Abstract—Network attacks and in particular denial of service (DoS) attacks have emerged as a major way to compromise the availability of servers and interrupt legitimate online services provided by servers. These attacks are among one of the hardest security problems to address because they are simple to implement but hard to prevent and difficult to trace. Tracing the attacker after an attack is crucial to institute protection measures against future attacks. Packet marking schemes have been proposed to traceback an attacker. The idea is to insert some traceback data in each packet when it passes through a router and use this information to construct the attack path. The major challenges in these schemes are to minimize the number of packets for successful traceback and to reduce the number of bits marked per packet by any router along the attack path. A general approach is to encode the 32-bit IP address of the router and store it in the 16-bit ID field of the IP packet header. However, this will result in collisions. In this work we develop a novel packet marking scheme of assigning marks (colors) where routers at a distance of two hops can reuse the colors (star coloring). Our proposed schemes assign color or mark to each router in a network such that the total number of colors used in the network is minimized. We also propose a technique to construct the attack path using these colors and mathematically show that the probability of attack paths colliding is minimal.

I. INTRODUCTION

Network security attacks and in particular denial of service (DoS) attack is one of the most difficult problems in the field of security. Attackers can easily target the network systems without being traced as they spoof the IP address of the legitimate user. Defending against such attacks is two-fold. First we want to filter the attack packets during the onslaught of the attack. Secondly we want to locate the attacker to thwart future attacks and fix responsibility.

There are two major reasons that make DoS attack attractive for attackers. The first reason is that there are effective automatic tools available for attacking any victim, so expertise is not necessarily required. The second reason is that it is usually impossible to locate an attacker without extensive human interaction or without new features in most routers of the Internet.

In figure 1 we show the different possible types of DoS attacks. In its simplest form the attacker directly sends attack packets like SYN attack and Smurf attack ([2]). In the distributed version of DoS, attack tools are installed in a set of compromised computers called zombies or agents. The process of recruiting agents is done by scanning machines on the Internet, probing for security holes that will enable subversion. The attacker sends an attack command to these zombies who in turn send spoofed packets to the victim or reflectors (for example web servers). In the first case the victim is flooded with spoofed packets and in the second case the victim is flooded with replies from the reflector. The second type of attack also known as distributed reflector DoS (DRDoS) is more dangerous since the traffic volume is increased due to replies from the reflectors.

The various approaches to counter DoS attacks are attack prevention, attack detection, attack reaction and source identification. Attack prevention involves fixing security loop holes like insecure protocols, weak authentication etc. Detecting DoS attack while the attack is in progress is called attack detection. The challenge is in detecting quickly without misclassifying any legitimate traffic. Attack reaction involves curtailing the effect of the attack by filtering the malicious traffic without disturbing the legitimate traffic. Locating the source of the attack accurately regardless of spoofed attacks is called source identification or popularly known as IP traceback. All these approaches complement each other.

In this paper our main focus is on IP traceback. The major challenge in source identification mechanism is to minimize the number of packets required for successful traceback. We propose to assign colors (or marks) to the routers, embed these marks on the en-route IP packets and finally use these marks to construct the attack path. Space available in the IP header being limited our main aim is to optimize the number of bits required for marking. The organization of the paper is as follows. In section II we present the background and motivation of our work. An algorithm to traceback attackers in star colored networks is given in section III. In section IV
we examine the star chromatic number of network topologies and propose a star coloring scheme. In section V we discuss the deployment issues of our proposed traceback mechanism. A brief review of IP traceback techniques is given in VI. The conclusion and future direction of our work is given in VII.

II. BACKGROUND

Defense mechanisms against network security attacks basically involve two steps: (i) attack response during an attack and (ii) traceback the attacker. Attack response refers to the strategies that a defense triggers to mitigate an attack. This may involve dropping packets at the victim or its uplink routers. IP traceback refers to the problem of identifying the source of an attack to fix accountability as well as to thwart off future attacks. The final aim of this work is to traceback the attacker using minimum number of packets. The traceback algorithm should be able to function independently and require minimal cooperation of other domains.

IP traceback [20] methods are either reactive or proactive. In reactive approach the traceback is initiated in response to an attack. The process starts from the victim and it recursively queries its upstream router until the attacker is reached. The main requirement of the reactive approach is that the traceback must be completed while the attack is active. In the proactive approach the routers store and exchange tracing information as the packets are routed. This approach allows post-mortem analysis. A general proactive traceback approach is packet marking where routers insert marks to en route packets. These marks are later used for constructing the attack path to the attacker.

Most of the existing packet marking techniques require a large number of packets to converge on the attack path(s) [21]. This is because the existing solutions use IP address of the router as a mark and encode it in the 16-bit identification field of the IP packet header. Thus a minimum of two packets are required at the victim to collect information about a router. A complex mechanism is required to integrate the packet mark information. To reduce this overhead on the IP header as well as to make the traceback mechanism simple a number of packet marking schemes have been proposed. However, it has been observed that reducing the overhead leads to false positive. An overview of packet marking techniques is given in VI.

In this work we use a variant of packet marking techniques. Colors are pre-assigned to routers and used as packet marks. The colors are re-used so as to reduce the IP header overhead but at the same time we are uniquely able to traceback a router.

III. CONSTRUCTING ATTACK PATH IN STAR COLORED NETWORKS

Star coloring of a graph G is a vertex coloring of G such that no path of length three is bi-colored. It is also called distance-2 coloring. Star chromatic number of a graph G, denoted as $\chi_s(G)$ is the minimum number of colors which are necessary to star color G. Star coloring bounds the total number of colors to $[\Omega(d); O(d^2)]$ ([10]), where $d$ is the maximum node degree.

In general $d << n$ where $n$ is the number of nodes of the graph. In this section we explore the possibility of star coloring network topologies and using this information for IP traceback.

A. Basic Star Coloring

As in packet marking where a router is identified by some functions of its IP address, in the IP traceback approach using star coloring, each router is identified by a pre-assigned color. A router inserts its color to packets that pass through it. The attack path constructed by the victim is thus a sequence of colors. In star coloring due to the spatial reuse of colors, a color can occur more than once in a path. However, star coloring assures that from any particular node given a sequence of colors the path can be uniquely traced.

Figure 2 shows the star coloring of a graph. As our graphs are expected to represent network topologies, the graphs are assumed to be undirected. The figure appearing by the side of the nodes indicate their color. The figure shows a unique path (color sequence) viz. 1-2-3-1-2 from the victim to the attacker. Proper vertex coloring, the basic graph coloring scheme, cannot be used since there will be ambiguity during traceback. For example if we had used proper vertex coloring then node c and f can be assigned the same color say 3. In such a case from node b we cannot decide which path to follow as the two nodes c and f have same color. The spatial reuse of colors not only reduces the bit-space required for packet marks but it also ensures fewer packets for successful traceback since the entire router information is embedded in a single packet.

B. Attack Path Construction

A standard technique for attack path construction using packet marks is node sampling ([11]). In this technique all routers on the path of a packet mark the packet with probability $p$. The victim keeps a count of packet received from nodes on the attack path and sorts them. If a router is at a hop distance $d$ from the victim, the probability that the mark of the router reaches the victim will be given by $p(1 - p)^{d-1}$. This means the highest packet count will be that of the first hop node from the victim and so on. The number of packets received from the router closest to the attacker will be the least. In the traditional packet marking approach, marks of the routers are assumed to be unique. Thus an attack path will comprise of unique marks sorted in decreasing order with respect to their distance from the victim.

In star coloring, since the colors are reused, the same color can occur more than once in an attack path and thus directly
sorting the packets based on the count of their colors will result in erroneous path. In figure 2, if we consider the attack path 1-2-3-1-2 the victim will receive packets from node a and node d as well as from node b and e bearing same color information. Constructing the attack path by sorting the count of packet colors will result in the attack path 1-2-3. In this work we discriminate packets coming from different nodes but with same color by considering the distance of the nodes from the victim. The TTL field ([11]) is used during path reconstruction to keep track of the distance of the node from the victim. Coming back to our example given in figure 2, packets from node a and node d will both have color 1 but their TTL values will be different. One can argue that we can directly use the TTL field to construct the attack path but two nodes with the same color can also have the same TTL value if they lie on different paths.

A question which has not been addressed by previous packet marking techniques is how a victim will discriminate packets coming from a particular attacker. Most of the previous work assume that an intrusion detection system (IDS) will flag the attack packets. Even with such an assumption if there are multiple simultaneous attackers, which is most likely, attack path construction using node sampling will not work. The IDS will not only need to detect attack packets but also uniquely identify packets coming from different attackers. To overcome this problem we propose to imprint in each packet a fingerprint of the path followed by the packet.

In this work we propose to construct a path identifier (path id) to be embedded en route routers in the IP Identification field along with its color. The 16 bit ID field in the IP header is overloaded as shown in figure 3. A k-bit color field is used by nodes to mark their color. The remaining bits are used to store the finger print or path identifier. Ideally the path identifier is a summation of colors used on the path. However, such a proposition would require large bit space to store the path id. In order to restrict the size of the path id to that of the color field we propose to use XOR or ones complement arithmetic to compute the sum. The first router or the source stores a hash of its color in the path id field. The intermediate routers XOR its color with the already existing path identifier in the ID field to get the new id. Finally when the packet arrives at the victim, the mark is the XOR summation of color of each router in the path. This process is shown in figure 4. Since the process of creating the path id field is deterministic every packet traversing a particular path is marked with the same path identifier. The victim uses a combination of node color information, path id and hop count to construct the attack path.

An algorithm to construct the attack path is given in algorithm 1. The algorithm assumes that the IDS flags a packet as an attack packet. The algorithm extracts the path id and TTL value from the malicious packet. It checks each incoming packet for these two values collect them and finally sort the packets based on their color field. The main advantage of our path construction algorithm is that the IDS only needs to flag a packet as malicious. The victim can extract the path id from that field, collect all subsequent packets bearing that id and sort them to get the attack path. An additional advantage of using the path identifier is in filtering. Once a path id is flagged as malicious the victim can use the path id to drop all subsequent packets. However, the path id for an entire path is not guaranteed to be globally unique since two paths can have the same color.

C. Analysis of Attack Path Construction

The performance of our path construction algorithm depends upon the probability of collision of the attack path’s identifier with some other paths identifier. If we use addition of colors to construct the path id, two paths will collide only if number of occurrences of colors in both the paths is same. To find the probability of collision we first find the number of ways we can star color a path.

Lemma 1 (Total coloring scheme of a path). A path of length \( l \) can be colored using \( c (\leq l) \) colors in \( C_c(l) \) ways.

**Proof:** The number of ways we can color the path using \( c \) colors is a way of writing \( l \) as a sum of ordered sequence of exactly \( c \) positive integers. This is nothing but finding the composition of \( l \) into \( c \) parts ([13]) which is given as

\[
C_c(l) = \binom{l-1}{c-1}
\]
For example if we have a path of length 9 and 5 unique colors are used in the path the total number of ways we can color the path is given by

\[ C_5(9) = \binom{8}{4} = 70 \]

Out of the total 70 compositions, two possible compositions are \( \{3,2,2,1,1,1\} \) and \( \{5,1,1,1,1,1\} \). The first composition means in the corresponding coloring scheme the first color will appear 3 times, second and third color will appear 2 times and the remaining two colors will appear 1 time. In the second composition the first color will occur 5 times and all the other colors 1 time. The second composition is not valid for star coloring since in a 9 length path there is no 5 positions where we can assign same color. In a 9 length star-colored path, a color can occur maximum of 3 times. Thus from the total compositions we need to select valid compositions for star coloring. Let \( C_i^c(l) \) denote the way of writing \( l \) as a sum of ordered sequence of exactly \( c \) positive integers where \( i \) is the largest addend.

**Lemma 2 (Total star coloring scheme of a path).** A path of length \( l \) can be star colored using \( c \) colors in \(|S|\) ways where

\[
S = S_1 \cap S_2
\]

\[
S_1 = \{k_i : \sum_{i=1}^{c} k_i = l, k_i > 0\}
\]

\[
S_2 = \{k_i : \sum_{i=1}^{m} k_i = l, \left\lfloor \frac{l}{k_1} \right\rfloor \leq m \leq l, 1 \leq k_i \leq \left\lfloor \frac{l}{5} \right\rfloor\}
\]

where \( k_1 \) is the largest addend.

**Proof:** \( S_1 \) is a list of all compositions of \( l \) as a sum of exactly \( c \) parts. \( S_2 \) is a list of all compositions of \( l \) as a sum of positive integers where largest addend is not more than \( \left\lfloor \frac{l}{5} \right\rfloor \). The intersection of these two set will give all compositions which is a sum of exactly \( c \) parts and largest addend does not exceed \( \left\lfloor \frac{l}{5} \right\rfloor \).

For example if we have a path of length 5 and 3 unique colors are used in the path then any one color can be used maximum of 3 times i.e. \( \left\lfloor 5/3 \right\rfloor = 2 \) times. The set \( S_1 = \{3,1,1\}, \{1,3,1\}, \{1,1,3\}, \{2,2,1\}, \{2,1,2\}, \{1,2,2\} \) i.e. list of all compositions of 5 as a sum of exactly 3 parts(as 3 colors are available to color the path). Similarly the set \( S_2 = \{2,2,1\}, \{2,1,2\}, \{1,2,2\}, \{2,1,1,1\}, \{1,2,1,1\}, \{1,1,2,1\}, \{1,1,1,2\}, \{1,1,1,1,1\} \) i.e. list of all compositions of 5 as a sum of all positive integers where largest addend is not more than 2(as any one color can be used maximum of 2 times). So, \( S = S_1 \cap S_2 = \{2,2,1\}, \{2,1,2\}, \{1,2,2\} \). Now \(|S|=3\) and this is the count of valid compositions. Then the probability of having unique color combination in a path is

\[
\frac{1}{|S|}
\]

In the Internet the average length of a path is 18 ([14]). The average node degree of Internet graphs is less than 4 ([14],[23]). In section IV we show that the minimum number of colors required to star color such graphs will be 4. Consider a path length of 18 and 4 colors are used to color the path. Computing the compositions of \( S_1 \) and \( S_2 \) for this example will be huge. Hence we compute their partitions. The partitions of \( S_1 \) are \( \{15,1,1,1\}, \{14,2,1,1\} \ldots \{5,5,4,4\} \) and that of \( S_2 \) are \( \{6,6,5,1\}, \{6,6,4,2\}, \{6,6,3,3\}, \{6,5,5,2\}, \{6,5,4,3\}, \{6,4,4,4\}, \{5,5,5,3\}, \{5,5,4,4\} \). The number of compositions of each partition in \( S \) are \( \frac{4!}{2!4!}, \frac{4!}{2!4!}, \frac{4!}{2!4!}, \frac{4!}{2!4!}, \frac{4!}{3!4!}, \frac{4!}{3!4!} \) and \( \frac{4!}{4!4!} \) respectively. Hence total number of composition of \( S \) is 80. Each such composition will have a unique path id. Thus the path id of the attack path will collide with that of another path if there color combinations are same. The probability of the other path having the same color combination as that of the attack path will be 1/80 or 1.25%. However, in practice since we use XOR for addition, the path id of all the compositions will not be unique and hence the probability of collisions will be higher than this value. In our attack path construction algorithm we use both the path id and hop count, thus the probability of collision will be \( \frac{1}{|S|} \times \frac{\text{Hop count of attack path}}{\text{Hop count of all paths}} \).

IV. COLOR BALANCED STAR COLORING

A. Star Chromatic number of Network Topologies

The star coloring problem has been shown to be NP-complete for general graphs as well as for planar graphs ([121]). In this work we restrict ourselves to the star coloring of network topologies and try to find their star chromatic number. The star chromatic number of a ring network with 3 or 6 nodes is 3. Similarly the star chromatic number of a ring network with 4 or 8 nodes is 4. In general the star chromatic number of a ring network with \( l \) nodes is \((3+l) \mod 3\). The maximum degree of the graph (\( \Delta \)) has a strong influence on its star chromatic number. In a fully connected mesh each node will require a unique since all nodes are at distance less than 2 of each other. The star chromatic number for a fully connected network is therefore \((\Delta + 1)\). The star chromatic number of a bipartite graph is again \((\Delta + 1)\). A partially connected mesh topology with a even length cycle can be converted into bipartite graph and hence its star chromatic number is \((\Delta + 1)\). In a complete bipartite graph all the nodes are at distance less than 2 of each other hence the number of colors required to star color such graphs is equal to number of nodes or \((\Delta + m)\) where \( m \) is the number of nodes in the smaller set. In table I we list the star chromatic number of basic network topologies and their relation with \( \Delta \).

Our main goal in this work is to color the routers of the Internet. The structure of the Internet topology is not completely known and it is still an active area of research. Internet topology can be viewed either as IP-level(router-level) topology or AS-level topology. In AS-level topology each AS (Autonomous Systems) is considered a node and AS-peering as links. In the AS-level representation researchers...
viewed Internet topology as a graph and focused on its degree distribution. In router-level topology router represents a node and router-peering represents a link. In this work our aim is to color each and every router in the Internet and as such router-level representation of the Internet remains a reasonable choice. However, data on the router-level topology of the Internet can only be obtained indirectly (e.g., by inference from traceroute measurements), since network administrators don’t like to reveal details of internal topology. Whether the router-level topology follows power-law is an open question. Internet topology is very large, complex and constantly changing. A conceptual model of topology is hard to get. Internet is widely believed to be hierarchical by construction. To visualize the router level topology, several efforts have been made. In this work we consider several conceptual topologies that generate router-level topologies. The Random graph model ([16]) uses randomized algorithm for generating networks. Edges are added in a random and unbiased fashion. This model exhibits two of the commonly observed structural property of real-world large-scale networks. One is the presence of the single large or giant components and other is having small diameter. This model assumes the degree distribution is not heavy tailed but it follows binomial distribution. There is no high clustering coefficient and it is equal to the edge density. Real networks are not random and biases often exist. Like the random model, edges are selected randomly but with a bias i.e. connecting friends of friends. The model is nothing but a structural model [17] e.g. $\alpha$-model. This is a different way of getting high clustering, low diameter network. But it requires proper tuning to achieve all properties simultaneously. Such graphs are considered to be more realistic than the random model. Neither the random model nor the $\alpha$-model exhibits heavy-tailed degree distribution. So power law [17] or preferential attachment concept has been introduced. In this model rich get richer so the process amplifies inequality. This concept leads to heavy-tailed degree distribution but real life networks are not purely growing. None of these conceptual models describe the Internet topology completely and it is assumed that real Internet topology is a mix of these models. A significant conclusion after studying these models is that the Internet graph has heavy-tailed degree distribution. This means the Internet graph follows a hierarchical structure where core is less dense as compared to the edges. As $\Delta$ is very high due to the heavy tail, the maximum degree of the graph will dominate the star chromatic number of such graphs. Internet graphs will therefore be usually star colourable with $(\Delta + 1)$ colors. Although we do not give any mathematical proof, we tested our presumption by generating a number of topologies using topology generators ([18][22]) and found that they are all colourable with $(\Delta + 1)$ colors.

### B. Star Coloring Schemes

As in packet marking techniques one of our main objectives is to minimize the bit space required for star coloring. Thus we would always like to color a graph with its star chromatic number. We introduce here the notion of color balanced star coloring and show how a graph can be colored with its star chromatic number using this concept. A coloring scheme for a graph G refers to the process of assigning colors to all the nodes of G. There are several ways to color a graph.

#### Definition 1 (Color Balanced Star Coloring.)

Color coloring scheme of a graph G such that occurrence of the colors is balanced.

Algorithms for star coloring of graphs pick the first available color while coloring a vertex. Our desire is to distribute the available colors in a balanced fashion, i.e, each color should be used an equal number of times. Let, $G$ be a graph with $n$ vertices and $\chi_s(G)$ is $m$. We make the following observations:

#### Observation 1. In color balanced star coloring:

- each color will occur $\frac{n}{m}$ times if $n \mod m = 0$.
- $z$ colors will occur $\left\lceil \frac{n}{m} \right\rceil$ times where $n \mod m = z (> 0)$ and $(m - z)$ colors will occur $\left\lfloor \frac{n}{m} \right\rfloor$ times.

A graph $G$ is colorable with its star chromatic number may have different occurrence of the colors. For example the graph in figure 2 is colored with four colors ($1, 2, 3$ and $4$) which is its star chromatic number. However, in this scheme the occurrence of the colors is not balanced. Color 4 appears once, color 1 and 3 appear twice and and color 2 appears thrice. The same graph has again been star colored with its star chromatic number in figure 5 but in this case the occurrence of the colors are balanced.
Theorem 1. For any graph $G$, if it is colorable with its star chromatic number then one of its coloring scheme will be color balanced star coloring scheme.

Proof: Let number of nodes in $G$ is $n$ and $\chi_s(G) = m$. For simplicity let us assume $n \mod m = 0$. According to lemma 2 the set of valid composition or star coloring schemes possible will be given by the set $S$. Let $S'$ represent the corresponding partition for the set $S$.

$$S' = \{ k_i : k_i \in S, i = 1 \to m, k_1 \geq k_2 \geq \cdots \geq k_m\}$$

where $k_1$ is the largest addend for a particular partition and $\frac{n}{m} \leq k_1 \leq y$. The upper bound of $k_1$ i.e $y$ is graph dependant. In the simplest case, for line graphs, the upper bound $y$ will be $\frac{n}{2}$ whereas for other graphs its value will be lower. The value of $k_1$ will determine the values the other $k_i$'s will take for that partition. Each such composition will give a different coloring scheme which is again graph dependant.

However, for any graph $G$, when $k_1$ takes the least value of $\frac{n}{m}$ then $k_i$ for $i = 2 \to m$ will also take the same value else the condition $\sum_{i=1}^{m} k_i = n$ will not hold.

This means all colors will appear $\frac{n}{m}$ times which is nothing but the color balanced star coloring scheme.

Theorem 1 says that given the $\chi_s$ of a graph $G$, the graph is guaranteed to be colored using the color balanced star coloring scheme. Thus in all our proposed coloring schemes one of our goal is to balance the usage of colors. In the following sections we propose three techniques to star color a graph.

C. Balanced Star Coloring with Color Reassignment

A color balanced star coloring scheme is given in algorithm 2. Inputs to the algorithm are the uncolored graph $G$ and its star chromatic number. In other word the algorithm has a color palette of size $\chi_s(G)$. Here we assume that the star chromatic number of a graph can be estimated using the methods mentioned in section IV-A. The node traversal of the algorithm follows a greedy approach. The node with the highest degree and all its one hop neighbours will require a distinct color. Hence the algorithm starts traversal by selecting the node with the highest degree. The forbidden color list of a node is the set of colors which have been used in its first and second hop neighbours. The algorithm also maintains a variable Occurrence which count the number of times a color has been used. For the starting node its forbidden color list will be empty. A color (with minimum index) is assigned to the starting node and its occurrence is updated. From among the neighbours of the starting node, the algorithm selects the node with the highest degree for coloring. Each node computes its available color list which is formed by subtracting the forbidden color list from the color palette. From this available list, the node chooses a color whose occurrence is minimum and if there are more than one, it chooses the color with minimum index.

The process continues recursively until all nodes are colored or we arrive at a deadlock situation. We say that a deadlock has occurred if no colors are available to color a node. Whenever the algorithm runs into a deadlock, it first examines its first and second hop neighbours. For each node in this set, it computes its current available list. From this set it chooses a pair of two nodes which have a common available color and are at a distance greater than two. This process continues until a color becomes available for the blocked node. Since we know the chromatic number we are sure that a color is available for the blocked node and it can only be blocked by its first or second hop neighbours. Therefore at the end of the process we will get a color free for the blocked node.

Algorithm 2 Color Balanced Star Coloring with Color Reassignment

Input: Uncolored Graph $G$ and $\chi_s(G)$

Output: Color Balanced Star Colored Graph

1. currNode ← Node with highest degree
2. repeat
3. Assignment: From available color list chose minimum occurrence.
4. -assign color and update Occurrence.
5. Traversal: currNode ← Highest degree neighbor of currNode.
6. until Deadlock or All nodes assigned color
7. if (Deadlock(currNode)) then
8. $L \leftarrow \{(x,y) : x \text{ and } y \text{ are 1st hop neighbors and second hop neighbours of } currNode\}$
9. $M \leftarrow \{(u,v) \in L : \text{distance}(u,v) > 2 \text{ and they have different colors}\}$
10. Compute available color list for $u$ and $v$
11. if (there exists a common color) then
12. Assign the color to $u$ and $v$.
13. Update Occurrence
14. end if
15. if (freed color can be used on the currNode) then
16. GOTO step 3.
17. else if (Choose another pair $u$ and $v$) then
18. GOTO step 9
19. else
20. Increment color palette and GOTO step 3.
21. end if

1) Analysis of Algorithm 2: The number of colors required to star color a graph depends on the way we traverse the graph and how the colors are chosen. In this algorithm we try to balance the occurrence of colors. We have already shown that color balanced star coloring scheme is a valid color scheme if the star chromatic number of a graph is known. Since the size of the color palette is set to the star chromatic number of the graph, it is guaranteed that there is a color available for each node. Hence if the algorithm runs into deadlock it is because the traversal method was not optimal. A node will be deadlocked because its color has been used either in its first or second hop neighbour. Thus in lines 10 - 16 of the algorithm we try to extort this color by re-coloring the first and second
hop neighbours of the blocked node. The algorithm will fall to line 19 if the deadlock cannot be resolved. Such a case will occur if the estimate of our star chromatic number was not correct. In general we can start with $(\Delta + 1)$ colors since this is the minimum number of colors required to star color a graph and most network topologies have this chromatic number.

V. DEPLOYMENT ISSUES

A major challenge in deployment of IP traceback mechanism is not technical but administrative and political. There may be domains which are unwilling to deploy such tools due to the overheads involved. One may argue that a traceback mechanism can be deployed within an autonomous system to detect attacks originating from the domain or the entry point of an attack. However, the actual benefits of the scheme can be derived only if it is deployed throughout the Internet.

A robust IP traceback mechanism should be able to construct the attack path in the midst of non cooperative routers. If the network contains any non cooperative routers, that does not effect in the attack path construction. In our coloring scheme we assign colors to all the routers offline. However, some of the routers(non cooperative) may not follow the coloring scheme. So, information about these non cooperative routers will not be available to the victim. In such cases, the attack path construction will not be able to proceed beyond the non cooperative router.

A possible solution is to infer such non cooperative routers from the number of packets received along the path. For example our observation is that for an attack path of length 7, the number of packets received from each hop starting from the first one is 2, 5, 9, 21, 28, 39 and 46. Now if the third router is non-cooperative we will not receive the 9 packets from this router. The victim by looking at the count of packets received can infer that some router information is missing. However, for this the victim needs to keep an upper bound on the packet count of each hop. We need to do more experiments to better understand the problem and provide more conclusive results.

VI. RELATED WORK

In this section we give an overview of the reactive and proactive traceback mechanisms available in literature.

A. Link Testing

This is a reactive and manual method of traceback. In hop-by-hop tracing ([11]) the first step is to construct an attack signature that describes the common feature in each attack packet. This signature is then communicated with the network operator which then installs the appropriate filter in the victim’s egress router. The egress router reveals the input port of the attack traffic and hence revealing the router from which it is coming. This process is then repeated recursively on the upstream links until the origin or the Internet service provider’s boundary is reached. There are two main drawbacks of this approach. The first is constructing the attack signature will be non trivial since the IP address is spoofed and the attack packets have same characteristics as legitimate packets. The other important issue is the management overhead. Communicating with network operators across different ISPs is really cumbersome with no incentive to some.

In controlled flooding ([3]) the technique is to inject large bursts of traffic and observe the attack traffic’s behaviour using the map of Internet topology and iteratively flooding each router closest to victim. By observing the packet drop rate from attacker, victim infers from which link they arrived. This is then applied recursively to the upstream links. This process will not work for distributed attacks where the number of packets from each attacker is few. The other main drawback is that the technique itself is a denial-of-service attack.

B. IP Logging

The key concept in this approach is to log packets at key routers and then use data mining techniques to extract information of the attack traffic source. The main drawback of this approach is the potentially enormous resource requirements in terms of storage and amount of processing. To reduce the space requirement, logging on probabilistic sampling of packets could be done but still the amount of log can be huge.

A. Snoren et. al. ([4]) proposed a hash based IP traceback using logging. In the hash based approach the en-route routers produce packet digest of each departing packet and stores them in a digest table. When attack is detected, an attack graph is generated for each region by an agent. These attack graphs are then combined by a manager to produce the complete attack graph. The victim triggers the traceback process when the intrusion detection system (IDS) signals a potential attack and sends relevant information to the manager. The manager asks the agents to poll the routers in its region for digests. The agent collects all digest tables of its region, simulates reverse path flooding and if a packet is found, the corresponding node is marked. To remove collisions, multiple hash functions are used in every router. To minimize the storage requirement, bloom filter is used. Digests are stored locally for some period of time depending on resource constraints of the router.

The traceback process is very complex; the victim cannot directly trace the attacker. The other drawback is that since digests are stored for a particular time interval and they are discarded probabilistically as they age, finding the attacker would not be possible if the query falls beyond the time interval.

C. ICMP based Traceback

In this approach out-of-band messaging is used to achieve attacker tracing. The idea was originally proposed by Bellovin ([15]). As an IP packet passes through a router, ICMP Traceback message (ITrace) is generated. This ITrace message is then sent with equal probability either to the destination or to the origin of the IP packet. The message contains forward and backward router links, the traced packet and an authentication field. In the event of an attack, the end system can use it to traceback the attacker. The origin of the packet can use the information in the ITrace message to decipher reflector attacks. To minimize the overhead traffic the ITrace messages
are generated with a low probability. The main drawback of this attack is that path reconstruction is difficult since only one or two links’ information is sent through the initial router. An enhanced version called ICMP Traceback with Cumulative Path (ITrace-CP) has been proposed by Henry C.J. Lee et. al. ([6]). In this approach the ITrace message is sent to next hop router instead of destination. The en-route routers append the ITrace message to the ITrace-CP packet. Since the path information is encoded in each ITrace-CP packet victim takes less time and less number of messages to generate the attack path. This approach requires lot of space and is vulnerable to a number of attacks. The other drawback is that some of the routers may not support ICMP for security reasons.

D. IP Marking

In packet marking scheme, some traceback data (mark) is inserted in each packet so that a victim can use this information to identify the attacker ([7]). The victim can directly construct the attack path using these marks and no additional traffic is required. The assumption in packet marking is that the attacker produces sufficient packet and the attack path is fairly stable during the time of the attack. Thus one of the main challenges is to minimize the number of packets required for constructing the attack path. A general approach is to use the 16-bit identification (ID) field of the IP header to embed the marks. The second challenge is therefore to minimize the number of bits required to encode the address of the router. Depending on how the marks are encoded, packet marking techniques can be broadly classified as marks dependent or independent of IP address. In the first approach the marks are a function of the router’s IP address and in the second approach they do not have any relation with the IP address.

1) IP Address Dependent Packet Marking: The simplest approach of IP dependent packet marking is where each router appends its IP address to the record route option of the IP header. This method increases the packet’s length at each router and can lead to additional fragmentation at the downstream router, leading to an increase in network traffic. In node sampling only one router writes its IP address to the forwarded packet with some probability \( p \) [19]. After the victim has received samples from all the en-route routers, it can construct the attack path by ordering these unordered samples. However, this approach still has an overhead of 32-bit.

In deterministic packet marking ([18]), only the edge routers participate in the marking procedure. The ID field is used to carry one half of an IP address and the RF bit is used to denote whether it is the first half or the second half of the IP address. Although this approach reduced the space overhead, it had a serious consequence since constructing an IP address from fragments would require trying out all the possible permutations. The scheme was modified by also sending a hash of the IP address along with the fragmented IP address to aid the victim in the reassembly procedure. However sending an additional hash increases the space overhead and the number of fragments now increases.

To minimize the space overhead Barua et. al. proposed to use a 16-bit hash of the router’s IP address (called the HashMark). The HashMark and the list of all routers that have this HashMark are mapped together in a table for later lookup. The advantage of this approach is that the time for reconstruction is considerably reduced and a single packet is sufficient to detect and traceback an attack. However, by encoding the 32-bit IP address in a 16-bit field, DERM introduces the probability of collisions and thus false positives.

In Path Identification (Pi) ([9]) the router marks the last \( n \) bits of its IP address in the IP ID field of the packets it forwards. It is concerned with reconstructing a path from a victim to an attacker; rather, it is concerned with identifying paths with unique markings. The Pi mechanism can also be used to detect spoofed IP addresses, with an appropriate filter. The victim needs only to build a table correlating the Pi mark of a packet to its source IP address, during a non-attack time. When under attack, the victim can check to see if the source IP addresses of incoming packets match against the IP addresses of their Pi marks from the table.

2) IP Address Independent Packet Marking: The main idea is to reuse the colors so that the number of colors and thus the bit space needed to represent them is minimized. At the same time the colors are assigned in such a way so that there is no collision during attack path construction. The problem is thus reduced to a graph coloring problem. Star coloring ([10]) is a general technique to color a network graph for IP traceback. However, there are no techniques available in literature that state how colors can be used for traceback. In this work we have proposed a scheme to star color a network. The main highlight of our scheme is that we have tried to optimize the use of colors. We also introduced the concept of using path identifier (path id) along with the colors(marks). The use of path identifier makes the task of path construction simple and efficient for the victim.

VII. Conclusion and Future Work

In this paper we have presented an IP traceback mechanism for star colored networks. We introduced the concept of path identifier which allows us to more or less uniquely identify a path. We have shown mathematically that the probability of two path identifiers colliding is very small. The use of path identifier makes the task of path construction simple and efficient for the victim. The highlight of our proposal is that it requires the intrusion detection system to flag a single packet as malicious. In the second part of our work we present color balanced star coloring scheme where the occurrence of colors is balanced. We show that this coloring scheme is one among the multiple ways to color a network with its star chromatic number. Next we proposed an algorithm that implements color balanced star coloring scheme. The coloring scheme that we have proposed is an offline scheme. A possible future direction of the work is to make this coloring scheme online which will make it a practically more feasible approach. Another possibility is to explore the use of the colors in attack classification and filtering.
REFERENCES


[22] [online] Modeling Topology of Large Internetworks http://www.cc.gatech.edu/projects/gtitm/
