# GOSSIB vs. IP Traceback Rumors

Marcel Waldvogel
IBM Research
Zurich Research Laboratory
mwl@zurich.ibm.com

***Abstract*—To identify sources of distributed denial-of-service attacks, path traceback mechanisms have been proposed. Traceback mechanisms relying on probabilistic packet marking (PPM) have received most attention, as they are easy to implement and deploy incrementally. In this paper, we introduce a new concept, namely *Groups Of Strongly SImilar Birthdays* (GOSSIB[1]), that can be used by to obtain effects similar to a successful birthday attack on PPM schemes. The original and most widely known IP traceback mechanism, *compressed edge fragment sampling* (CEFS), was developed by Savage et al. [SWKA00]. We analyze the effects of an attacker using GOSSIB against CEFS and show that the attacker can seed misinformation much more efficiently than the network is able to contribute real traceback information. Thus, GOSSIB will render PPM effectively useless. It can be expected that GOSSIB has similar effects on other PPM traceback schemes and that standard modifications to the systems will not solve the problem.**

## I. Introduction

The February 2000 distributed denial-of-service (DDoS) attacks brought down some of the largest sites on the Internet for several hours by flooding them with packets, causing link and server overloads [PPS00, LRST00, Mar00]. A first-hand account of an attack and the resulting experiences with ISPs, tools, and the attackers is given in [Gib02]. Recently, even an Internet Service Provider (ISP) had to close shop, claiming that continuous DDoS attacks made business operation impossible [Ric02]. These attacks are assumed to be launched by an individual or a small group of people, taking advantage of the openness of the Internet infrastructure and the insecurity of many systems [PPS00]:

*a) Spoofing:* Every end system can create packets with arbitrary source addresses. This capability is used to hide the sender identity, but can be used to have innocent systems further flood the host listed in the fake source address by sending replies. Owing to the effort involved, ISPs are currently unwilling to install filters at their customer links that restrict customer source addresses to the valid range.

*b) Broadcast Amplification:* Certain network messages, such as ICMP Echo Request ("ping"), will be replied to, even when sent to a broadcast address. Sending packets with a spoofed return address to many broadcast addresses in other networks will cause these other, unsuspecting, hosts to swamp the owner of the return address.

*c) Lack of Appropriate Response to Attacks:* Many organizations ignore messages indicating an attack originating from within their site, making it difficult to close down attacking sources.

*d) Unprotected Computers:* It is often very easy for attackers to start "owning" systems unauthorizedly, i.e., by breaking into them and installing tools that turn these systems into willing slaves. They are used as intermediaries to provide both amplification and tracing protection to the attacker.

It turns out that, with the current Internet infrastructure, such attacks are almost impossible to *prevent*. Proposals that would allow hosts being flooded to tell their upstream routers to filter packets for them exist [PL00, MBF+01, IB02]. However, without clever security infrastructures and authentication frameworks for these filtering requests, such a system can be abused to provide even more sophisticated denial-of-service (DoS) attacks.

Even just *identifying* the sources or intermediaries by tracing the traffic back towards the originators involves such a large amount of manpower that it is close to impossible. This is unlikely to change, because source addresses are easily faked and Internet Service Providers (ISPs) are resisting to install inbound filters that would prevent address spoofing. In the aftermath of the February 2000 large-scale DDoS attacks, researchers therefore started working on providing traceback solutions that would not require any protocol changes and could be incrementally deployed. Recent research [MVS01] using scatterback analysis shows that DDoS activities are still ongoing in the current Internet, without getting the publicity of the February 2001 attacks. Furthermore, these attacks also seem to result in less impact, possibly due to not being coordinated well enough and due to the lack of the surprise element.

One of the earliest and most widely known schemes for Internet Protocol (IP) traceback, *compressed edge fragment sampling* (CEFS), was developed by Savage et al. [SWKA00, SWKA01]. CEFS is simple and lightweight to implement on routers where the forwarding process can be influenced at some stage, requires no protocol changes, does not increase network traffic, and can be deployed incrementally: all features of a likely candidate for widespread acceptance and use. The routers

[1] **"Gossib" is also an early version of today's "gossip," which relates to the sharing of information among groups, where the information is typically changed only slightly.**

mark the passing IP packets with a predefined probability (probabilistic packet marking, PPM). These marks can then be used by the victim to determine the paths the arriving packets had taken, i.e. to determine the originating organizations—mostly themselves victims of a large-scale break-in—and tell them to take appropriate action. Incremental deployment and minimal impact on existing infrastructure and compatibility with current traffic or implementations are key requirements. Yet, in the current Internet protocol, almost all bits have assigned functions. Therefore, only a few remaining bits and some bits that can be shared with their existing purpose remain available to convey the necessary information. This requires the information about the current edge to be split into small chunks, which have to be reassembled at the receiver.

The reassembly of these chunks is key to path reconstruction, but also the weak spot of CEFS. Already in their papers introducing CEFS [SWKA00, SWKA01], the authors describe how the scarcity of bits available for transmission of the chunks renders the reconstruction more complex, a problem that is further compounded by the presence of uninitialized chunks, filled with random data.

This paper shows that by carefully selecting the chunks to transmit, an attacker can seed misinformation *much more efficiently than the network is able to contribute real traceback information.* GOSSIB is an algorithm that can be used to create the chunks to transmit and thus add false path components to the state being searched, reducing the usefulness of the traceback information to almost zero.

The paper is organized as follows. In Section II, we describe PPM and CEFS. In Section III, we show the vulnerability, and describe the GOSSIB way of choosing chunks. In Sections IV and V, we provide a theoretical analysis and a simulation of GOSSIB, respectively. Before drawing the conclusions in Section VII, we present and critique other possible traceback solutions in Section VI.

## II. Compressed Edge Fragment Sampling

CEFS is a specific algorithm of the general PPM family. The basic idea behind PPM is to have routers label a subset of transit packets with information about the router labeling router, thus enabling the receiver to reconstruct the path back to the source. The result of a successful reconstruction enables a target to identify the attack origin, typically followed by out-of-band mechanisms to stop the attack or prevent further attacks.

To achieve this, the label at least has to contain information identifying the originating router. Furthermore, some authenticating information should be included, which can range from simple sanity checks (e.g., whether the reconstructed topology makes sense) to cryptographic authentication information. Encoding the link (i.e., edge of the graph) by indicating both endpoints is one possibility to encode the network route as well as to enable a simple sanity check.

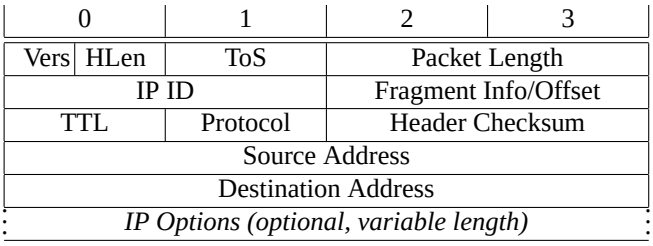



Fig. 1. IP version 4 header format

CEFS is based on this edge encoding. Unfortunately, there is no place in the existing Internet Protocol (IP) packet header to encode the edge (see Figure 1). Transmitting the information in an IP option has to be ruled out, as most routers handle packets with IP options very slowly. Among the fields in the IP header, the most expendable are *ToS* and *IP ID*. The ToS (Type of Service) byte was originally planned to request type-specific treatment. This byte has since been converted to encode Differentiated Services Code Points (DSCP) [NBBB98] and Explicit Congestion Notifications (ECN) [RFB01]. The IP ID field is used for matching IP packet fragments during reassembly. CEFS opted to use these 16 bits as fragmentation is increasingly rare today and further measures are taking to improve backwards compatibility.

Thanks to the creative reuse of the IP ID field, 16 bits are available to encode an edge, a tuple of two 32-bit IP addresses. The problem is solved as follows:

1) The edge is encoded as the bitwise exclusive-or of the two router IDs.
2) To enable reconstruction when there is only a single path between attacker[2] and victim, the relative position of the encoded edge in the path chain as well as the ID of either end needs to be known. The latter is satisfied, as the victim's ID is known, and the former can be provided for by including a hop counter.
3) To enable reconstruction in the presence of multiple paths (e.g., due to multiple attackers), the hop count does not uniquely identify the position of the edge in the graph. The ID of a router is thus not just its IP address, but is created by bitwise interleaving its IP address and a hash of the same IP address. For a good hash function, the result of the exclusive-or between two such IDs is unique enough to allow quite an accurate reconstruction of the graph.
4) The resulting 64 bit address is split into 8 chunks ("edge fragments," EF) of 8 bits each. The EF is sent together with a 5-bit hop count ("distance") field and 3 bits indicating which of the 8 possible edge fragments ("offset") is being sent.

[2]In the following, we consider any machine actively generating attack traffic as an attacker.

**Listing 1** Attack graph reconstruction

(See Table I for legend information.)

```
T ← Set of all received EF tuples
Initialize reconstruction graph G
P ← {victim's address(es)}
for all p ∈ P do
   Add node for p to G
end for
for all increasing distances d do
   N ← ∅
   F ← {∀t ∈ T : t.distance = d}
   for all 8-tuple combinations t from F with distinct offset
   fields do
      Set t.address and t.hash according to the tuple fields
      for all p ∈ P do
         n.address ← p.address ⊕ t.address
         n.verifier ← p.verifier ⊕ t.verifier
         if n.verifier = H(n.address) then
            N ← N ∪ {n}
            Add node for n to G
            Add directed edge (p, n) to G
         end if
      end for
   end for
   P ← N
end for
```

The IP ID field is thus populated with the information shown in Table I.

The edges of the attack-path network are reconstructed in order of increasing distance from the victim. For each distance, all possible ordered 8-tuples with distinct *offset* fields are examined, to verify whether they result in a valid matching node ID (i.e., IP address interleaved with hash verifier) when paired with any of the addresses identified when running the preceding distance iteration. Pseudo-code for the reconstruction is presented in Listing 1.

Note that because of space limitations, the above description of CEFS is complete and accurate only to the extent that is required to understand the GOSSIB attack and the analysis.[3] For an in-depth description of CEFS, consult [SWKA01].

## III. GOSSIB Introduction

State explosion is inherent to CEFS graph reconstruction. The idea behind GOSSIB is twofold. First, insert misleading edges into the graph. Second, simultaneously and purposefully increase the state space to be searched, creating an excessive state explosion beyond the victim's capacity.

[3]The main difference is that two adjacent routers are actually involved in filling the IP ID field, which simplifies incremental deployment. This also slightly changes the first step in the reconstruction process.

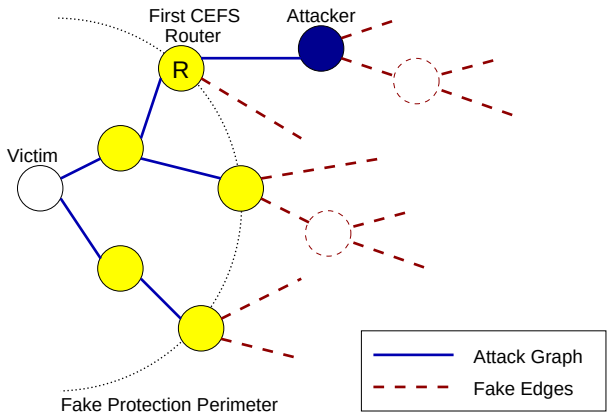



Fig. 2. Edge faking

Increasing the state space is done by increasing the EFs. The enumeration of the resulting 8-tuples is exponential in the number of tuples. Already small increases in the number of tuples would thus be highly desirable from an attacker's point of view, as it delays the reconstruction of the attack graph. For this to work, the EFs themselves do not need to be carefully chosen, any random value will do. This is outlined in Section III-B.

Adding fake edges to the graph is even more desirable, as it results in more work of the victim's network administrator, namely to determine which ISPs and/or networks are involved and asking them to disconnect and fix the attacking machines. In addition, the path reconstruction work is also proportional to the number of nodes identified at the previously calculated distance. Edge faking is introduced in Section III-A.

The GOSSIB way of attacking CEFS is described in Section III-C. It combines the above two mechanisms such as to increase state space and the number of fake edges, at the same time minimizing the number of packets that the attacker needs to send to achieve this purpose.

### A. Edge Faking

The CEFS description in [SWKA01] enforces the saturating increment of the distance field as packets travel through the network. Therefore, it is impossible for an attacker to fake an edge that is closer (from the CEFS viewpoint) than the first CEFS-capable router $R$ its packets have to pass through (cf. Figure 2). For a single attacker, the path reconstructed by the victim will have no branches up to $R$. As an edge will only be recognized when one of the nodes it connects to is already present in the graph, lone edges cannot be introduced.

Accordingly, that part of the graph cannot be influenced, allowing the victim to narrow down the position of the attacker. Nevertheless, an attacker may transmit fake edges from $R$ to other hosts, from itself to other hosts, and from fake hosts created in this way to even other hosts, masking itself as an innocent transit router.

TABLE I
ENCODING EDGE FRAGMENTS INTO THE IP ID FIELD

| Field | Offset | Distance | Address EF | Hash EF |
|---|---|---|---|---|
| # Bits | 3 | 5 | 4 | 4 |
| Value | $i \in [0,7]$ | hop count | $\mathit{this}_i \oplus \mathit{prev}_i$ | $H_i(\mathit{this}) \oplus H_i(\mathit{prev})$ |

**Legend:** $i$ is chosen randomly. The 4 bits of $x$ starting at bit $4i$ are indicated using $x_i$, where $x$ is any symbol. $H(\cdot)$ is a hash function. *this* and *prev* are the addresses of the current and the preceding router.

With multiple attackers (as is the case in DDoS), the attackers closest to the victim will be unable to hide themselves, but they can introduce a sufficient number of false paths to increase the amount of resources needed by the victim and significantly delay reliable detection of the bulk of the attackers. Any attacker can add fake edges even to other parts of the attack graph, as long as the distance requirement is met. Therefore, a few attackers close to the source can aid in having the state space explode and innumerable fake edges being created, widely distributed over the entire graph.

### B. Standard IP Stack Attack

The simplest attack against CEFS is not to use an attack at all, but simply delegate the "attack" to the standard way of implementing IP IDs, where the IP stack continuously enumerates all possible $2^{16}$ IP IDs.

The CEFS marking rate is proposed as $1/25 = 4\%$ in [SWKA01]. This means that a machine which has to send via 17 CEFS routers will still get about $50\%$ of its packets through unchanged. As it is expected that only a subset of the routers will perform CEFS, the distance as measured in actual router/link hops will be much higher than the number of CEFS routers. Even at the maximum CEFS distance of 32 CEFS hops, about $27\%$ of the packets' IP IDs will get through unmodified.

As attackers at CEFS distance $d$ have the distance field of any packet incremented (saturating) by $d$, only $t = 2^{11} \cdot (32 - d)$ distinct IP IDs may arrive at the destination (for a 16-bit field split into 5 bits of distance and 11 bits of other information). At a marking rate of $1 - p$, an attacker needs to send out only $p^d t$ packets to get $t$ packets through to the victim. These $t$ packets are not all distinct, but will cover a great variety of combinations. The exact value of the expected distribution is not critical, but the inclined reader can use the well-known solution to the *coupon collector's problem* [FGT87, Fel66] for a more detailed analysis.

This "attack" against CEFS is quite inefficient if run by only one machine, as the increment operation insures that no hops less than $d$ CEFS hops from the victim may be faked. Therefore, the first CEFS router in the attack path (closest to the attacker), $R$, will be clearly identified as being in the path. This fact typically reveals enough information to locate the ISP from which the attack originates.

In the DDoS case however, with many machines along different paths and at different distances, the closest machine(s) at distance $d$ can efficiently mask those further away, as described above. Thus, the closest machines first need to be identified and fixed, before traceback may identify the machines at $d+1$, rendering the cutting off of the attack packet stream a lengthy and painful process.

The impact of this "attack" can be minimized by counting the frequency of the individual IP IDs and eliminating those below a threshold based on the expected arrival rate of untampered IP IDs for a given distance. In the next section, we describe how even a statistical analysis can be fooled, as GOSSIB enables attackers to insert false edges with fewer packets than legitimate CEFS routers require to transmit information about an actual edge.

### C. GOSSIB Attack

A standard CEFS router tries to convey a single edge reliably to the receiver, which requires the successful arrival of all the fragments. An attacker, on the other hand, is not interested in accuracy (i.e., quality) of information, only in the quantity: he wants to generate as many fake edges as possible, the actual identity of these edges is at most secondary. The attacker may thus reach that goal by sending out a bunch of packets that are designed such that they can be used to create more than one edge. After the attacker has determined where to add fake edges, he sends out packets of which some are part of multiple reconstructions.

Is it reasonable to assume that such bunches of packets can be found? CEFS uses a cryptographic hash function to assure the integrity of the reconstruction. As these hash functions are known for their resistance against hash collisions, the result of this effort seems unlikely or at least disproportional to the computational effort required. This first intuition turns out to be untrue. The reasons include the fact that the hash function size is limited to only 32 bits and that by replacing part of the IP address, we are also allowed to replace part of the hash, thus increasing our chances. Thus, not a full-fledged hash collision is required for the attack to work, only a much more frequent *near-birthday* (or GOSSIB) collision, as described below.

Figure 3 shows a pair of such near collisions, in which only a single nibble differs between the respective interleaved address-verifier IDs. Recall that for a given distance, a matching pair of (address, verifier) nibbles is transmitted in each marked packet (Table I), labeled with the appropriate nibble offset. The hash

**(Address, verifier) pair similarity.**

| Address | Verifier |
|---|---|
| 0x00a18 **4** e0 | 0xd0246 **7** 1c |
| 0x00a18 **a** e0 | 0xd0246 **5** 1c |

**Messages sent out.**
Format: (distance, offset $i$, address$_i$, verifier$_i$)

$(x, 7, 0, \mathrm{d})$
$(x, 6, 0, 0)$
$(x, 5, \mathrm{a}, 2)$
$(x, 4, 1, 4)$
$(x, 3, 8, 6)$
$\boldsymbol{(x, 2, 4, 7)}$ $\boldsymbol{(x, 2, \mathrm{a}, 5)}$
$(x, 1, \mathrm{e}, 1)$
$(x, 0, 0, \mathrm{c})$

Fig. 3. Near-birthday collision example (nibble values given in hexadecimal)

function used for all examples and simulations is MD5 [Riv92]. While similar results hold for other "collision-free" hash functions, simpler hash functions are expected to be even easier to abuse.

In our example, only $9$ distinct messages are necessary to transmit $2$ fake edges ($4.5$ messages per edge), whereas a CEFS router requires $8$ distinct messages to communicate a single (true) edge to the victim.

Near-collisions for collision-free hash functions have to be computed using brute force. This is done by enumerating all "close" node addresses, and checking whether the resulting hashes are "close" as well. Closeness is measured in the number EFs to transmit, and thus in the number of (address, verifier) nibble pairs in which neither component differs between the two addresses.

To add fake edges from a given "base" node, the attacker combines the base node ID fragments with the fake node fragments by exclusive-or-ing the appropriate address and verifier fragments. This is the case discussed below.

A slight modification to the near-collision generation process can be used to create a single edge each from a family of multiple potential base nodes, which would be of similar usefulness and computational complexity, but is not discussed in this paper.[4]

Useful nodes to add fake edges to can be determined ahead of time by the attacker using mapping tools such as Skitter [Ski] or simple traceroutes.

[4]Another possibility to add multiple fake edges includes generating a single new node, but multiple edges from base nodes leading to that single node. This leaves much fewer degrees of freedom for the attacker, as the number of previous nodes will be very small, thus increasing the resources required to find such a near-birthday collision. It also is only of doubtful usefulness to the attacker, as this single node will not significantly further his aim of creating endless confusion.

## IV. GOSSIB ANALYSIS

In this section, we provide analytical results on the frequency of such near-birthday collisions. The next section will compare these with actual collision results from our simulation.

Recall the traditional birthday paradox: For $n$ randomly chosen samples $a$ out of a set of size $N$, the probability $\mathrm{B}(n, N)$ of a collision is

$$\mathrm{B}(n, N) = 1 - \left(1 - \frac{1}{N}\right)^{\frac{n(n-1)}{2}} \quad . \tag{1}$$

Also, for $n = \sqrt{N}$, the probability of a collision becomes $p \approx 1/2$.

For the analysis, we first need to establish the properties of address $a$ and the corresponding verifier $v$, which are grouped into $k$ chunks of $b$ bits each. Again, the $i$th chunk of $a$ or $v$ is indicated as $a_i$ or $v_i$, respectively, $i \in [0, k)$. Each $a$ is chosen uniquely out of a space of $2^{kb}$, and is associated with a $v$ which can be considered an iid (independent and identically distributed) random variable and is thus not unique.

It follows from the unique choice of $a$ that no $(a, v)$ tuple for two different addresses can cause a collision. But how many collisions exist in $c$ out of $k$ chunks?

### A. Pairs with $k-1$ Matching Chunks

For $c = k - 1$ and a given offset $i$ at which the difference can occur, $2^{cb}$ groups of $2^b$ members are created, i.e., $cb = (k-1)b$ bits of the address define the group ID, whereas the remaining $b$ bits distinguish between the members of the group. For each experiment, we thus fix the bits used for the group ID to a single location. The "birthdays" of these members are again randomly chosen out of a space $2^{cb}$, resulting in $I = 2^{cb}$ *independent* birthday problems with $n_{k-1} = 2^b$ "people," whose "birthdays" are chosen out of $N = I = 2^{cb}$ "days." The expected value for a single experiment of $k = 8$ and $b = 4$ is $e_{k-1} = N\mathrm{B}(n_{k-1}, N)$, the numerical results are shown in Table II.

When we release the fix on the bits used for group ID, we get $k$ experiments. As $a$ is not a random process, there can be no collision for $c = k$. Also, the systematic reshuffling that occurs when changing to a different bit set for the group ID, ensures that all $k-1$-matching pairs of $(a, v)$ tuples meet in exactly one experiment. Thus, we get $k$ independent experiments, resulting in

$$r_{k-1} = k \cdot e_{k-1} = kN\mathrm{B}(n_{k-1}, N) \tag{2}$$

(expected) near-collisions for $c = k - 1$. Each of these pairs enables the encoding of two edges in only $2k - c$ messages instead of $2k$, as illustrated in Figure 3.

### B. Pairs with $k-2$ Matching Chunks

For each experiment with $c = k - 1$, the above formulas again apply, with the exception of $n$, which needs to be written

TABLE II
NUMBER OF COLLISIONS IN $c$ CHUNKS

$k = 8$, $b = 4$. Analytical results for $c \leq 6$ rounded to nearest integer.
The "corrected total" discounts the measurements at higher $c$ values.

| | Fake | Messages | Analytical | | Simulation |
|---|---|---|---|---|---|
| $c$ | Edges | Per Edge | Per Experiment | Corrected Total | Total |
| 7 | 2 | 4.5 | 120 | 960 | 933 |
| 6 | 2 | 5 | 32,608 | 906,311 | 907,274 |
| 5 | 2 | 5.5 | | | >400,000,000 |
| 5 | 3 | $4.\overline{6}$ | | | 104,726,158 |

as $n_{k-2} = 2^{(k-c)b}$ in the general case. Each experiment thus results in $32608$ near-matches.

As the two nibbles to be fixed for each experiment can be chosen independently, the number of experiments reaches $k!/c!$. Each of the $r_{k-1}$ results from $k-1$ is counted in $c+1$ experiments, with $c = k-2$. Thus the total number of events with two mismatches is expected to be

$$r_{k-2} = \binom{k}{k-2} N\mathrm{B}(n_{k-2}, N) - (k-1)r_{k-1}. \qquad (3)$$

### C. Pairs with $c$ Matching Chunks

By induction, we obtain the following recursive formula for arbitrary $c$:

$$r_c = \binom{k}{c} N\mathrm{B}(n_c, N) - (c+1)r_{c+1} \quad . \qquad (4)$$

Unfortunately, it shows that the above formulas assume that only a single pair would match happen in a single group. For $c = k-1$ and $c = k-2$, the probability of multiple matches is negligible. For large $\mathrm{B}(n_c, N)$ (close to 1), as they occur for $c < k-2$, multiple matches happen frequently, as the groups become huge. This renders the simple birthday formula highly inaccurate for an estimate. Therefore, no analytical results are provided for $c = 5$ in Table II. As pairs with fewer matching chunks also quickly become ineffective, and thus uninteresting for attackers, a detailed analysis has not been included here, but the interested reader should refer to [FGT87, Cam98].

## V. GOSSIB SIMULATION

To complement the analysis and gain further insight, we also performed a brute force search to find close matches. The results are shown in the *Simulation* column of Table II. As the sheer number of near-collisions at $k-3$ was excessive, we limited ourselves to triple matches at this level, i.e., finding three $(a, v)$ tuples, where two of the pairs would be near-collisions at $k-3$, such that they could be encoded with a total $3k-2c$ messages. The third pair would not necessarily have to be a match at $k-3$, but could also be a better match ($c > k-3$).

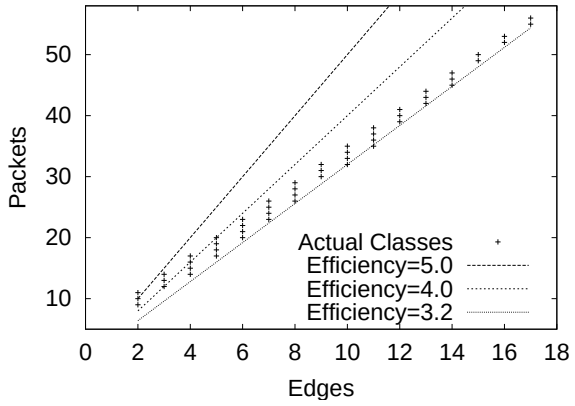



Fig. 4. Efficiency of various GOSSIBs

With the pairwise matches at $k-1$ and $k-2$ plus the triple matches at $k-3$ at hand, we evaluated whether these pairs and triplets were in fact the most effective ways of transmitting the information for fake edges. Every pair and triplet was considered a partial description of an equivalence class. Wherever these partial equivalence classes had overlaps, they were joined. To our surprise, the largest equivalence classes (GOSSIBs) were of size 17. Figure 4 shows the efficiency ratios (messages per fake edge) of the various equivalence classes. Multiple lines of constant efficiencies have been included for comparison.

The number of such equivalence classes is plotted by efficiency and size, as shown in Figures 5 and 6, respectively.

### A. GOSSIB Similarity Analysis

Packet loss is part of the normal Internet operation and will substantially increase under high load, such as during a DDoS attack. Also, overwriting of the IP ID field is non-deterministic. Therefore, GOSSIB should not rely on single critical packets. To get an impression on the relationship between nodes, Figure 7 shows some GOSSIB similarity graphs, where the distance (in terms of differing $(a, v)$ nibble pairs) has been drawn

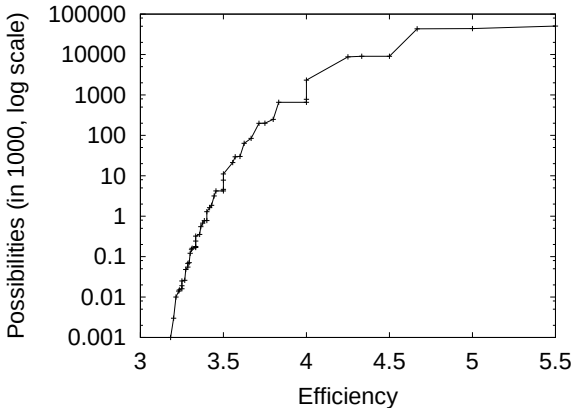



Fig. 5. Cumulative number of GOSSIBs by efficiency (log scale)

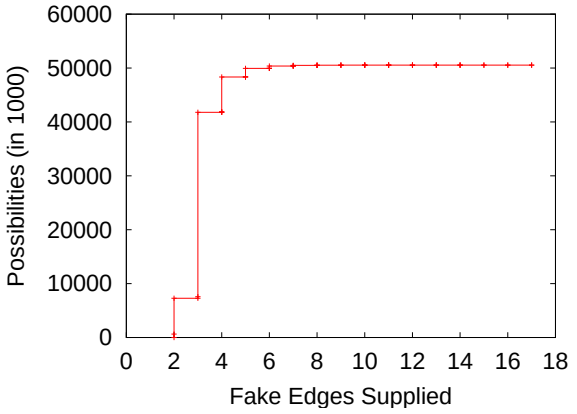



Fig. 6. Cumulative number of GOSSIBs by fake edges created

for two GOSSIBs. Please not that the edges and vertices in a similarity graph are not related to the attack reconstruction graph. Instead, the edges in the similarity graph indicate how many extra EFs need to be successfully transmitted to have an additional fake edge installed in the victim's reconstruction of the attack graph.

Figure 7(a) shows a quite typical representative, where the lines drawn indicate differences of three nibble pairs. If the EFs for the fake edge labeled "4" have been sent, sending the fake edges named "1," "2," or "6" could each also be installed in just three more messages. Figure 7(b) shows the same GOSSIB, but all the links with distances $\leq 5$ are shown, not just the minimum cost links (darker links indicate fewer messages).

Figure 7(c) shows the minimum graph of another GOSSIB. While this GOSSIB looks more densely connected, a look at all links with distances $\leq 5$ (Figure 7(d)) shows that this observation does not extend to non-minimum links. These similarity graphs are the exception, however.

Our studying of similarity graphs indicate that many graphs show connectivities close to the ones shown in Figures 7(a) and 7(b). This indicates that GOSSIB is highly resilient against a few lost packets; most of the fake edges are installed even when some EFs did not make it to the victim.

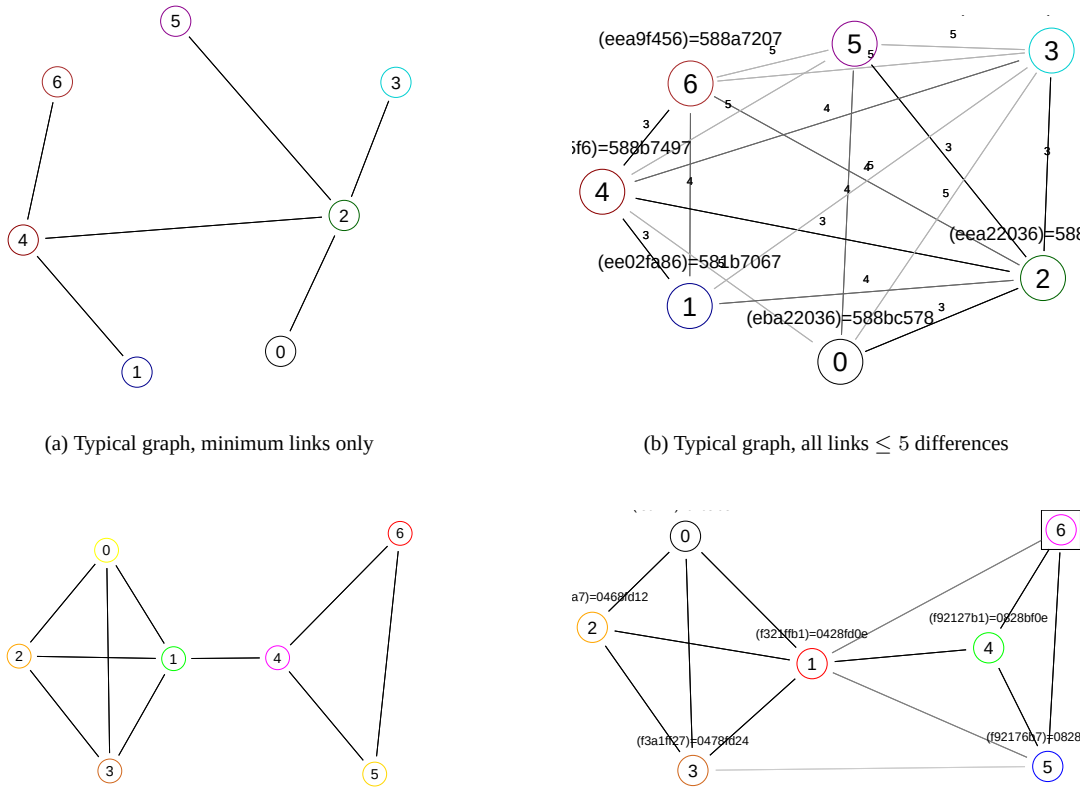
## VI. DDoS Countermeasures Overview

Besides making systems more secure to prevent their misuse as attack amplifiers, the most obvious countermeasure against hard-to-trace DDoS attacks certainly is ingress filtering based on source address. As this breaks some existing protocols and setups, such as some variants of Mobile IP, multi-homing, or asymmetric links (e.g., satellite downlink and modem uplink), Li et al. propose the Source Address Validity Enforcement Protocol, SAVE [LMW+01]. History seems to show that it is quite difficult to convince ISPs to install, configure, maintain, and support new protocols that cannot be sold as part of a service. As ingress filtering mostly protects users at other ISPs, the paying customers of an ISP implementing ingress filtering would not directly have a benefit, but instead might run into problems caused by the above-mentioned protocols. As this decreases customer happiness and increases customer service calls, ISPs thus seem unlikely to implement ingress filtering in the near future. The experiences presented in [Gib02] seem to support this.

The next step is victim pushback, where a site that believes to be under attack can send back messages installing filters at upstream routers [PL00, MBF+01, IB02]. First, due to the current lack of incentives for ISPs to provide such a service, we do not expect this to become widely deployed anytime soon. Second, before such a system can be put into place, utmost care needs to be taken to avoid attackers being able to install malicious filters to throw away legitimate traffic, thus creating another kind of DoS attack. Third, the authentication involved needs to be fast and reliable in order to prevent a DoS on the DoS prevention systems. The pushback systems described currently do not seem to adequately provide all these features.

With respect to traceback and identification of the attackers, Song et al. [SP01] improve on the Savage scheme by predetermining the network topology. This solution is limited to cases when the topology is static (at least locally to the potential victim) and the victim is immobile. The probing of the topology can be very taxing to the network, especially if a large proportion of Internet sites would start doing it. This map also allows for a more efficient encoding of edges and thus resulting in fewer chunks to reconstruct paths and in greatly improving the efficiency and accuracy of the protocol. However, given its high pre-attack overhead and need for continuous topology updates, we believe it to be infeasible for large-scale deployment.

Dean et al. [DFS01] provide another avenue to improve CEFS. Instead of using a hash function as a verifier, the routers algebraically encode the path or edge information iteratively using Horner's rule. The resulting $(x, y)$ coordinate tuples allow the reconstruction of the contributing polynomial coefficients




(a) Typical graph, minimum links only

(b) Typical graph, all links $\leq 5$ differences

(c) Dense graph, minimum links only

(d) Dense graph, all links $\leq 5$ differences

*(Node labels of the form $(a) = v$ indicate the $(a, v)$ tuples.)*

Fig. 7. GOSSIB similarity graphs

which encode the complete path. We believe that this scheme is also susceptible to a GOSSIB attack.

A different approach for traceback is shown by Snoeren et al. [SPS+01]: They propose storing a hash of each packet along with information about where it arrived from in a memory-efficient fashion. Given ubiquitous deployment of such a service, a network node can immediately ask for a traceback of an individual packet it just received. We expect this system to be the most useful of the traceback class, given complete (or at least very dense) deployment. On the downside, it is incompatible with the probabilistic traceback systems, as it requires the IP ID header field to pass through the network unmodified.

The Internet Engineering Task Force (IETF) also has a working group dedicated to establishing a standard traceback mechanism. The working group proposed that each router would periodically (every few hundred or thousand packets) select a packet and "append" authenticated traceback information to this packet. This information would convey that the packet was seen by this router. "Appending" would not be done by modifying the packet, but by creating a second packet tailgating the original packet. The working group has been largely dead since about a year, and its Internet-Drafts have all expired since, so the approach seems to have been abandoned.

Mankins et al. [MKB+01] instead try to discourage DoS attacks by charging for traffic based on the availability of resources. While it should drastically reduce attacks originating at the attacker's premises, we doubt that it will have the desired effect when the attacker uses compromised systems or weak protocol implementations as attack amplifiers. On the contrary, the owners of the compromised systems will incur financial charges for "their" attack traffic, even though it might be argued that this would be an incentive to end users to drastically improve their systems' security.

An early warning shortly before the actual DDoS attack strikes can be detected by analyzing traffic characteristics and trends [MBF+01, CLQ+01]. This could improve the victim's situation slightly, unless the start of the attack is highly synchronized [Gib02], but can neither prevent the DDoS or identify the actual sources. Mechanisms such as the XenoService [YEA00] may use this notification to start replicating the service to other machines while there is still bandwidth available. As this mechanism is scheduled to incur cost to the service owner to prevent

abuse, this can also be used to incur financial charges on the victim.

## VII. Conclusions

In our opinion, thy by far most important remedy is to close security holes that allow attackers to take over otherwise innocent machines and convert them into their compliant accomplices. As this is unlikely to happen fully, additional measures have to be taken in the network. Also, ingress filtering should be applied to prevent untraceable large-scale attacks. When ingress filtering is impossible, a traceback approach akin to the (now mostly abandoned) IETF approach or Snoeren et al. [SPS+01] should be deployed. Only if none of this is possible, should PPM be chosen. To prevent GOSSIB attacks by end users, the initial router in a customer ISP should replace all IP ID fields eligible for CEFS processing with random (from the customer's point of view). Still, all PPM traceback approaches are expected to continue to showing their inherent state explosion problem.

We believe that the GOSSIB attack does not exploit problems that are unique to CEFS, but rather they are inherent in any mechanism that distributes its data among too many packets and only uses limited security, as shown by the heavily truncated hash function. Therefore, an adaptation of the GOSSIB attack is expected to succeed against other PPM models including fragmentation, such as [DFS01]. We seriously doubt that traceback can be done efficiently unless extra trusted header fields (e.g., as part of MPLS) are added, all routers are equipped with hash-based traceback capability, every host connected to the Internet is secured, or ingress filtering is ubiquitously deployed.

## Acknowledgments

I thank Thomas Angst, Daniel Bauer, Germano Caronni, Wei Deng, Ulrich Fiedler, and Ramaprabhu Janakiraman for insightful technical discussions and providing computing resources. The comments of Charlotte Bolliger and the anonymous reviewers helped improving the quality of the paper.

## References

[Cam98] Michael Brett Camarri. *Asymptotics for Repeat Times in Random Sampling*. PhD thesis, University of California, Berkeley, May 1998.

[CLQ+01] João B. D. Cabrera, Lundy Lewis, Xinzhou Qin, Wenke Lee, Ravi K. Prasanth, B. Ravichandran, and Raman K. Mehra. Proactive detection of distributed denial of service attacks using mib traffic variables - a feasibility study. In *Proceedings of International Symposium on Integrated Network Management*, 2001.

[DFS01] Drew Dean, Matt Franklin, and Adam Stubblefield. An algebraic approach to IP traceback. In *Proceedings of the Network and Distributed System Security Symposium*, February 2001.

[Fel66] William Feller. *An Introduction to Probability Theory and Its Applications*, volume 1. Wiley, New York, NY, USA, 2nd edition, 1966.

[FGT87] Philippe Flajolet, Danièle Gardy, and Loÿs Thimonier. Birthday paradox, coupon collectors, caching algorithms and self-organizing search. Technical Report 720, INRIA, Rocquencourt, France, August 1987.

[Gib02] Steve Gibson. The strange tale of the denial of service attacks against grc.com. `http://grc.com/dos/grcdos.htm`, 2002.

[IB02] John Ioannidis and Steven M. Bellovin. Implementing pushback: Router-based defense against DDoS attacks. In *Proceedings of Network and Distributed System Security Symposium*, Reston, VA, USA, February 2002. The Internet Society.

[LMW+01] Jun Li, Jelena Mirkovic, Mengqiu Wang, Peter Reiher, and Lixia Zhang. SAVE: Source address validity enforcement protocol. Technical Report 010004, University of California, Los Angeles, 2001.

[LRST00] Felix Lau, Stuart H. Rubin, Michael H. Smith, and Ljiljana Trajovic. Distributed denial of service attacks. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 2275–2280, Nashville, TN, USA, October 2000.

[Mar00] Brian Martin. Have script, will destroy (Lessons in DoS). `http://www.attrition.org/~jericho/works/security/dos.html`, 2000.

[MBF+01] Ratul Mahajan, Steven M. Bellovin, Sally Floyd, John Ioannidis, Vern Paxson, and Scott Shenker. Controlling high bandwidth aggregates in the network. Technical report, AT&T Center for Internet Research at ICSI, July 2001.

[MKB+01] David Mankins, Rajesh Krishnan, Ceilyn Boyd, John Zaho, and Michael Frentz. Mitigating distributed denial of service attacks with dynamic resource pricing. In *Proceedings of Annual Computer Security Applications Conference (ACSAC 2001)*, 2001.

[MVS01] David Moore, Geoffrey Voelker, and Stefan Savage. Inferring internet denial of service activity. In *Proceedings of the 2001 USENIX Security Symposium*, Washington, DC, USA, August 2001.

[NBBB98] Kathleen Nichols, Steven Blake, Fred Baker, and David L. Black. Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers. RFC 2474, Internet Engineering Task Force, December 1998.

[PL00] Kihong Park and Heejo Lee. A proactive approach to distributed dos attack prevention using route-based distributed filtering. Technical Report CSD-00-017, Department of Computer Sciences, Purdue University, 2000.

[PPS00] Rich Pethia, Alan Paller, and Gene Spafford. Consensus roadmap for defeating distributed denial of service attacks. `http://www.sans.org/ddos_roadmap.htm`, 2000.

[RFB01] K. K. Ramakrishnan, Sally Floyd, and David L. Black. The addition of explicit congestion notification (ECN) to IP. RFC 3168, Internet Engineering Task Force, September 2001.

[Ric02] Tim Richardson. Cloud nine blown away, blames hack attack. `http://www.theregister.co.uk/content/6/23770.html`, January 2002.

[Riv92] Ronald L. Rivest. The MD5 message digest algorithm. Internet RFC 1321, April 1992.

[Ski] Skitter. `http://www.caida.org/tools/measurement/skitter/`.

[SP01] Dawn X. Song and Adrian Perrig. Advanced and authenticated marking schemes for IP traceback. In *Proceedings IEEE INFOCOM*, 2001.

[SPS+01] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Stephen T. Kent, and W. Timothy Strayer. Hash-based ip traceback. In *Proceedings of SIGCOMM 2001*, August 2001.

[SWKA00] Stefan Savage, David Wetherall, Anna R. Karlin, and Tom Anderson. Practical network support for IP traceback. In *Proceedings of ACM SIGCOMM*, pages 295–306, 2000.

[SWKA01] Stefan Savage, David Wetherall, Anna R. Karlin, and Tom Anderson. Network support for IP traceback. *ACM/IEEE Transactions on Networking*, 9(3):226–237, June 2001.

[YEA00] Jianxin Yan, Stephen Early, and Ross Anderson. The xenoservice - a distributed defeat for distributed denial of service. In *Proceedings of CERT Information Survivability Workshop 2000*, October 2000.