Tradeoffs in Probabilistic Packet Marking for IP Traceback.

Micah Adler*
Department of Computer Science
University of Massachusetts
Amherst, MA 01003-4610

February 20, 2002

Abstract

There has been considerable recent interest in probabilistic packet marking schemes for the problem of tracing a sequence of network packets back to an anonymous source. An important consideration for such schemes is the number of packet header bits that need to be allocated to the marking protocol. Let b denote this value. All previous schemes belong to a class of protocols for which b must be at least $\log n$, where n is the number of bits used to represent the path of the packets. In this paper, we introduce a new marking technique for tracing a sequence of packets sent along the same path. This new technique is effective even when $b = 1$. In other words, the sequence of packets can be traced back to their source using only a single bit in the packet header. With this scheme, the number of packets required to reconstruct the path is $O(2^{2n})$, but we also show that $\Omega(2^n)$ packets are required for any protocol where $b = 1$. We also study the tradeoff between b and the number of packets required. We provide a protocol and a lower bound that together demonstrate that for the optimal protocol, the number of packets required (roughly) increases exponentially with n, but decreases doubly exponentially with b. The protocol we introduce is simple enough to be useful in practice. We also study the case where the packets are sent along k different paths. For this case, we demonstrate that any protocol must use at least $\log(2k-1)$ header bits. We also provide a protocol that requires $\lceil \log(2k+1) \rceil$ header bits in some restricted scenarios. This protocol introduces a new coding technique that may be of independent interest.

1 Introduction

In recent years, the Internet has seen an alarming increase in what are known as denial-of-service attacks. Such an attack consists of a malicious party sending enormous volumes of traffic to a remote host or a network, thereby denying legitimate users access to this shared resource. Unfortunately, such attacks are easy to perform, and in fact there are well known techniques for mounting attacks against a single shared resource that are coordinated to occur simultaneously from a large number of distributed hosts [4]. To make matters worse, in the current and foreseeable routing architectures of the Internet, a host transmitting packets can use a forged source address for those packets. This means that there is little or no accountability for the source of these attacks and the process of halting an attack in progress is both slow and requires significant resources. Thus, one of the most

*This work supported in part by the National Science Foundation under NSF grant EIA0080119.

important tools needed to fight denial-of-service attacks is an automated technique for tracing a stream of packets back to its source. This is known as the *IP traceback* problem.

A number of different approaches to the IP traceback problem have been suggested. In this paper, we study one of the most promising, which is called *probabilistic packet marking*, or PPM (for advantages of PPM over other techniques, see [3] and [15].) PPM was suggested in [3], with the first actual schemes for PPM described in [15] and [6]. In [15], Savage *et. al.* propose the following clever approach to the IP traceback problem: some fixed number of bits in the packet header are allocated to IP traceback, and are used to store a router ID and a hop count. Every router that forwards a packet, independently with some probability p, writes its unique ID to those bits, and sets the hop count to 0. With probability $1-p$, the router ID is left unchanged, and the hop count is incremented. Now, say an *attacker* is performing a denial-of-service attack on a *victim* by sending a stream of packets along a path of length ℓ. If $p = \Theta(1/\ell)$, then after the victim has received $O(\ell \log \ell)$ packets, with high probability this scheme provides the victim with the entire path back to the attacker.

The elegant PPM scheme of [15] has produced a flurry of activity in the networking community, and in the relatively short time since [15] and [6] appeared, a number of variations on PPM have been introduced to improve and further analyze the [15] technique [5, 11, 17] (see also [9] and [12]). One important concern in this literature is reducing the number of header bits required for PPM. In [15], they further refine their scheme so that they require 16 header bits, and can reconstruct the entire path with high probability after a few thousand packets have been received. This has subsequently been improved to 13, achieved by a scheme in [5], which is the minimum required bits achieved prior to the work described in this paper.

There has also been significant effort to develop PPM techniques that are effective when the packets travel to the victim of the attack along multiple paths [5, 17]. This is a concern both since the attacker may send packets from a number of distributed sources simultaneously, and also since packets from a single source may travel to the victim using a number of different paths.

Despite the number of papers in this area, a rigorous theoretical analysis of PPM has been lacking. There has been no real understanding of how the number of header bits and the number of packets required grow as the size of the underlying network increases. Also, there has been no understanding of the interplay and inherent tradeoffs between the number of header bits used, the number of paths of attack, and the number of packets required to reconstruct (with high probability) the path or paths used by the attacker. In addition to the significant practical importance of PPM, it turns out that developing a thorough understanding of these questions is an interesting and challenging theoretical problem.

Furthermore, all previous techniques for PPM have used techniques that belong to a restrictive class of protocols. In particular, all previous PPM protocols encode the path information in such a manner that the victim only uses the information of *what* packets it receives, and it can ignore the information of *how many* of each type of packet it receives. Let b be the number of header bits allocated to IP traceback, and let n be the number of bits required to represent a path of attack (the definition of n will be made precise in Section 2). For any protocol with this property, if $b < \log n$ then there is some attacker path that can only be correctly identified with probability less than $1/2$.[1] Thus, $\log n$ is a lower bound on b for this class of protocols.

[1]To see this, note that there are 2^{2^b} different sets of packets that can be received. Since there are 2^n possible n-bit strings, if the victim must determine each possible correct attacker path with probability at least $1/2$, then 2^{2^b} must

1.1 Summary of results

In this paper, we consider two different scenarios for the IP traceback problem: the important special case (studied in [11], [6], and [15]) where the attacker sends all of its packets along the same path, and the more general case where there are multiple paths of attack. For the case of a single path of attack, we introduce a new type of PPM technique that does not belong to the class of protocols described above, allowing for a more efficient encoding of the path description. This new technique allows any path to be revealed to the victim even when $b = 1$. In other words, this new scheme requires using only a single header bit, which is obviously the minimum possible. Unfortunately, this requires $\Theta((2 + \epsilon)^{2n})$ packets to be received by the victim, for any constant $\epsilon > 0$, and thus is only appropriate for small values of n. However, we also provide an information theoretic lower bound demonstrating that $\Omega(2^n)$ packets are necessary for any one-bit protocol where the victim is able to determine the correct path with probability greater than $1/2$.

The large number of packets required by one-bit protocols leads to the following question: how does the number of packets required decrease as b increases? In this paper we provide a good understanding of the optimal tradeoff between these two quantities. We demonstrate that the optimal number of packets that must be received for given values of n and b grows exponentially with n, but decreases *doubly* exponentially with b. Specifically, we provide a protocol that requires only $O(bn^2 2^b (2+\epsilon)^{4n/2^b})$ packets, for any constant $\epsilon > 0$, to reconstruct the path (w.h.p.), as well as an information theoretic lower bound showing that $\Omega(2^b 2^{n/2^b})$ packets are necessary for the victim to be able to determine the correct path with probability greater than $1/2$. The protocol that achieves the upper bound is simple (although its analysis is not simple), and the communication model is realistic, and thus we expect the protocols for the single path case to be quite effective in practice.

For the case of multiple paths of attack, we demonstrate that one-bit protocols are not possible, no matter how many packets the victim receives. In particular, let k represent the number of paths used by the attacker. We provide a lower bound demonstrating that any correct protocol must use at least $\log(2k - 1)$ header bits, regardless of the number of packets received by the victim. This lower bound reveals an inherent limitation of all existing PPM protocols that has not been addressed previously. We also provide a protocol demonstrating that for a restricted class of attacker strategies, $\lceil \log(2k+1) \rceil$ bits are sufficient. This protocol relies on a new encoding technique that is based on a novel use of Vandermonde matrices, looks promising in terms of leading to an upper bound for an unrestricted adversary, and may also be of independent interest.

2 The Models

We use slightly different models for the protocols and for the lower bounds, where the lower bound model is at least as powerful as the upper bound model. We first describe the model used for the protocols. We assume that from the perspective of the victim, the routing topology of the network consists of a tree rooted at the victim. Thus, any packet sent to the victim travels up this tree until it reaches the victim. At the start of the attack, the attacker chooses a set of nodes of the tree, and then for each packet, it determines which of these nodes sends that packet to the victim. We first examine the case where the attacker only chooses a single path; additional details on the model where there are multiple paths are provided in Section 4.

be at least $2^n/2$. Assuming that $n \geq 2$, the fact that b must be an integer implies that $b \geq \log n$.

We first introduce the protocols by making a number of simplifying assumptions about the network. We then demonstrate in Section 5 that our results can easily be extended to hold in scenarios where these assumptions are relaxed. In particular, we start by assuming that the tree is a complete binary tree of height n, with the victim forming an additional node connected to the root of the tree. We also assume that when a packet is sent to a node, that node is able to distinguish which child of the node the packet came from. Finally, we also assume that the victim has complete knowledge of the topology of the routing tree. Again, we want to emphasize that Section 5 describes how each of these assumptions can be removed. For example, our results apply to an arbitrary routing topology that is unknown to either the victim or the intermediate nodes[2].

The header of each packet contains b bits that are allocated to traceback information. No other bits of the packet can be utilized for IP traceback, and thus we assume that each packet consists of only these b bits. For each packet that is forwarded from the attacker to the victim, the attacker sets the initial value of these bits, and then each of the intermediate nodes is allowed to alter them, but no other communication occurs.

We also make the restriction that protocols do not require any state information at the intermediate nodes. Due to the memoryless nature of Internet routing, the lack of state information is an important requirement for PPM protocols: it is impractical for routers to store any information on individual flows. All of our protocols have the property that for each node, the set of b bits that the node forwards to its parent in the tree are only a function of the incoming b bits, which child of that node the packet arrives from, and random bits (that are not remembered). The victim, on the other hand, does have storage.

For a given placement of the attacker at a leaf of the tree, we shall refer to the node on the path from the root to the attacker at distance i from the victim as N_i (where the victim is N_0, and the attacker is N_{n+1}). Since we are assuming a binary tree, we can represent the path as a binary string $B = B_1B_2 \dots B_n$, where $B_i = 0$ if the path goes to the left child of N_i, and $B_i = 1$ otherwise. Note that when determining the outgoing bits for any packet, node N_i has access to one bit of the string B: the bit B_i. It does not require state information to use this bit when setting the header bits of a packet it forwards, since every incoming packet reveals the value of this bit.

The objective is for the intermediate nodes to inform the victim of the string B. In the case that the attacker chooses a leaf node, the string B uniquely identifies the identity of the attacker, thereby solving the IP traceback problem. On the other hand, if the attacker chooses a node that is not a leaf of the tree, it may be able to set the initial bits of the packets in such a manner that it exactly simulates what would occur if one of the children of the chosen node were sending the packets. In other words, the path would look like it extends beyond its actual source. Various ways of dealing with this have been suggested, including using cryptographic techniques [17], or topological knowledge [15]. For simplicity, we assume here that it is sufficient to determine a path that contains the correct path from the victim to the attacker as a prefix.

For the lower bound, we assume a stronger model (i.e., a model where the problem is at least as easy to solve as in the model for the protocols). For the lower bound model, we assume a system consisting of only two parties, called the *Victim* and the *Network*. The Network has an n-bit string to send to the Victim. No communication occurs from the Victim to the Network. The Network is allowed to send b-bit packets to the Victim, but it is stateless: for each packet it sends, it has no memory of the previous packets that it has sent. This lower bound model actually captures the

[2] We point out that under this assumption, the resulting value of n can be larger.

difficulty of sending information from a memoryless node using packets consisting of a bounded number of bits. This seems like a fundamental problem, and may be of interest beyond the context of the IP traceback problem.

It is easy to show that any protocol for the upper bound model can be simulated in the lower bound model, and thus lower bounds for the lower bound model also apply to the upper bound model. Furthermore, it seems likely that the lower bound model is strictly more powerful than the upper bound model, since the lower bound model has the advantage that a single party knows the entire n-bit string, instead of that string being distributed across n nodes. Also, in the upper bound model, the protocol must deal with a malicious attacker that sets the bits of the initial packet the n nodes receive.

3 Single Path of Attack

We start by providing a protocol for the case where $b = 1$. The basic idea behind this protocol is to encode the string B into p, the probability that the bit received by the victim is a 1. For example, consider the encoding where $p = \sum_{i=1}^{n} B_i(\frac{1}{2})^i$. With such an encoding, if the victim receives enough packets to determine the bias of p (with the required confidence) within an additive term of $(\frac{1}{2})^{n+2}$, then it is able to determine all n bits of the binary string B. All of our protocols use variations on this kind of encoding to transmit information to the victim, and so we start by proving a general lemma concerning such encodings.

Lemma 1 *Consider any set of bits $B_1 \ldots B_\ell$, and any protocol where the victim is able to determine real numbers p, σ, and $c_1 \ldots c_\ell$, that satisfy the following conditions:*

1. $|p - \sum_{j=1}^{\ell} c_j B_j| \leq \sigma$.
2. *For all i, $1 \leq i \leq \ell - 1$, $c_i > 2\sigma + \sum_{j=i+1}^{\ell} c_j$.*
3. $c_\ell > 2\sigma$.

These values allow the victim to efficiently and uniquely determine the bits $B_1 \ldots B_\ell$.

Proof: We demonstrate that for any i, $1 \leq i \leq \ell$, if the victim knows $B_1 \ldots B_{i-1}$ (or none of the bits in the case that $i = 1$), then it can determine the value of B_i. To do so, let $p' = p - \sum_{j=1}^{i-1} c_j B_j$. If $p' \geq c_i - \sigma$, then it must be the case that $B_i = 1$. On the other hand, if $p' < c_i - \sigma$, it must be the case that $B_i = 0$. Thus, the values of the B_i can be computed in a greedy fashion, starting with B_1 and working one bit at a time towards B_ℓ. ■

Let **DECODE**$(p, \sigma, c_1, \ldots, c_\ell)$ be the result of performing this decoding using the real numbers p, σ, and $c_1, \ldots, c_\ell$. It turns out that the encoding (described above) where $c_i = (\frac{1}{2})^i$ can be achieved if we assume that the attacker always sets the initial bit to 0. However, if the attacker is allowed to set the initial value arbitrarily, then the resulting protocol does not quite uniquely specify the encoded string.[3] Thus, we use the encoding where $c_i = \frac{r^{i-1}}{2}$, for $r = 1/2 - \epsilon$, for any ϵ such that $0 < \epsilon < 1/2$. To achieve such an encoding using only one-bit packets, we describe what any node

[3] Specifically, the attacker can make lexicographically adjacent strings encode to the same probability.

N_i does on the receipt of a packet from its neighbor. Note that there are only four possible inputs for the node N_i, differentiated by the bit B_i and the bit that N_i receives from node N_{i+1}. The following table describes the probability that N_i forwards a 1 to node N_{i-1} on the four possible inputs:

	Incoming bit	
B_i	0	1
0	0	$1/2-\epsilon$
1	$1/2$	$1-\epsilon$

Otherwise, in all four cases, N_i forwards a 0. The victim uses the following decoding process: Obtain $F = \frac{6\ln(2/\Delta)}{\epsilon^2 r^{2n}}$ packets. Let x be the number of 1s in this set of packets, and let $p = x/F - r^n/2$. Let $\sigma = r^n/2 + \epsilon r^n$. Set the bits $B_1 \ldots B_n$ according to the process $\mathbf{DECODE}(p, \sigma, \frac{1}{2}, \frac{r}{2}, \frac{r^2}{2}, \ldots, \frac{r^{n-1}}{2})$. We call this protocol **Single-Bit**.

Theorem 1 *With probability* $1-\Delta$*, protocol* **Single-Bit** *allows the victim to determine the correct values of* $B_1 \ldots B_n$.

Proof: For $t \in \{0,1\}$, let p_i^t be the probability that the bit received by node N_i is a 1, given that the attacker sets the initial bit to t.

Claim 1 *For* $t \in \{0,1\}$*, it holds that* $p_0^t = t \cdot r^n + \sum_{i=1}^n B_i \frac{r^{i-1}}{2}$.

Proof: We see that for $i \leq n$, if $B_i = 0$, then $p_{i-1}^t = rp_i^t$. If $B_i = 1$, then $p_{i-1}^t = (1-\epsilon)p_i^t + \frac{1}{2}(1-p_i^t) = rp_i^t + \frac{1}{2}$. The claim now follows by induction. ∎

From this claim, we see by a Chernoff bound that $\Pr[|p - p_0^0| > r^n/2 + \epsilon r^n] \leq \Delta$. When $|p - p_0^0| \leq r^n/2 + \epsilon r^n$, condition 1 of Lemma 1 is satisfied. To see that condition 3 is always satisfied, note that $r^n/2 + \epsilon r^n < r^{n-1}/4$ is equivalent to requiring that $r(\frac{1}{2} + \epsilon) < 1/4$, which follows from the fact that $r = 1/2 - \epsilon$. To see that condition 2 is also always satisfied, note that for all i, $1 \leq i \leq \ell - 1$, $c_i - \sum_{j=i+1}^{\ell} c_i > c_n$. Thus, by Lemma 1, the victim is able to determine the entire string with probability $1 - \Delta$. ∎

Note that this algorithm requires a number of packets that is exponential in n. We show in Section 3.1 that for the case where $b = 1$, such a dependence is necessary. Since this makes the protocol impractical for all but small values of n, we must use a larger value of b. We next describe how to extend the one bit scheme to the case where $b > 1$. In fact, it is possible to obtain a doubly exponential decrease in the number of packets required as b increases. To do so, we partition the nodes of the path into $d = 2^{b-1}$ sets, numbered 0 to $d-1$, where node N_i is in the set $i \bmod d$. Each of these sets performs the one-bit protocol (almost) independently, thereby decreasing the effective length of the path encoded using the one-bit protocol by a factor of d. The doubly exponential improvement comes from the fact that the number of packets required grows exponentially with n, and we decrease the effective value of n by a factor that is exponential in b.

To see how to develop this idea into a valid protocol, we first consider an idealized scenario, where for every packet, the attacker sets the initial b bits by choosing a random sample from the uniform distribution over all 2^b possible settings. We then describe how to convert this into a protocol where the attacker can set the b initial bits arbitrarily. Denote the b bit positions in the header as $h_0 \dots h_{b-1}$. For the idealized setting, in each packet, the nodes in one cell of the partition perform the one bit protocol using the bit h_0. The remainder of the bits are used as a counter to specify which cell of the partition participates in the one bit protocol. In particular, for a packet P, let I_i^P be the integer corresponding to the binary representation of the bits $h_1 \dots h_{b-1}$ that are received at node N_i. Thus, we say node i sets $I_{i-1}^P = j$ to mean that on packet P, the bits $h_1 \dots h_{b-1}$ sent from node N_i to node N_{i-1} are set to the binary representation of j.

Each node N_i performs the following protocol for each packet P:

- If $I_i^P = 0$, then perform the one bit protocol using the received bit h_0. Forward h_0 as the resulting bit of the one bit protocol, and set $I_{i-1}^P = 1$.
- Otherwise, forward h_0 unchanged, and set $I_{i-1}^P = I_i^P + 1 \bmod d$.

It is not hard to see that for each possible setting of the initial bits by the attacker, only one cell of the partition participates in the one bit protocol. Let t_k be the number of packets received that are involved in the one bit protocol for cell k of the partition. If the total number of packets received is $\Theta\left(d(1/\epsilon r^{n/d})^2 \log(d/\Delta)\right)$, then as long as $n \gg d$, $t_k \geq \Theta\left((1/\epsilon r^{n/d})^2 \log(d/\Delta)\right)$ with probability at least $1 - \Delta/d$. This result follows from a Chernoff bound; we omit the details since we give a full treatment for the analysis of the full version of the protocol below. For this value of t_k, the effect is the same as performing the one bit protocol on a path of length n/d. Thus, the victim is able to reconstruct all of the bits in cell k of the partition with probability $1 - \Delta/d$, and hence is able to reconstruct all of the bits with probability $1 - \Delta$.

To make this algorithm work for an attacker that is allowed to set the initial bits arbitrarily, we need to modify the protocol slightly. Note that otherwise the attacker could, for example, set the initial bits to the same value for every packet, which would only inform the victim of the bits in one cell of the partition. The change to the protocol is simple: with a probability ρ (which can be any probability such that $\rho \neq 0$ and $\rho \neq 1$, but ideally $\rho = 1/n$), each node N_i performs what is called a *reset*: it ignores the incoming bits completely, and sets $I_{i-1}^P = 1$. Bit h_0 is forwarded as B_i with probability $\frac{1}{2}$ and as 0 otherwise. This has the effect of resetting the counter with some probability, thereby allowing the bits from every cell of the partition to be sent to the victim.

With this more powerful attacker, we also need to develop a more complicated decoding procedure. We start with some intuition for this decoding algorithm by developing an expression for the probability that a packet has h_0 set to 1 when it arrives at N_0. Let v_k^n be the probability that a packet P is reset by some node between N_n and N_1, inclusive, and $I_0^P = k$. If $z(n,k)$ is the number of integers i, $1 \leq i \leq n$, such that $i \bmod d = k$, we see that $v_k^n = \sum_{j=1}^{z(n,k)} \rho(1-\rho)^{(j-1)d+k-1}$. Also, let α_j^k be the probability that a packet that arrives at N_0 is reset last by some node between N_n and $N_{k+(j-1)d}$, given that it is reset by some node between N_n and N_1, and that $I_0^P = k$. From Bayes rule, we obtain $\alpha_j^k = \frac{1}{v_k^n} \sum_{t=j}^{z(n,k)} \rho(1-\rho)^{(t-1)d+k-1}$.

Let P_k be the set of packets P such that $I_0^P = k$. For $0 \leq k \leq d-1$, let q_k^n be the fraction of packets in P_k such that no node between N_1 and N_n (inclusive) performs a reset on the packet. Note that q_k^n is not a value readily available to the victim; an important portion of the decoding algorithm is computing for each k a value $\bar{q}_k^n$ that serves as an estimate for q_k^n. Consider a packet chosen uniformly at random from the set of packets in P_k for which the attacker sets $h_0 = t$, for $t \in \{0,1\}$. The probability that the packet has h_0 set to 1 when it arrives at N_0 is

$$p_k^t = t \cdot q_k^n r^{z(n,k)} + \sum_{j=1}^{z(n,k)} B_{k+(j-1)d}(q_k^n + (1-q_k^n)\alpha_j^k)\frac{r^{j-1}}{2}.$$

Thus, if we knew exactly the values q_k^n, the decoding process would not be very different from the single bit protocol. However, without at least a fairly accurate estimate for q_k^n, such a decoding process would not be able to determine the string B uniquely. We next describe a decoding algorithm that computes such an estimate. We here describe this algorithm for the case where the value of n is known. However, the same process applies for any value of $\ell \leq n$ determined by the victim: the victim can decode any prefix of the path up to the attacker. We here also describe the easier case of the decoding process where $\rho \leq \frac{1}{n}$. We also want to emphasize that for this version of the paper, we have made no attempt to optimize the constants of the algorithm. The algorithm works as follows:

- N_0 waits until it has received $F = \left(\left(\frac{48e^2}{\rho r^{\lfloor n/d \rfloor}}\right)^2 \left(\frac{4ed}{n\rho}\right) \ln(4d/\Delta)\right)$ packets.
- For $0 \leq k \leq d-1$, $t \in \{0,1\}$, let f_k^t be the total number of packets in P_k for which the value of the bit h_0 received at N_0 is t. Note that $F = \sum_{k=0}^{d-1}(f_k^1 + f_k^0)$.
- Let $\bar{q}_k^n = \frac{f_k^1 + f_k^0 - v_k^n \cdot F}{f_k^1 + f_k^0}$.
- For $k = 0$ to $d-1$:
  - For $j = 1$ to $z(n,k)$,
    - Let $c_j^k = (\bar{q}_k^n + (1-\bar{q}_k^n)\alpha_j^k)\frac{r^{j-1}}{2}$.
  - Let $\sigma_k = (\bar{q}_k^n + (1-\bar{q}_k^n)\alpha_{z(n,k)}^k)\frac{r^{z(n,k)}}{2}$.
  - Let $p_k = \frac{f_k^1}{f_k^1 + f_k^0} - \frac{\bar{q}_k^n r^{z(n,k)}}{2}$.
  - Set the bits B_i, for $i = k + 1 + (j-1)d$, $1 \leq j \leq z(n,k)$, according to the process **DECODE**$(p_k, \sigma_k, c_1^k, \ldots, c_{z(n,k)}^k)$.

We call the resulting combination of the encoding algorithm at the nodes and the decoding algorithm at the victim the protocol **Multi-bit**. Note that in the case that $\rho = \Theta(1/n)$, the number of packets required by **Multi-bit** is $O\left(\frac{2^b n^2}{r^{4n/2^b}} \ln(2^b/\Delta)\right)$. Also note that protocol **Multi-bit** is reasonably efficient in terms of memory requirements at the victim: $O(d \log F)$ bits of memory are sufficient. Finally, note that the interesting case of the algorithm is when $2 \leq b \leq \lceil \log n \rceil$, since **Single-Bit** handles the case when $b = 1$, and when $b > \lceil \log n \rceil$, then techniques such as those used in [15] are sufficient.

Theorem 2 *If* $2 \leq b \leq \lceil \log n \rceil$ *and* $\Delta \leq 1/8$, *then with probability at least* $1 - \Delta$, *protocol* **Multi-bit** *allows the victim to determine the correct values of* B_i, $\forall i$, $1 \leq i \leq n$.

Proof: We here show that each of the d decoding processes produces the correct answer with probability at least $1 - \Delta/d$, from which the theorem follows directly from a union bound. For each call to the decode process, we demonstrate that the conditions of Lemma 1 are satisfied. For condition 2, we must show that for $1 \leq j \leq z(n,k)-1$, $c_j^k > 2\sigma_k + \sum_{t=j+1}^{z(n,k)} c_t^k$. Note that since the expression $(\bar{q}_k^n + (1-\bar{q}_k^n)\alpha_j^k)$ is monotonically nonincreasing as j increases, and $r < 1/2$, we see that $c_j^k - \sum_{t=j+1}^{z(n,k)} c_t^k > r^{z(n,k)-1}/2$. Since $\bar{q}_k^n + (1-\bar{q}_k^n)\alpha_{z(n,k)}^k \leq 1$, we have that $\sigma_k \leq r^{z(n,k)-1}/4$, implying condition 2. Also note that condition 1, i.e., that $c_{z(n,k)}^k > 2\sigma_k$, follows directly from the definitions of $c_{z(n,k)}^k$ and σ_k, and the fact that $r < 1/2$.

Thus, we only have left to prove that condition 1 holds with probability at least $1 - \Delta/d$, or that

$$\Pr\left[\left|p_k - \sum_{j=1}^{z(n,k)} c_j^k B_{(j-1)d+k}\right| \leq \sigma_k\right] \geq 1 - \Delta/d.$$

If it were the case that our estimates of q_k^n were exact, and the fraction of packets for which $h_0 = 1$ at N_0 were exactly the expectation, then condition 1 would follow easily. Of course, the probability of these random variables being exactly their expectation is too small for our purposes, but we can demonstrate that, with sufficiently high probability, they do not deviate far from their expectation.

To do so, we use two versions of the Chernoff bound [10]. In particular, if $X_1 \ldots X_t$ are i.i.d. random variables, such that $\Pr[X_i = 1] = p$, and $\Pr[X_i = 0] = 1 - p$, then for any δ such that $0 \leq \delta \leq 1$,

$$\Pr\left[\sum_{i=1}^{t} X_i \geq (1+\delta)tp\right] \leq e^{-\delta^2 tp/3},$$

and

$$\Pr\left[\sum_{i=1}^{t} X_i \leq (1-\delta)tp\right] \leq e^{-\delta^2 tp/2}.$$

We first use these bounds to demonstrate that it is likely that P_k is large enough to provide good estimates on the quantities of interest. In particular, we show the following:

Claim 2 *Let* $\mu = \left(\frac{48e^2}{\rho r^{z(n,k)}}\right)^2 \ln(4d/\Delta)$. *It holds that* $\Pr\left[f_k^0 + f_k^1 < \mu\right] \leq \frac{\Delta}{4d}$.

Proof: Regardless of what the attacker does, for any packet P, $\Pr[I_0^P = k] \geq v_k^n$. Thus, we can define a set of i.i.d. indicator variables $X_1 \ldots X_F$ such that $X_j = 1$ if packet j is in P_k. We see that $f_k^0 + f_k^1 = \sum_{j=1}^{F} X_j$, and $\Pr[X_j = 1] \geq v_k^n$. Since the probability that $f_k^0 + f_k^1$ is too small is maximized when $\Pr[X_j = 1] = v_k^n$, we can assume that this is the case. From the definition of v_k^n, we see that $v_k^n \geq \frac{z(n,k)\rho}{e}$, which by the assumption that $b \leq \lceil \log n \rceil$ implies that $v_k^n \geq \frac{n\rho}{2ed}$. The claim now follows from the second Chernoff bound above, using $\delta = \frac{1}{2}$. ■

We next demonstrate that our estimate of q_k^n is quite accurate:

Claim 3 *Given that* $f_k^0 + f_k^1 \geq \mu$, *it holds that* $\Pr\left[|\bar{q}_k^n - q_k^n| > \frac{\rho r^{z(n,k)}}{12e^2}\right] \leq \frac{\Delta}{4d}$.

Proof: Let $\bar{v}_k^n$ be the actual fraction of the F packets P which are reset by some node and $I_0^P = k$. Since $q_k^n = \frac{f_k^1 + f_k^0 - \bar{v}_k^n \cdot F}{f_k^1 + f_k^0}$, we see that $|\bar{q}_k^n - q_k^n| = \frac{\bar{v}_k^n \cdot F - v_k^n \cdot F}{f_k^1 + f_k^0}$. If we do not condition on $f_k^0 + f_k^1 \geq \mu$, then the fact that $\Pr\left[|\bar{v}_k^n \cdot F - v_k^n \cdot F| > \frac{\rho r^{z(n,k)}}{24e^2} v_k^n F\right] \leq \frac{\Delta}{5d}$ follows from the Chernoff bounds above and the fact that $v_k^n \geq \frac{\rho n}{2ed}$. If we then condition on $f_k^1 + f_k^0 \geq \mu$, by Claim 2, this increases $\Pr\left[|\bar{v}_k^n \cdot F - v_k^n \cdot F| > \frac{\rho r^{z(n,k)}}{24e^2} v_k^n F\right]$ to at most $\frac{\Delta}{4d}$. Thus, with probability at most $\frac{\Delta}{4d}$, $|\bar{q}_k^n - q_k^n| > \frac{\rho r^{z(n,k)}}{24e^2} \frac{v_k^n F}{\mu} \geq \frac{\rho r^{z(n,k)}}{12e^2}$, where the second inequality again uses the fact that $v_k^n \geq \frac{\rho n}{2ed}$. ■

We next demonstrate what the implications of this are on our algorithm:

Claim 4

$$\left| p_k^0 - \sum_{j=1}^{z(n,k)} c_j^k B_{(j-1)d+k} \right| \leq |\bar{q}_k^n - q_k^n| - (|\bar{q}_k^n - q_k^n|) r^{z(n,k)}$$

Proof: Since $\alpha_j^k \leq 1$, $\left|p_k^0 - \sum_{j=1}^{z(n,k)} c_j^k B_{(j-1)d+k}\right| \leq \sum_{j=1}^{z(n,k)} |\bar{q}_k^n - q_k^n| \frac{r^{j-1}}{2} \leq |\bar{q}_k^n - q_k^n| - (|\bar{q}_k^n - q_k^n|) r^{z(n,k)}$. ■

Claim 5 *Given that* $f_k^0 + f_k^1 \geq \mu$, $\Pr\left[|p_k - p_k^0| > \frac{\bar{q}_k^n}{2} r^{z(n,k)} + (|\bar{q}_k^n - q_k^n|) r^{z(n,k)} + \frac{\rho r^{z(n,k)}}{12e^2}\right] \leq \frac{\Delta}{4d}$.

Proof: We here bound the probability that p_k is too large; the bound on the probability that p_k is too small is similar. It is easy to see that $\Pr\left[p_k - p_k^0 > \frac{\bar{q}_k^n}{2} r^{z(n,k)} + (|\bar{q}_k^n - q_k^n|) r^{z(n,k)} + \frac{\rho r^{z(n,k)}}{12e^2}\right]$ is maximized when the attacker sets all initial values of h_0 to 1, and thus we assume that the attacker does so. Note that this implies that $E[p_k] = p_k^1 - \frac{\bar{q}_k^n r^{z(n,k)}}{2} = p_k^0 + q_k^n r^{z(n,k)} - \frac{\bar{q}_k^n r^{z(n,k)}}{2} \leq p_k^0 + \frac{\bar{q}_k^n}{2} r^{z(n,k)} + (|\bar{q}_k^n - q_k^n|) r^{z(n,k)}$. We now let X_j, for $1 \leq j \leq t_k$, be a random variable, where $X_j = 1$ if the jth packet in P_k arrives to N_0 with $h_0 = 1$ and $X_j = 0$ otherwise, where $t_k = f_k^1 + f_k^0$. We shall bound the probability that $\sum_{j=1}^{t_k} X_j > t_k(p_k^1 + \frac{\rho r^{z(n,k)}}{12e^2})$.

Unfortunately, we can not use a Chernoff bound on this sum directly, since conditioning on $f_k^0 + f_k^1 \geq \mu$ can result in a small amount of dependence between the X_js (this is actually somewhat subtle). To remove this dependence, we partition the integers from 1 to t_k into two sets, where $j \in S_0$ if packet j arrives without being reset, and $j \in S_1$ otherwise. The variables X_j for $j \in S_0$ are independent, as are the variables X_j for $j \in S_1$. Let $s_0 = \Pr[X_j = 1]$ for $j \in S_0$. We see that $s_0 = r^{z(n,k)} + \sum_{j=1}^{z(n,k)} B_{k+(j-1)d} \frac{r^{j-1}}{2}$. Likewise, let $s_1 = \Pr[X_j = 1]$ for $j \in S_1$. We see that $s_1 = \sum_{j=1}^{z(n,k)} B_{k+(j-1)d} \alpha_j^k \frac{r^{j-1}}{2}$.

We show that for $w \in \{0,1\}$, $\Pr\left[\sum_{j \in S_w} X_j > |S_w| s_w + t_k \frac{\rho r^{z(n,k)}}{24e^2}\right] \leq \Delta/16d$. Since $|S_0| = q_k^n t_k$ and $|S_1| = (1 - q_k^n) t_k$, this implies that $\Pr\left[\sum_{j=1}^{t_k} X_j > t_k(p_k^1 + \frac{\rho r^{z(n,k)}}{12e^2})\right] \leq \Delta/8d$, from which the claim follows. By the first Chernoff bound above,

$$\Pr\left[\sum_{j \in S_w} X_j > |S_w| s_w + t_k \frac{\rho r^{z(n,k)}}{24e^2}\right] \leq e^{-\left(\frac{t_k \rho r^{z(n,k)}}{24e^2 |S_w| s_w}\right)^2 \frac{|S_w| s_w}{3}}.$$

This probability is maximized by making $|S_w| s_w$ as large as possible, but it must be the case that $|S_w| s_w \leq t_k$. Thus, we may consider only the case where $|S_w| s_w = t_k$. Now, since we are conditioning $t_k \geq \mu$, and we have that $b \geq 2$ and $\Delta \leq 1/8$, we see that

$$e^{-\left(\frac{t_k \rho r^{z(n,k)}}{24e^2 |S_w| s_w}\right)^2 \frac{|S_w| s_w}{3}} \leq \Delta/16d.$$

■

Now note that Claims 2, 3, 4, and 5 together give us that

$$\Pr\left[\left|p_k - \sum_{j=1}^{z(n,k)} c_j^k B_{(j-1)d+k}\right| > \frac{\bar{q}_k^n r^{z(n,k)}}{2} + \frac{\rho r^{z(n,k)}}{6e^2}\right] \leq 3\Delta/4d.$$

Thus, we only have left to show that

$$\frac{\bar{q}_k^n r^{z(n,k)}}{2} + \frac{\rho r^{z(n,k)}}{6e^2} \leq \left(\bar{q}_k^n + (1-\bar{q}_k^n)\alpha_{z(n,k)}^k\right)\frac{r^{z(n,k)}}{2},$$

or that $\frac{\rho}{3e^2} \leq (1-\bar{q}_k^n)\alpha_{z(n,k)}^k$. We have that

$$\alpha_{z(n,k)}^k \geq \frac{\rho(1-\rho)^{n-1}}{\sum_{t=1}^{z(n,k)} \rho},$$

and so using the assumption that $\rho \leq 1/n$, we see that $\alpha_{z(n,k)}^k \geq \frac{1}{ez(n,k)}$. Thus, we only need to show that $1 - \bar{q}_k^n \geq \frac{z(n,k)\rho}{3e}$. By the definition of $\bar{q}_k^n$, this is equivalent to $f_k^1 + f_k^0 \leq \frac{3e}{z(n,k)\rho} \cdot v_k^n \cdot F$. Since $v_k^n \geq \frac{z(n,k)\rho}{e}$, we only need that $f_k^1 + f_k^0 \leq 3F$. This follows simply from the fact that at worst, all the packets are in the set P_k. ■

3.1 Lower bound for a single path of attack

Recall that the lower bound model requires the memoryless Network to send an n-bit string to the Victim using b-bit packets. For any protocol $\mathcal{P}$, let $\mathcal{E}(\mathcal{P})$ be the expected number of packets received by the Victim when the input is chosen uniformly at random from the set of all 2^n possible inputs. Let $w(\mathcal{P})$ be the probability that using $\mathcal{P}$, the Victim does not return the input string given to the Network when that input is chosen uniformly at random from the set of all 2^n possible n-bit strings.

Theorem 3 *For any protocol $\mathcal{P}$, if $\mathcal{E}(\mathcal{P}) \leq \frac{2^b-1}{8e}2^{n/2^b} - 2^{b-2}$, then $w(\mathcal{P}) \geq 1/2$.*

Proof: Any algorithm employed by the Victim can be thought of as a (possibly randomized) procedure for deciding, for each possible sequence of packets that the Victim has received, whether or not to continue receiving packets, and if the Victim decides to not continue, then the procedure must specify a (possibly deterministic) probability distribution over possible results for the Victim to output. We refer to such an algorithm as a *general* protocol. A restricted class of protocols is *Monte Carlo* protocols, where the Victim waits until it has received exactly T packets, where T depends only on n and b. The protocol maps the set of T received packets to a distribution over possible results, which the Victim uses to produce an output.

Lemma 2 *For any general protocol $\mathcal{P}$, there is a Monte Carlo protocol $\mathcal{P}'$, such that $\mathcal{E}(\mathcal{P}') = 4\mathcal{E}(\mathcal{P})$, and $w(\mathcal{P}') \leq w(\mathcal{P}) + 1/4$.*

Proof: We define $\mathcal{P}'$ as follows: collect $T = 4\mathcal{E}(\mathcal{P})$ packets. Using the order that the packets arrive at the Victim, simulate the protocol $\mathcal{P}$. If $\mathcal{P}$ produces a result before it receives T packets, then $\mathcal{P}'$

produces the same result, ignoring the remainder of the packet that it has. If $\mathcal{P}$ has not produced a result after receiving T packets, then $\mathcal{P}'$ specifies to the Victim to output a result chosen uniformly at random from the set of all 2^n possible outputs. The bound on $w(\mathcal{P}')$ follows from the fact that by Markov's inequality, the probability that $\mathcal{P}$ has not produced a result after receiving T packets is at most $1/4$. ■

We shall demonstrate that for any Monte Carlo protocol, if the number of packets received is too small, then the probability that the protocol makes a mistake is at least $3/4$. This result, combined with Lemma 2, implies the theorem. Thus, we henceforth only consider Monte Carlo protocols.

The input to the Victim can be described via a *receipt sequence*: a sequence $(r_1 \dots r_T)$, where r_i is a b-bit string describing the ith b-bit packet that is received by the Victim. Any Monte Carlo protocol for the Victim is a function that maps a receipt sequence to a probability distribution over n-bit strings. Another kind of description of the input to the Victim is a *receipt profile*: a 2^b-tuple $R = (f_0, \dots, f_{2^b-1})$, where f_i is the number of packets of type i received by the Victim. Note that $\sum_{j=0}^{2^b-1} f_j = T$. For any receipt profile R, let $S(R)$ be the set of receipt sequences S such that for all i, $0 \leq i \leq 2^b - 1$, the number of packets of type i in the sequence S is exactly f_i. Let a *permutation oblivious* algorithm for the Victim be a function that maps a receipt profile to a probability distribution over n-bit strings. Intuitively, a permutation oblivious algorithm is a Monte Carlo algorithm that ignores the permutation information of the input, and only uses the receipt profile of the input.

Lemma 3 *For any Monte Carlo algorithm $\mathcal{P}'$ for the Victim, there is a permutation oblivious algorithm $\mathcal{P}''$ for the Victim, such that $\mathcal{E}(\mathcal{P}'') = \mathcal{E}(\mathcal{P}')$, and $w(\mathcal{P}'') = w(\mathcal{P}')$.*

Proof: Given a Monte Carlo algorithm $\mathcal{P}'$ for the Victim, we define $\mathcal{P}''$ as follows: on an input receipt profile R, choose a receipt sequence S from $S(R)$ uniformly at random. The probability distribution over n-bit strings returned by $\mathcal{P}''$ is the same as $\mathcal{P}'$ would return when the input is S. To see that $w(\mathcal{P}'') = w(\mathcal{P}')$, note that since the Network is memoryless, on any n-bit string that is input to the Network, and for any receipt profile R, the probability that the receipt sequence is any receipt sequence in $S(R)$ is the same for all receipt sequences in $S(R)$. Thus, it does not matter whether the Network "chooses" a receipt sequence uniformly at random from the set of receipt sequences in the receipt profile, or whether the Victim makes this same choice. ■

Thus, we can simply show a lower bound for permutation oblivious algorithms, and this will imply a lower bound for all possible algorithms. Let $\psi(T)$ be the set of all possible receipt profiles for which the total number of packets received is exactly T. Let $\iota(n)$ be the set of all 2^n inputs to the Network. For any $\tau \in \psi(T)$ and $I \in \iota(n)$, let $p(\tau, I)$ be the probability that the algorithm outputs I when the receipt profile is τ. Note that for any input I, the probability that the algorithm outputs I, given that the Network receives the input I, is at most $\sum_{\tau \in \psi(T)} p(\tau, I)$. Thus, for any permutation oblivious algorithm $\mathcal{P}''$,

$$w(\mathcal{P}'') \geq \frac{\sum_{I \in \iota} \left(1 - \sum_{\tau \in \psi(T)} p(\tau, I)\right)}{2^n}.$$

Now, note that

$$\sum_{\tau \in \psi(T); I \in \iota(n)} p(\tau, I) \leq |\psi(T)|.$$

This implies that $w(\mathcal{P}'') \geq 1 - \frac{|\psi(T)|}{2^n}$. Thus, if $|\psi(T)| \leq 2^n/4$, the permutation oblivious protocol must make a mistake with probability at least $3/4$. Thus, we only need to compute $|\psi(T)|$ for a given value of b. By a standard combinatorial argument, the number of receipt profiles in $\psi(T)$ is simply

$$\binom{T + 2^b - 1}{2^b - 1} \leq \left(\frac{(T + 2^b - 1)e}{2^b - 1}\right)^{2^b - 1}.$$

Thus, $w(\mathcal{P}'') \geq 3/4$, provided that $\left(\frac{(T+2^b)e}{2^b-1}\right)^{2^b} \leq 2^n/4$, or that $T \leq \frac{2^b-1}{2e} 2^{n/2^b} - 2^b$. By Lemmas 2 and 3, this implies that for any general protocol $\mathcal{P}$, $p(\mathcal{P}) \geq 1/2$, provided that $\mathcal{E}(\mathcal{P}) \leq \frac{2^b-1}{8e} 2^{n/2^b} - 2^{b-2}$. ■

We also point out that a slightly tighter analysis using the same techniques gives us that when $b = 1$, the lower bound is $\Omega(2^n)$. Furthermore, a tighter analysis of the protocol for the case where the attacker sets the initial bits randomly leads to an upper bound of $O(b 2^b 2^{4n/2^b})$. Since such an attacker can be simulated in the lower bound model, this is also an upper bound for the lower bound model. Asymptotically, this differs from our lower bound by only a factor of 4 in the exponent, and a factor of b.

4 Multiple paths of attack.

We next consider the case where the packets sent to the victim travel on multiple paths. For protocols, we assume the same model as in the single path of attack case (i.e., complete binary tree of height n and every node sees which child it receives any given packet from.) In addition, there is a parameter k that represents an upper bound on the number of paths of attack. We assume that at the start of the attack, the attacker chooses a set of at most k nodes, and then for each packet it sends, it chooses which of the k nodes sends that packet to the victim. A protocol should work correctly as long as the attacker chooses k or less paths of attack, but can have any behavior in the case that the attacker uses more paths.

We also introduce a second parameter α. To see why, notice that if the attacker sends all but one of its packets along one path, for small values of b it is not possible for the victim to determine the path used by the single packet that takes a different path. The parameter α represents the relative bias in the number of packets that must be sent along a path in order for the victim to recover that path. In particular, we say that a protocol is α-sensitive, if during any given attack, the victim is able to reconstruct (with sufficiently high probability) all paths P, such that at least a fraction of $\frac{\alpha}{k}$ of the packets the attacker sends travel along P. Note that protocols where $\alpha > 1$ are not of interest to us, since the attacker could choose to send an equal number of packets along every path, in which case an α-sensitive protocol with $\alpha > 1$ would not be guaranteed to return any path information.

We here also make the assumption that the attacker sends each packet with the initial b header bits set to 0. The lower bounds we prove also hold without this restriction, since the attacker can always choose to do this. This assumption does restrict the applicability of the protocol that we introduce. However, we consider protocols in this model an important step towards a full solution. We also state without proof that the technique for the multiple path protocol can be adapted to work for a number of different restrictions on the adversary (for example, it can be adapted to a model where the attacker chooses the initial bits using a uniform or any other known distribution). Furthermore, the technique we use for this model looks promising in terms of a general solution, and may also be of independent interest.

For the lower bounds, we assume the same model as the lower bounds for the single path of attack case, except that the Network now has k strings to send to the Victim, but it only has access to one of these strings for each packet that it sends to the Victim. Each time the Network sends a packet, the *Attacker* is allowed to choose which of the n strings the Network sees. Since the Network has no memory, it can only use the current string in determining the contents of each b-bit packet. We shall refer to each of the n-bit strings of the Victim as a path to be determined.

4.1 Intuition

We demonstrate that if $b \leq \log(2k-2)$, then the attacker is information theoretically able to hide its location in the network. Specifically, regardless of the number of packets received by the victim, the victim is not able to determine even a single path P such that the probability that P is an actual path of attack is greater than 1/2. On the other hand, if $b \geq \lceil \log(2k+1) \rceil$, then there is a protocol such that for any α and Δ, with probability at least $1-\Delta$, the packets received by the victim encode all paths used to send a fraction of at least $\frac{\alpha}{k}$ of the packets. To see why b must grow as k grows, consider first the case where $k=2$ and $b=1$. We describe why increasing k from 1 to 2 allows the attacker to "hide" its location in the network.

As is described in the single path of attack case, a protocol communicates information to the victim by specifying the probability distribution of the header bits that arrive at the victim for a packet that travels along a given path. For any path P, let $p_1(P)$ be the probability that a packet traveling along P has the single header bit set to 1 when it arrives at the victim. Now, consider the case where there are three possible paths of attack P_1, P_2 and P_3, out of which the attacker is allowed to choose up to $k=2$. We can assume w.l.o.g. that $p_1(P_1) \leq p_1(P_2) \leq p_1(P_3)$. Consider two different attacker strategies: 1) the attacker sends all packet along path P_2, and 2) for each packet independently, the attacker sends the packet along P_1 with probability $\frac{p_1(P_2)-p_1(P_3)}{p_1(P_1)-p_1(P_3)}$ and along P_3 with probability $1-\frac{p_1(P_2)-p_1(P_3)}{p_1(P_1)-p_1(P_3)} = \frac{p_1(P_1)-p_1(P_2)}{p_1(P_1)-p_1(P_3)}$. In both attacker strategies, the probability that the header bit of any packet is 1 is $p_1(P_2)$, but the two cases do not share any paths. Thus, when the probability of receiving a 1 is $p_1(P_2)$, the victim is not able to do better than guessing between the two cases. Also note that obtaining more packets does not give any information beyond a better estimate of $p_1(P_2)$, and thus increasing the number of packets received is not helpful.

With this motivation, we now see that the encoding for the paths is as follows: for any path P, let $p_i(P)$ be the probability that a packet sent along path P arrives at the victim with the header bits set to the binary representation of i. Each path P can be represented by a vector $V'(P)$ of length 2^b, where component i of $V'(P)$, for $1 \leq i \leq 2^b-1$, is $p_i(P)$, and component 2^b is $p_0(P)$. Since it must be the case that $\sum_{i=1}^{2^b} p_i(P) = 1$, we can represent the distribution as the vector $V(P)$, which is the same as the vector $V'(P)$ except it does not have component 2^b, and thus has length 2^b-1.

Let $A_1 \ldots A_k$ be the k paths used by the attacker. For the moment, assume that the attacker decides on a set of probabilities $\lambda_1 \ldots \lambda_k$ such that λ_i is the probability that a packet is sent on path i. With such a strategy, for any packet, there is a probability q_i that a packet received by the victim has the header bits set to i. Let W be the vector with 2^b-1 components such that entry i of W is q_i. We see that $W = \sum_{j=1}^{k} \lambda_j V(A_j)$. Thus, in order to have a encoding for the set of 2^n

paths $P_1 \ldots P_{2^n}$ that allows the victim to correctly determine the paths of attack being used, we need a set of 2^n vectors $V(P_1) \ldots V(P_{2^n})$ such that for any two sets of k paths $S_1 = \{A_1, \ldots, A_k\}$ and $S_2 = \{A'_1, \ldots, A'_k\}$, there are no two sets of probabilities $\lambda_1 \ldots \lambda_k$ and $\lambda'_1 \ldots \lambda'_k$, such that $\sum_{j=1}^{k} \lambda_j V(A_j) = \sum_{j=1}^{k} \lambda'_j V(A'_j)$, and either for some j such that $A_j \notin S_2$, $\lambda_j > 0$, or for some j such that $A'_j \notin S_1$, $\lambda'_j > 0$. Note that this is a weaker requirement than that the set of vectors be $2k$-wise independent in the following sense: if the vectors are $2k$-wise independent, then they satisfy our requirement. However, a set of vectors that satisfies our requirement is not necessarily $2k$-wise independent.

4.2 Lower bound for multiple paths of attack

Note that if the requirement on the set of vectors were exactly that they be $2k$-wise independent, then we would immediately have a lower bound of $b \geq \log(2k+1)$, since any smaller value of b would result in vectors with less than $2k$ components. Still, we can demonstrate the following lower bound of $\log(2k-1)$ by using the actual requirement on the vectors. We point out that this bound is information theoretic, and thus applies to all previous PPM techniques for the case of multiple paths of attack.

Theorem 4 *If $b \leq \log(2k-2)$ and there are at least $2k$ paths out of which the k paths of attack are chosen, then the Attacker can cause a situation where regardless of how many packets the Victim receives, it is not able to determine any path P such that P is one of the paths of the Network with probability at least $1/2$.*

Proof: We first demonstrate that if the Attacker can cause the same distribution of header bits to be received at the Victim for two disjoint sets of paths, then the Attacker can cause the situation described. Given two sets of paths $S_1 = \{P_1, P_2, \ldots, P_k\}$ and $S_2 = \{P'_1, P'_2, \ldots, P'_k\}$, we say that S_1 and S_2 are *convex equivalent* if there are probabilities $\lambda_1, \ldots, \lambda_k, \lambda'_1, \ldots, \lambda'_k$ such that $\sum_{j=1}^{k} \lambda_j V(P_j) = \sum_{j=1}^{k} \lambda'_j V(P'_j), \sum_{i=1}^{k} \lambda_i = 1$ and $\sum_{i=1}^{k} \lambda'_i = 1$, .

Lemma 4 *For any protocol, if there exist two disjoint sets of paths S_1 and S_2 that are convex equivalent and are each of size at most k, then the Attacker can create a situation such that the Victim is unable to return a single path P that is held by the Network with probability greater than* $1/2$.

Proof: Let A_1 be an Attacker strategy where the Network has the set of paths S_1, and the Attacker chooses the path for each packet by choosing path P_j with probability λ_j independently of the choice for all previous packets. Let A_2 be an Attacker strategy where the Network has the set of paths S_2 and the Attacker chooses path P'_j with probability λ'_j independently of all previous packets. For both Attacker strategies, the probability distribution over header bits received by the Victim is the same. Let W be the $2^b - 1$ dimensional vector that describes this distribution. We consider the scenario where the Attacker chooses each of strategies A_1 and A_2 with probability $1/2$.

If, at the start of the attack, we reveal to the Victim some additional side information, in particular the vector W, then (by what is referred to as the "little birdie" principle) this cannot make the Victim's task any harder. If the Victim knows W, then the packets that arrive to the victim do not provide it with any additional information, since it knows W, and could simulate any such packets without actually seeing them. Therefore, regardless of how many packets the Victim receives, the Victim does not obtain any information past the vector W. However, with W, both strategy A_1 and strategy A_2 are equally likely. Since sets S_1 and S_2 are disjoint, the Victim is not able to determine any path that is in the set of paths of the Network with probability greater than 1/2. ■

To complete the proof of the Theorem, we show that if $b \leq \log(2k-2)$ and there are at least $2k$ possible paths out of which the k paths of the Network are chosen, then there exist two disjoint sets of paths S_1 and S_2 of size at most k that are convex equivalent. Let the $2k$ paths be $P_0, P_1, \ldots, P_{2k-1}$. Let Z be the zero vector. We first show that we can assume that $V(P_0) = Z$.

Claim 6 *For a given value of b, if for any arbitrary $V(P_1) \ldots V(P_{2k-1})$ and $V(P_0) = Z$ there exist two disjoint sets of paths S_1 and S_2, each of size at most k, that are convex equivalent, then it is also the case that for any arbitrary $V(P_0) \ldots V(P_{2k-1})$ there exist two disjoint sets of paths S_1 and S_2, each of size at most k, that are convex equivalent.*

Proof: Given an arbitrary set of path vectors $V(P_0) \ldots\ V(P_{2k-1})$, for $0 \leq i \leq 2k-1$, let $V_i = V(P_i) - V(P_0)$. Thus $V_0 = Z$. Given probabilities λ_i, $0 \leq i \leq 2k-1$, such that $\sum_{i=0}^{k-1} \lambda_i = 1$ and $\sum_{i=k}^{2k-1} \lambda_i = 1$, if $\sum_{j=0}^{k-1} \lambda_j V_j = \sum_{j=k}^{2k-1} \lambda_j V_j$, then $\sum_{j=0}^{k-1} \lambda_j (V(P_j) - V(P_0)) = \sum_{j=k}^{2k-1} \lambda_j (V(P_j) - V(P_0))$. This implies that $\sum_{j=0}^{k-1} \lambda_j V(P_j) = \sum_{j=k}^{2k-1} \lambda_j V(P_j)$. ■

Thus, we henceforth assume that $V(P_0) = Z$. Furthermore, we can also assume that $V(P_i) \neq Z$, for $i > 0$, since the theorem is trivial if two path vectors are the same.

Claim 7 *If there exist $\mu_1 \ldots \mu_{2k-2}$ such that $\forall j, \mu_j \geq 0$, $\exists j$ such that $\mu_j > 0$, and $\sum_{j=1}^{k-1} \mu_j V(P_j) = \sum_{j=k}^{2k-2} \mu_j V(P_j)$, then there exist two disjoint sets of paths S_1 and S_2, each of size at most k, that are convex equivalent.*

Proof: Let $Q_1 = \sum_{i=1}^{k-1} \mu_i$, and let $Q_2 = \sum_{j=k}^{2k-2} \mu_j$. We can assume w.l.o.g. that $Q_2 \geq Q_1$. The two sets are $S_1 = \{P_0, \ldots, P_{k-1}\}$ and $S_2 = \{P_k, \ldots, P_{2k-1}\}$. If we let $\mu'_j = \mu_j / Q_2$, for $1 \leq j \leq 2k-2$, $\mu'_0 = 1 - \frac{Q_1}{Q_2}$, and $\mu'_{2k-1} = 0$, then we see that $\sum_{j=k}^{2k-1} \mu'_j = 1$, and $\sum_{j=0}^{k-1} \mu'_j = 1$. Furthermore, it must be the case that $\sum_{j=0}^{k-1} \mu'_j V(P_j) = \sum_{j=k}^{2k-1} \mu'_j V(P_j)$, since we have merely multiplied both sides by a scalar, and added the zero vector. ■

Thus, we only need to find such $\mu_1 \ldots \mu_{2k-2}$. If $b \leq \log(2k-2)$, the dimension of the vectors is at most $2k-3$, and thus there must exist some P_i and P_j, $i, j > 0$ and $j \neq i$, such that $V(P_i)$ and $V(P_j)$ are each a linear combination of the $2k-3$ other non-zero vectors. We can assume w.l.o.g. that $i = 1$ and $j = 2$. Thus, there exist $\lambda_1 \ldots \lambda_{2k-1}$ such that $\sum_{i=1}^{2k-1} \lambda_i V(P_i) = V(P_0)$, where $\lambda_1 = -1$ and $\lambda_2 = 0$. Similarly, there exist $\lambda'_1 \ldots \lambda'_{2k-1}$ such that $\sum_{i=1}^{2k-1} \lambda'_i V(P_i) = V(P_0)$, where $\lambda'_1 = 0$, and $\lambda'_2 = -1$.

Let T_1^+ (T_2^+) be the set of i such that $\lambda_i > 0$ ($\lambda'_i > 0$, respectively), let T_1^- (T_2^-) be the set of i such that $\lambda_i < 0$ ($\lambda'_i < 0$, respectively), and let T_1^0 (T_2^0) be the set of i such that $\lambda_i = 0$ ($\lambda'_i = 0$, respectively). If $|T_1^+| \leq k-1$ and $|T_1^-| \leq k-1$, then we can obtain $\mu_1 \ldots \mu_{2k-2}$ by renumbering the P_i so that the P_i with $i \in T_1^+$ are numbered using integers from $[1, k-1]$, the P_i with $i \in T_1^-$ are numbered using integers from $[k, 2k-2]$, and the remainder of the paths are numbered arbitrarily to fill in the remaining integers. By renumbering the λ_i in the corresponding fashion, and by negating λ_i for $i \in T_1^-$, we obtain the required values of μ_i. Thus, we henceforth assume that either $|T_1^+| \geq k$

or $|T_1^-| \geq k$. In fact, since we could multiply all of the λ_i by -1, we shall assume that $|T_1^+| \geq k$. Similarly, we can assume that $|T_2^-| \geq k$.

Using the fact that $|T_2^-| \geq k$, we see that there exists R such that for any $r > R$, the number of values of i such that $\lambda_i + r\lambda'_i < 0$ is at least k. Since $|T_1^+| \geq k$, there must exist some value s, $0 < s \leq R$ such that the number of values of i such that $\lambda_i + s\lambda'_i > 0$ is at most $k-1$, and the number of values of i such that $\lambda_i + s\lambda'_i < 0$ is also at most $k-1$. Thus, we can obtain $\mu_1 \ldots \mu_{2k-2}$ by setting $\mu_1 \ldots \mu_{k-1}$ to nonnegative values of $\lambda_i + s\lambda'_i$, and $\mu_k \ldots \mu_{2k-2}$ to the negation of nonpositive values of $\lambda_i - s\lambda'_i$, where the 0 values are assigned so as to ensure that both sets have exactly $k-1$ elements. To satisfy the conditions of Claim 7, we also renumber the paths as appropriate.

This gives us the required order on the paths and required values of $\mu_1 \ldots \mu_{2k-2}$. Thus, by Claim 7 and Lemma 4, there is an Attacker strategy such that there is no path P that the Victim can determine such that probability that P is held by the Network is greater than 1/2. ■

4.3 Upper Bound for Multiple Paths of Attack

We saw in the previous subsection that if there are two disjoint sets of paths S_1 and S_2 that are convex equivalent, then the attacker is able to hide in the network. In this section we demonstrate how to encode an arbitrarily large set of paths in such a way that the resulting vectors produce no such sets S_1 and S_2. In fact, our technique produces a set of vectors that satisfy a stronger criteria: every set of $2k$ vectors is linearly independent. In order to do so, let $d = 2^b - 1$. We shall consider a curve in d-dimensional space such that **any** set of $2k$ distinct vectors with endpoints on this curve are linearly independent. With our encoding, the vector for every path lies on this curve.

This curve is defined in terms of a parameter t. Let $\mathcal{V}(t)$ be the d-dimensional vector such that the ith component of $\mathcal{V}(t)$ is t^i. As in the case of a single path of attack, let any path P be described by bits $B_1(P) \ldots B_n(P)$, which specify the entire path from the victim to the attacker. To determine the path P, it is sufficient to determine the value $X_P = \sum_{i=1}^{n} B_i(P)/2^i$. To encode the path P, we use the probability distribution defined by the vector $V(P) = \mathcal{V}(\frac{1}{4}X_P)$.

We first demonstrate how to compute the vectors on this curve in a distributed fashion. Our technique works correctly provided that $b \geq \lceil \log(2k+1) \rceil$, i.e., that $d+1$ (the number of possible headers) is at least $2k+1$. This technique does not require the intermediate nodes of the network to know the value of k; they are only required to know the value of b. Recall that $p_i(P)$ is the probability that a packet sent along path P arrives at the victim with the header bits set to i. We describe a protocol for each of the distributed nodes such that $p_i(P) = (\frac{1}{4}X_P)^i$, for $i > 0$, and $p_0(P) = 1 - \sum_{i=1}^{d} p_i(P)$.

Let $p_{i,j}^e$ be the probability that a node holding the bit e, for $e \in \{0,1\}$, forwards the header j when it receives the header i. Note that it must be the case that $\forall i, e, \sum_{j=0}^{d} p_{i,j}^e = 1$. When a node holds the bit 0, the probability transitions are defined as follows:

- For $0 < i \leq d$, $p_{i,i}^0 = 2^{-i}$, and $p_{i,0}^0 = 1 - 2^{-i}$.
- For $i \neq j$, and $j \neq 0$, $p_{i,j}^0 = 0$.
- $p_{0,0}^0 = 1$.

When a node holds the bit 1, the probability transitions are defined as follows:

- For $1 \leq i \leq j \leq d$, $p_{i,j}^1 = 2^{2i-3j}\binom{j}{i} + 2^{-3j}$.

- For $1 \le j < i \le d$, or $i = 0 < j \le d$, $p^1_{i,j} = 2^{-3j}$.
- For $j = 0 \le i \le d$, $p^1_{i,j} = 1 - \sum_{j=1}^d p^1_{i,j}$.

Claim 8 *For each possible header received by a node, this protocol defines a valid probability distribution over headers that the node forwards. In particular,* $\forall i, j, e$, $0 \le p^e_{i,j} \le 1$, *and* $\forall i, e$, $\sum_{j=0}^d p^e_{i,j} = 1$.

Proof: The proof of this fact is easy for the case where $e = 0$, as well as the case where $e = 1$ and $i = 0$. Thus, we here show that for any i, $1 \le i \le d$, $\sum_{j=1}^d p^1_{i,j} < 1$. Since $p^1_{i,0} = 1 - \sum_{j=1}^d p^1_{i,j}$, the claim then follows easily. For any i, we see that $\sum_{j=1}^d p^e_{i,j} = \sum_{j=1}^d 2^{-3j} + \sum_{j=i}^d 2^{2i-3j}\binom{j}{i}$. Since we know that $\sum_{j=1}^d 2^{-3j} < 1/7$, we only need to demonstrate that the second sum is at most $6/7$. We see that $\sum_{j=i}^d 2^{2i-3j}\binom{j}{i} = 2^{-i} + \sum_{j=i+1}^d 2^{2i-3j}\binom{j}{i}$. Since $\binom{j}{i} < 2^j$, this sum is less than $\frac{1}{2} + \sum_{j=i+1}^d 2^{2i-2j} = \frac{1}{2} + \sum_{j=1}^{d-i} 2^{-2j} < \frac{5}{6} < \frac{6}{7}$. ■

Claim 9 *For any path P and $1 \le i \le d$, $p_i(P) = \left(\frac{X_P}{4}\right)^i$.*

Proof: We prove this by induction on n. We start with the inductive step: if we assume that the claim is true for paths of length $n-1$, we can show that it is true for paths of length n. Let $p^{n-1}_i(P)$ be the probability that a packet sent on path P received by the node just prior to the victim has the header bits set to i. Since all nodes perform the same protocol, the inductive hypothesis gives us that $p^{n-1}_i(P) = \left(\frac{X^{n-1}_P}{4}\right)^i$, where $X^{n-1}_P = \sum_{i=2}^n b_i(P)2^{1-i}$. By the definition of the $p^0_{i,j}$, if $b_1(P) = 0$, then $p_i(P) = 2^{-i}p^{n-1}_i(P)$, and thus $p_i(P) = \left(\frac{X^{n-1}_P}{8}\right)^i = \left(\frac{X_P}{4}\right)^i$.

Similarly, for the case where $b_1(P) = 1$, we only need to show that $p_j(P) = \left(\frac{X^{n-1}_P}{8} + \frac{1}{8}\right)^j$. In the following, we use the standard convention that $\binom{j}{i} = 0$ if $j < i$. We see that for all $j > 0$,

$$
\begin{aligned}
p_j(P) = \sum_{i=0}^{d} p^{n-1}_i(P)p^1_{i,j} = \\
\left(1 - \sum_{i=1}^{d}\left(\frac{X^{n-1}_P}{4}\right)^i\right)2^{-3j} + \sum_{i=1}^{d}\left(\frac{X^{n-1}_P}{4}\right)^i\left(2^{2i-3j}\binom{j}{i} + 2^{-3j}\right) = \\
2^{-3j} + \sum_{i=1}^{d}\left(\frac{X^{n-1}_P}{4}\right)^i 2^{2i-3j}\binom{j}{i} = \\
\sum_{i=0}^{j}\left(\frac{X^{n-1}_P}{4}\right)^i 2^{2i-3j}\binom{j}{i} = \\
\sum_{i=0}^{j}\binom{j}{i}\left(\frac{X^{n-1}_P}{8}\right)^i\left(\frac{1}{8}\right)^{j-i} = \\
\left(\frac{X^{n-1}_P}{8} + \frac{1}{8}\right)^j
\end{aligned}
$$

The base case of the inductive proof follows from a similar argument, since we assume that the attacker must set all header bits to 0 in the packets it transmits. ■

We next demonstrate that with high probability, this process provides the victim with information that specifies all paths P that receive a large enough fraction of packets.

Theorem 5 *After the victim has collected* $6\left[\frac{48k^2}{\alpha}2^{(2k^2+k)(n+2)}\right]^2 \ln\frac{2k}{\Delta}$ *packets, with probability at least* $1-\Delta$*, the victim is able to determine all paths* P *such that at least a fraction of* $\frac{\alpha}{k}$ *of the packets the attacker sends travel along* P*.*

Proof: We here provide the proof for the case where there are at least $2k$ possible paths for the attacker to choose from; it is not difficult to remove this assumption. For convenience, we also assume that the encoding is done in such a manner that there is no path P such that $X_P = 0$. This can be assured either by using an encoding of the paths that does not have such a path, or by having the victim append a 1 to the end of every path description. Denote the k paths used by the attacker as $P_1 \dots P_k$. Let λ_i be the fraction of the received packets that are sent by the attacker along path P_i. If the attacker uses only k' paths, for $k' < k$, then choose an arbitrary set of $k - k'$ other paths so that there are k disjoint paths, and set the corresponding values of $\lambda_i = 0$. The probability that a randomly chosen packet from the set of received packets has its header bits set to i is $q_i = \sum_{j=1}^{k} \lambda_j p_i(P_j)$. The set of received packets provides the victim with an estimate on the values of the q_i.

To get some intuition, let's first assume that the victim knows the values of the q_i exactly. We demonstrate that this uniquely determines the entire set of paths used by the attacker. To prove this, we show that if we assume that it does not uniquely determine this set of paths, we reach a contradiction. In particular, assume that there is some set $P_{k+1} \dots P_{2k}$ of paths and fractions $\lambda_{k+1} \dots \lambda_{2k}$ such that $\sum_{j=1}^{k} \lambda_j V(P_j) = \sum_{j=k+1}^{2k} \lambda_j V(P_j)$. For the set of paths to not be uniquely determined, it must be the case that there is some path P_j, such that $\lambda_j > 0$ and if $j \leq k$ then $P_j \notin \{P_{k+1}, \dots, P_{2k}\}$, and if $j > k$ then $P_j \notin \{P_1, \dots, P_k\}$. Assume here that such a path is path P_{2k}; the case where $j \leq k$ is similar. In this case, we see that

$$\lambda_{2k} V(P_{2k}) = \sum_{j=1}^{k} \lambda_j V(P_j) - \sum_{j=k+1}^{2k-1} \lambda_j V(P_j).$$

This in turn implies that there is some set of $2k$ distinct paths $P'_1 \dots P'_{2k}$ and real numbers $\lambda'_1 \dots \lambda'_{2k}$, with $\lambda'_{2k} > 0$, such that

$$\lambda'_{2k} V(P'_{2k}) = \sum_{j=1}^{2k-1} \lambda'_j V(P'_j). \tag{1}$$

Now, consider the $2k \times 2k$ matrix M where entry $M_{i,j} = p_i(P'_j)$. From (1), we see that the columns of M must be linearly dependent. However, from Claim 9, we see that $M_{i,j} = \left(\frac{X_{P'_j}}{4}\right)^i$. The matrix M', where entry $M'_{i,j} = \left(\frac{X_{P'_j}}{4}\right)^{i-1}$, is a Vandermonde matrix. Since the paths $P'_1 \ldots P'_{2k}$ are distinct, if $i \neq j$ then $X_{P'_i} \neq X_{P'_j}$, and thus M' has full rank. Since we assume that for all paths P, $X_P \neq 0$, this implies that the matrix M has full rank as well, which contradicts the assumption that the columns of M are linearly dependent. Therefore, the exact values of the q_i exactly determines all paths P_j, $1 \leq j \leq k$, such that $\lambda_k > 0$.

We next examine the effect of the fact that the victim does not know the values of the q_i exactly. In this case, with high probability the victim determines a good estimate on all of the q_i values. We demonstrate that with this estimate, any path that is used to send a large enough fraction of the packets can be determined. The estimate used is as follows: let Y_i be the number of times that header i is seen in the $N = 6\left[\frac{48k^2}{\alpha}2^{(2k^2+k)(n+2)}\right]^2 \ln\frac{2k}{\Delta}$ packets. We set $\bar{q}_i = Y_i/N$. The victim returns any path P_j such that P_j is contained in a linear combination of at most k path vectors, with the coefficient associated with P_j being at least $\frac{\alpha}{k}$, such that the Euclidean distance of the resulting linear combination from the corresponding point defined by the $\bar{q}_i$s is at most $h_0 = \frac{1}{3}\frac{\alpha}{k}2^{-(2k^2+k)(n+2)}$.

To see that this protocol does in fact return every path P such that a fraction of at least $\frac{\alpha}{k}$ of the packets travel on P, let $D_q = \sqrt{\sum_{i=0}^{2k}(q_i - \bar{q}_i)^2}$. Standard Chernoff bound techniques demonstrate that with Npackets, the values $\bar{q}_0, \ldots, \bar{q}_{2k}$ are such that $\Pr[D_q > D_0] \leq \Delta$. Thus, with probability at least $1 - \Delta$, every required path is returned by the protocol.

We next show that there is no path P, such that P is not used by the attacker, but the probability that the protocol returns P is greater than Δ. If such a path exists, then there must be some set of paths $P_1 \ldots P_{2k}$, where $P_1 \ldots P_k$ are the paths used by the attacker, $P_{k+1} \ldots P_{2k}$ are the paths contained in the incorrect linear combination, and P_{2k} is the path returned incorrectly. Thus, $P_{2k} \notin \{P_1, \ldots, P_k\}$, and there exist fractions $\lambda_1 \ldots \lambda_{2k}$, with $\lambda_{2k} \geq \frac{\alpha}{k}$, such that

$$\Pr\left[\sqrt{\sum_{i=1}^{2k}\left(\sum_{j=k+1}^{2k}\lambda_j p_i(P_j) - \sum_{j=1}^{k}\lambda_j p_i(P_j)\right)^2} - D_q \leq D_0\right] > \Delta$$

This in turn implies that there are $2k$ distinct paths $P'_1, \ldots, P'_{2k}$ and real numbers $\lambda'_1 \ldots \lambda'_{2k}$, with $\lambda'_{2k} \geq \frac{\alpha}{k}$, such that

$$\sqrt{\sum_{i=1}^{2k}\left(\lambda'_{2k}p_i(P'_{2k}) - \sum_{j=1}^{2k-1}\lambda'_j p_i(P'_j)\right)^2} \leq 2D_0 \tag{2}$$

Let D_1 be the Euclidean distance in $\Re^{2k}$ from the point $\lambda'_{2k}V(P'_{2k})$ to the subspace spanned by $V(P'_1), \ldots V(P'_{2k-1})$. For (2) to be true, it must be the case that $D_1 \leq 2D_0$. Thus, to demonstrate that no such incorrectly returned path P_{2k} can exist, it is sufficient to show that $D_1 \geq \frac{\alpha}{k}2^{-(2k^2+k)(n+2)}$.

To see that this is the case, let $\mathcal{V}_1$ be the $2k$-dimensional volume of the parallelepiped defined by the vectors $V(P'_1), \ldots, V(P'_{2k-1}), \lambda_{2k}V(P'_{2k})$ in $\Re^{2k}$, and let $\mathcal{V}_2$ be the $(2k-1)$-dimensional volume of the parallelepiped defined by the vectors $V(P'_1), \ldots, V(P'_{2k-1})$ in $\Re^{2k}$. We see that $D_1 = \frac{\mathcal{V}_1}{\mathcal{V}_2}$. Since all of the vectors $V(P'_1) \ldots V(P'_{2k-1})$ have at most unit length, $\mathcal{V}_2 \leq 1$. Due to the convenient form of the vectors $V(P'_1), \ldots, V(P'_{2k})$, we can determine a lower bound on $\mathcal{V}_1$. In particular, a standard result from linear algebra is that $\mathcal{V}_1$ is equal to the absolute value of the determinant of the matrix T, where column j of T, for $1 \leq j \leq 2k-1$, is $V(P'_j)$, and column $2k$ is the vector $\lambda_{2k} V(P'_{2k})$.

To compute $|det(T)|$, consider the matrix T', where column j of T', for $1 \leq j \leq 2k$, is $\frac{4}{X_{P'_j}} V_j$. By Claim 9, the matrix T' is Vandermonde, and thus

$$det(T') = \prod_{1 \leq i < j \leq 2k} \left(\frac{X_{P'_i}}{4} - \frac{X_{P'_j}}{4} \right).$$

Since for any $i \neq j$, $\left| \frac{X_{P'_i}}{4} - \frac{X_{P'_j}}{4} \right| \geq \frac{1}{2^{n+2}}$, we see that $|det(T')| \geq (\frac{1}{2^{n+2}})^{\binom{2k}{2}}$. Since it is also the case that $\forall j$, $\frac{X_{P'_j}}{4} \geq \frac{1}{2^{n+2}}$, this implies that

$$|det(T)| \geq \frac{\alpha}{k} (\frac{1}{2^{n+2}})^{2k} (\frac{1}{2^{n+2}})^{\binom{2k}{2}}.$$

Thus, $\mathcal{V}_1 \geq \frac{\alpha}{k}(\frac{1}{2^{n+2}})^{2k^2+k}$. This implies that $D_1 \geq \frac{\alpha}{k}(\frac{1}{2^{n+2}})^{2k^2+k}$, completing the proof of the theorem. ■

We also point out that this use of Vandermonde matrices is fundamentally quite different from how they have been previously used in coding theory. There are constructions of linear codes where the generator matrix is Vandermonde (see, for example, [14]). On the other hand, in our use of Vandermonde matrices, the "codewords" themselves (i.e., the probability distributions over packet headers) have the property that any set of $2k$ codewords form a Vandermonde matrix. In fact, the traditional type of Vandermonde encoding was already used for PPM in [5], which relies on a technique from [1]. We point out that this technique results in a PPM encoding that falls into the class of protocols described in the introduction (where the victim only checks what packet headers it has received, as opposed to how many of each it has received), and hence is subject to the $\log n$ lower bound on b provided there.

5 Applications of the Model

In this section, we demonstrate that the upper bound model actually leads to a quite general solution. In particular, it is easy to map our solutions to a number of other models. We demonstrate this by considering three scenarios here: the case where the routers are not given the information of which child forwards them a given packet, the case where the underlying tree is not binary, and the case where neither the routers nor the victim have any information about the network topology or routing strategy other than their own unique ID.

We first consider the case where nodes still know the entire routing topology, and the tree is binary, but nodes do not obtain the information of which child they receive a given packet from. In this case, the information that is reconstructed is slightly different: instead of being able to reconstruct a path that contains the attacker, this smaller amount of information only allows us to reconstruct a path such that the attacker is a child of some node along that path. We refer to this model as the *source oblivious* model, and the model used for the bulk of the paper as the *source cognizant* model. We show that the two models are equivalent in the following sense:

Claim 10 *Any protocol for the source cognizant model for a tree of height n provides a protocol for the source oblivious model for a tree of height $n-1$. Any protocol for the source oblivious model for a tree of height n provides a protocol for the source cognizant model for a tree of height $n+1$.*

Proof: To simulate a protocol for the source cognizant model in the source oblivious model, each node simply does exactly what its parent would do in the source cognizant model. Each node clearly has enough information to do this. The parent of the attacker follows the source oblivious protocol correctly, and this simulates the parent of the parent of the attacker following the source cognizant protocol correctly. Thus, the path is followed correctly back to the parent of the attacker.

For the reverse simulation, each node N does exactly what its child N_c on the path would do in the source oblivious protocol on receiving the bits that N receives. In the case that N_c is actually the attacker, the node N simulates what N_c would do if it were not the attacker, but a child of N_c were the attacker. Since the node N has the information of which child it receives a packet from, N has enough information to perform such a simulation. This gives the source cognizant protocol the ability to obtain a path that contains the attacker. ■

We next consider the case where neither the victim nor the routers have any information about the network topology, nor does a router see the information of which child sends it a packet. Each router simply knows its own unique router ID, and our task is to inform the victim of an n-bit string that represents the concatenation of the unique IDs of the routers along the path of attack. These IDs might for example correspond to IP addresses. To use any of the protocols we have developed for the upper bound model in such a scenario, each router, on receiving a b-bit packet from a preceding router would simulate a complete binary tree of height k in the source oblivious model, where k is the length of its unique ID. The leaf of the tree that the router simulates receiving the packet is the leaf with a path description that is the same as the unique ID of the router. It is not difficult to see that the overall string received by the victim is the concatenation of the router IDs.

Finally, we consider the case where the victim knows the entire routing topology, and also each node is able to see which child it receives a given packet from, but the underlying tree is not binary. For this model, we could use the same technique of using unique IDs for the routers along the path of attack, but instead we present a more efficient encoding of the path. In particular, for every node N of the tree, we represent what child of that node is the predecessor of N in the attacker's path of attack using a Shannon code, where the distribution used in the code is the distribution over the children of N when a node of the subtree rooted at N is chosen uniformly at random.

Claim 11 *The maximum number of bits required to represent any path using this scheme is at most $h + \log m$, where m is the number of nodes in the system, and h is the height of the tree.*

Proof: Let $C_1 \ldots C_\ell$ be the sequence of code words used along any path from the victim to a node. Let T_j be the number of children in the subtree rooted at the jth node of this path. Since we are using a Shannon code, $\frac{T_j}{T_{j+1}} \geq 2^{|C_j|-1}$. Since $T_h = 1$ for any path, we have that $T_1 = \prod_{j=1}^{h-1} \frac{T_j}{T_{j+1}}$, and so $T_1 \geq 2^{(\sum_{i=1}^{\ell} |C_j|) - \ell}$. Since $m \geq T_1$, $\log m \geq (\sum_{i=1}^{\ell} |C_j|) - \ell$, from which the claim follows. ■

To use our protocols for the upper bound model to actually compute this encoding, consider some router R on the path of attack with t children, and corresponding code words $C_1 \ldots C_t$. On a message received from child j, router R simulates what would occur in the source cognizant model with a binary tree of height $|C_j|$ when the child specified by the path specified by C_j receives a packet from the attacker. This provides a mechanism for encoding the string C_j.

6 Conclusion

We have studied two scenarios for using PPM to solve the IP traceback problem: the case of a single path of attack, as well as the case of multiple paths of attack. For a single path of attack, we have introduced upper and lower bounds on the optimal tradeoff between the number of bits allocated to PPM and the number of packets that must be received to determine the source of the attack. For the case of multiple paths of attack, we have provided a lower bound on the number of header bits required as a function of the number of paths of attack. We also provide a nearly matching upper bound that applies to some restricted scenarios.

A number of interesting open problems remain. For the case of a single path of attack, it would be interesting to close the gap between the upper and lower bounds. For the case of multiple paths of attack, we have barely scratched the surface of obtaining a complete understanding of PPM. One interesting question is designing a protocol that does not rely on the restrictions on the attacker required by the protocol of Section 4.3. Furthermore, we have not addressed the issue of computational efficiency with respect to the decoding portion of that algorithm: it only provides an encoding that information theoretically specifies the correct paths. Finally, to obtain a complete understanding of the problem, we must incorporate the number of packets into the results for multiple paths of attack. In particular, we would like to obtain matching upper and lower bounds on the tradeoffs between the number of header bits used, the number of paths of attack, as well as the number of packets required.

7 Acknowledgments

The author would like to thank John Byers and Faith Fich for numerous helpful conversations and insightful suggestions. Thanks especially to Faith Fich for her help in simplifying the model for the protocols.

References

[1] Sigal Ar, Richard Lipton, Ronitt Rubinfeld, and Madhu Sudan, Reconstructing Algebraic Functions from Mixed Data. In *Proc. of 33^{rd} Annual Symposium on Foundations of Computer Science*, pp. 503-512, October 1992.

[2] S. M. Bellovin, ICMP Traceback Messages. Internet Draft: draft-bellovin-itrace-00.txt, Mar. 2000.

[3] Hal Burch and Bill Cheswick, Tracing Anonymous Packets to Their Approximate Source. In *Proc. Usenix LISA '00*, 2000.

[4] Sven Dietrich, Neil Long, and David Dittrich, Analyzing Distributed Denial of Service Attack Tools: The Shaft Case. In *Proc. 14th Systems Administration Conference, LISA 2000.*

[5] Drew Dean, Matt Franklin, and Adam Stubblefield, An Algebraic Approach to IP Traceback. In *Proc. 2001 Network and Distributed System Security Symposium.*

[6] Thomas Doeppner, Philip Klein, and Andrew Koyfman. Using router stamping to identify the source of IP packets. In*Proceedings of the 7th ACM Conference on Computer and Communications Security*, pages 184–189, Athens, Greece, November 2000.

[7] P. Ferguson and D. Senie, RFC 2267: Network Ingress Filetring: Defeating Denial of Service Attacks which employ IP Source Address Spoofing. *The Internet Society*, 1998.

[8] S. Floyd and V. Jacobson, Random Early Detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4), August 1997.

[9] S. Lee and C. Shields, Tracing the Source of Network Attack: A Technical, Legal and Societal Problem. In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, June 2001.

[10] Rajeev Motwani and Prabhakar Raghavan, *Randomized Algorithms.* Cambridge University Press, New York, NY, 1995.

[11] K. Park and H. Lee. On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack. In *Proc. IEEE INFOCOM '01*, pp. 338-347, 2001.

[12] K. Park and H. Lee. On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets. To appear in *Proc. ACM SIGCOMM '01*, August 2001.

[13] C. Perkins, IP Mobility Support. RFC 2002, Oct. 1996.

[14] L. Rizzo. Effective Erasure Codes for Reliable Computer Communication Protocols. *ACM Computer Communication Review*, Vol. 27, n.2, pp. 24-36, April 1997.

[15] Stefan Savage, David Wetherall, Anna Karlin and Tom Anderson, Practical Network Support for IP Traceback. In *Proceedings of ACM SIGCOMM 2000* , pp. 295-306, August 2000.

[16] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Stephen T. Kent, and W. Timothy Strayer. Hash-Based IP Traceback. To appear in *Proc. ACM SIGCOMM 2001*, August 2001.

[17] Dawn X. Song and Adrian Perrig, Advanced and authenticated marking schemes for IP traceback, In *Proc. IEEE INFOCOM '01*, 2001.