2000)*, pp.295–306.

Savage, S., Wetherall, D., Karlin, A. and Anderson, T. (2001) 'Network support for IP traceback', *IEEE/ACM Transactions on Networking*, Vol. 9, No. 3, pp.226–237.

Snoeren, A., Partridge, C., Sanchez, L. and Jones, C. (2001) 'Hash-based IP traceback', *Proceedings of 17th ACM SIGCOMM (SIGCOMM'01)*, pp.3–14.

Snoeren, A., Partridge, C., Sanchez, L., Jones, C., Tchakountio, F., Schwartz, B., Kent, S. and Strayer, W. (2002) 'Single-packet IP traceback', *IEEE/ACM Transactions on Networking*, Vol. 10, No. 6, pp.721–734.

Song, D. and Perrig, A. (2001) 'Advanced and authenticated marking schemes for IP traceback', *Proceedings of 20th IEEE INFOCOM (INFOCOM'01)*, pp.878–886.

Waldvogel, M. (2002) 'GOSSIB vs. IP traceback rumors', *Proceedings of 18th ACSA Annual Computer Security Applications Conference (ACSAC'02)*, pp.5–13.

Wu, C., Wu, S. and Zhang, L. (2003) 'On design and evaluation of Intention-driven ICMP traceback', *Proceedings of 12th IEEE International Conference on Computer Communications and Networks (ICCCN'03)*, pp.159–165.

Yaar, A., Perrig, A. and Song, D. (2003) 'Pi: a path identification mechanism to defend against DDoS attacks', *Proceedings of 24th IEEE Security and Privacy (SP'03)*, pp.93–107.

Yaar, A., Song, D. and Perrig, A. (2005) 'FIT: fast internet traceback', *Proceedings of 24th IEEE INFOCOM (INFOCOM'05)*, pp.1395–1406.