Secure Routing Protocols Using Consistency Checks and S-RIP *

Tao Wan † Evangelos Kranakis † Paul van Oorschot †

Abstract

Internet routing infrastructures are vulnerable to various attacks due to the lack of strong authentication mechanisms, software vulnerabilities/misconfiguration, and the risky assumption of a trustworthy and cooperative environment. Existing solutions do not solve the problem because they neither validate factual correctness of routing updates nor support incremental deployment. In this paper, we propose a data correlation approach for validating routing information. A routing update is validated for its factual correctness before being used to update a routing table by cross checking its consistency among selected nodes which are informed of that update. The notion of trust or distrust is replaced by node reputation measured by numerical values. The tradeoff between security and efficiency is made by configurable thresholds and a sized window which determines how many nodes to involve in a consistency check. As a first example of applying the framework, we develop an incrementally deployable protocol, namely (S-RIP), for securing Routing Information Protocol (RIP). We have implemented S-RIP in the network simulator NS2. We show that with S-RIP, a nonfaulty node can uncover inconsistent routing information in a network with many misbehaving nodes given that no two of them are in collusion. Additional routing overhead generated by S-RIP is adjustable and can be reduced to a reasonalbe level.

1 Overview

It is well-known that today's Internet is not secure. Both Internet applications and the underlying routing infrastructures are vulnerable to a variety of attacks. Although a majority of incidents reported so far are realized by the exploitation of software vulnerabilities in client and server machines, it has been noted that abusing routing protocols may be the easiest way for launching attacks [2]. Perlman [29] pointed out that a single misbehaving router can completely disrupt routing protocols and cause disaster. This viewpoint has been more recently expressed by a group of network and security experts [7].

There are many factors that make today's routing infrastructures insecure. Three of them are as follows. First, there are no strong security services built into routing protocols. Many routing protocols only provide weak authentication mechanisms, e.g., plain-text password or system-wide shared keys, for authenticating peers or routing updates. As a result, it is easy for an adversary to gain access to the routing infrastructure and manipulate routing information. It is also easy for an insider to impersonate others. Second, software vulnerabilities and misconfigurations expose routing infrastructures to severe risks. Third, most routing protocols assume a trustworthy environment. In the case where no authentication mechanisms are implemented, routing updates are accepted only with rudimentary validation – for example, RIP [21], one of the most popular distance vector routing protocols, only checks that a routing update is from an IP address of a neighbor node and that the source UDP port number is 520. When authentication mechanisms are present, routing updates are verified for the correctness of data origin and integrity only. However, after a route update is verified to be "authentic", the routing information conveyed in the update is trusted and used to update the recipient's routing table. This is risky since data origin authentication which includes data integrity [23], cannot guarantee the factual correctness of a message. A malicious entity or a compromised legitimate entity can send false information in a correctly signed message. This is one of the byzantine failures mentioned by Perlman already in 1988 [28]. A recipient can detect unauthorized alteration of the message, but

*Version: August 30, 2003. For the latest version, please contact us.

†{twan, kranakis, paulv}@scs.carleton.ca. School of Computer Science, Carleton University, Ottawa, Canada.

cannot tell if the information conveyed in the message is factually correct unless the recipient has the perfect knowledge of what it expects to receive. For example, a malicious node can claim a longer distance [15] to avoid traffic without being detected unless the receiving node has the correct information of the network topology. In summary, cryptographic mechanisms can prevent several types of attacks, e.g., impersonation or unauthorized modification, but may not be able to prevent propagation of fraudulent information about network topology or network connectivity.

This paper focuses on validating routing advertisements for distance vector routing protocols. The difficulty of validating such information was noted by Perlman [28]. The problem arises due to the fact that the advertised information is the result of distributed computation. Smith, Murphy, and Garcia-Luna-Aceves [35] propose to include an additional field, next-to-last-hop, in routing advertisements. A loop-free path finding algorithm is used to detect routing loops, but they do not address prevention of longer or shorter distance fraud. Mittal and Vigna [24] propose to use intrusion detection sensors for validating routing advertisements by comparing a routing update with a master routing database that is pre-computed off-line. One disadvantage of this intrusion detection based approach is that it cannot prevent fraudulent misinformation from poisoning others' routing tables, although it may be able to detect it.

1.1 Summary of Results

We propose a framework for validating routing information for distance vector routing protocols. First, we propose to use *consistency* as an approximation of *correctness.* Given the difficulty of validating routing information of a distance vector routing protocol, we propose to correlate an advertised route among a group of nodes that are informed of that route. The confidence in the correctness of the route is increased if it is consistent in the corroborating group. The more nodes that agree, the higher the confidence. By this approach, we hope that nodes surrounding a *questionable* node will uncover inconsistency and prevent further spreading of misinformation.

Second, we propose to *validate* an advertised route for correctness before using it to update a routing table. The validation protocol could be implemented as a wrapper of routing table update functions. When a routing table needs to be updated, the validation protocol is triggered. In contrast to the above intrusion detection approach, our approach can prevent fraudulent routing information from propagating and poisoning routing tables.

Third, the notion of either trusting or distrusting a node is replaced by *node reputation* measured by a numeric value. Fully trusting or distrusting any individual node may introduce the vulnerability that a malicious node can call into question the legitimacy of other nodes. It is questionable if any node should deserve the full trust of any other node given that either the owner of a node may be malicious or a legitimate node could be compromised. We employ numeric values for measuring node reputations to provide the flexibility for relaxing this notion.

Fourth, we use a *sized window* for balancing security and efficiency. A node may get maximum confidence in the correctness of a particular advertised route if every informed node with respect to that route is involved in the consistency check and they agree. However, this incurs high verification overhead in terms of the number of verification messages required. As is often the case, mechanisms are needed to obtain a tradeoff between *security* and *efficiency*. In our approach, a window size, or the number of nodes involved in a consistency check, is determined by node reputations and configurable thresholds.

As a first example, we apply our reputation-based validation framework to securing RIP. We call our protocol Secure-RIP (*S-RIP*). The rest of the paper is organized as follows. The reputation-based validation framework is presented in Section 2. *S-RIP* is presented and analyzed in Section 3. Section 4 presents simulation results. Section 5 reviews related work for securing routing protocols, with emphasis on securing distance vector routing protocols. Further discussions and future work are given in Section 6.

2 Reputation-based Validation Framework

2.1 Consistency Check

To validate the correctness of routing information, ideally there would be a *master* node (cf. [24] above), with perfect knowledge of a network. Such knowledge could then be distributed to other nodes periodically or on demand. To validate a routing advertisement from an advertising node, a recipient node would only need to check its consistency with the true network topology from the master node. Unfortunately, there is no such master node in today's Internet architecture. Otherwise, there would be no need for dynamic protocols for updating routing tables. Since it appears impossible to validate the factual correctness of routing information without a master node, we propose using consistency to approximate correctness. This approach is motivated by the fact that network topology information conveyed by a routing advertisement may already be known to a subset of nodes in the network (namely *informed* nodes of that particular route). By cross-checking the consistency of a routing advertisement with those informed nodes, a recipient can increase its confidence in the correctness of that information. By this approach, we hope that direct neighbors of a *questionable* node will uncover inconsistency and prevent further spreading of misinformation.

Using inconsistency to detect misinformation in routing protocols is comparable to the approach of data correlation for detecting node capture in a distributed sensor network [42]. For example, a sensor dropped in desert for measuring temperature may be compromised. An attacker can manipulate the output of the captured sensor with or without decoding its cryptographic information (e.g., by cooling it with water). To detect such captured sensor, we need to correlate the output reported by sensors geographically located nearby each other. If the output of a sensor is significantly different from others, it may indicate that the sensor is in a questionable state. We observe that routing information correlation appears to allow for discovery of malicious routing updates that cannot be detected by message authentication.

2.2 Reputation Definitions

Since fully trusting or distrusting a node has shortcomings (as mentioned earlier), we propose a method for computing a value which is assigned and used as a measure of reputation of an individual node. Every node will keep a value as a measure of the reputation of every other node in a network (e.g., a RIP domain).

Definition 1 *(***Node Reputation***). Node i's rating of node j's reputation at time t_m, denoted by $r_i(j, t_m)$ is a measure of the historical correctness of routing information provided by j to i, defined as*

$$r_i(j, t_m)^1 = \sum_{t=1}^{t_m} c_i(j, t) \cdot w(t) \tag{1}$$

where $c_i(j, t)$ is a value assigned by i, associated with j, based on the result of i's determination of the correctness of j's information at time slot t, and $w(t)$ is a system-wide time weighting factor. $r_i(j, t_m)$ is computed based on the following two factors:

1) Historical Behaviour. Suppose the time unit is the interval between two consecutive consistency checks. Node j's historical behaviour in the viewpoint of i can be represented by a vector $[c_i(j, 1), c_i(j, 2), \ldots, c_i(j, t_m)]$. Many possibilities exist for $c_i(j, t)$; one we propose for its simplicity is:

$$c_i(j, t) = \begin{cases} 0 & \text{if } j \text{ provides incorrect information at time } t \\ 0.25 & \text{if } j \text{ provides conflicting information at time } t \\ 0.5 & \text{if } j \text{ provides consistent information at time } t \end{cases} \tag{2}$$

2) Time Weighting Factor. The time weighting factor $w(t)$ allows a differential weighting for different time periods. As one simple example, Equation 3 gives a higher weight to a more recent consistency check,

[1] When the parameter t_m is omitted from $r_i(j, t_m)$, $r_i(j)$ refers to i's rating of j's reputation at the most recent time, t_m.

motivated by the belief that a node's most recent behaviour can predict its near future behaviour with a higher probability:

$$w(t) = \frac{1}{2^{t_m - t}} \quad 1 \leq t \leq t_m \tag{3}$$

One property of Equation 3 is that it allows one to compute a more recent reputation from the previous one, which reduces both computational overhead and the memory requirement. For example, $r_i(j, t+1)$ can be computed from $r_i(j, t)$ by Equation 4. One property of Equation 4 is that if $r_i(j, t) \neq 1$, $r_i(j, t+1)$ will be always less than 1. Thus, if node i does not assign an initial value of 1 or higher as j's reputation, $r_i(j)$ will always be in the range $[0, 1)$.

$$r_i(j, t+1) = \frac{r_i(j, t)}{2} + c_i(j, t+1) \tag{4}$$

One node's reputation can be viewed as an estimation of the confidence that this node will provide correct information in the near future. The higher $r_i(j)$ is, the greater confidence i has in information provided by j.

Definition 2 *(**Accumulated Confidence**). Let $r_x(v_1), r_x(v_2), \ldots, r_x(v_n)$ be x's rating of the reputation of nodes $v_1, v_2, \ldots, v_n$, respectively. In the case that routing information from nodes $v_1, v_2, \ldots, v_n$, is consistent, node x's confidence in that information, denoted by $r_x(v[1..n])$, is defined as follows, where $v[1..n]$ denotes $v_1, v_2, \ldots, v_n$:*

$$r_x(v[1..n]) = \begin{cases} r_x(v_1) & \text{if } n = 1 \\ r_x(v_1) + \big(1 - r_x(v_1)\big) \cdot r_x(v_2) & \text{if } n = 2 \\ r_x(v[1..n-1]) + \Big(1 - r_x(v[1..n-1])\Big) \cdot r_x(v_n) & \text{if } n > 2 \end{cases} \tag{5}$$

The rational behind Equation 5 is that one's confidence in the correctness of an advertised route is increased when the number of nodes confirming the route is increased. The amount by which the confidence will be increased depends on the confidence room left over and the reputation of the confirming node. Equation 5 has the following properties: 1) If the reputation of a confirming node is 0, it will not raise the accumulated confidence – in other words, information from a distrusted node will be disregarded; 2) If the reputation of a confirming node is 1, it will raise the accumulated confidence to 1; 3) The order of confirming nodes makes no difference.

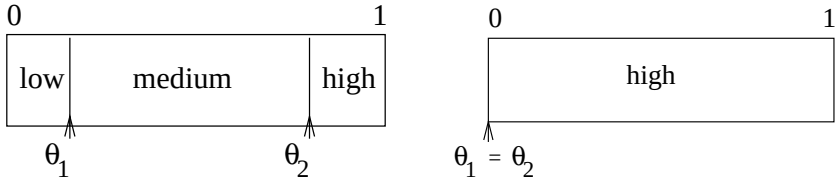
Although developed independently based on our intuition, it turns out that Equation 5 is consistent with Dempster-Shafer theory (DST) of evidence reasoning [8, 33] if we assume that in our case, for all i ($1 \leq i \leq n$), v_i acquires its information from an independent source. The advantage of Equation 5 is that it is intuitive and computationally efficient. Although Dempster-Shafer theory is more general, e.g., it can handle conflicting information, it is computationally less inefficient since it involves set operations.

2.3 Validation Rules

Since the notion of trust or distrust has been replaced by node reputation, we propose a set of rules for determining how to process routing advertisements based on node reputation. Two thresholds (θ_1, θ_2) are used to divide the reputation domain into three levels, namely low, medium, and high (Figure 2.3). Each reputation level has its own rules for processing routing update messages.

Rule 1 *(**Low Reputation**). If node j's reputation rated by i is in the low range ($0 \leq r_i(j) < \theta_1$), node i will ignore a routing advertisement from j without cross-checking its consistency with other node(s).*

According to Rule 1, a node with a low reputation is *untrusted.* This rule can effectively mitigate potential denial of service attacks launched by a malicious node which may try to engage another node into a long period of validation by advertising a large amount of useless routing information. We propose to timeout a node's low reputation and reassign it a medium reputation value to allow a node to raise its reputation after a specified time period P_1.



(a) Partially Trusted Network (b) Fully Trusted Network

Figure 1: Moving θ_1 and θ_2 close to each other increases trust degree and decreases network overhead. The extreme case, where $\theta_1 = \theta_2 = 0$, emulates a network in which all routing information is assumed to be fully trustworthy, which is in fact the assumption (at some risk) made for today's Internet.

Rule 2 *(***Medium Reputation***). If node j's reputation rated by i is in the medium range ($\theta_1 \leq r_i(j) < \theta_2$), node i will cross check the consistency of a routing advertisement from j with other node(s).*

A node with a medium reputation can be said to be *on probation.* If every node initially rates every other node a medium reputation, Rule 2 allows a good node to raise its reputation quickly into the high range. The reputation of a misbehaving node will quickly decrease into the low range. Therefore, the validation overhead is lowered if the network remains stable.

Rule 3 *(***High Reputation***). If node j's reputation rated by node i is in the high range ($\theta_2 \leq r_i(j) \leq 1$), node i will accept a routing advertisement from j without cross checking its consistency with other node(s).*

By Rule 3, we can say that a node with a high reputation is *trusted.* One disadvantage is that fraudulent information from a trusted node may be propagated since it will not be validated. To minimize the risk, we propos that a node maintain a high reputation for only specified period of time (P_2). After that period, its reputation is re-initialized with a value in the medium range. The risk window of accepting malicious information from a compromised or failed trusted node can be reduced by decreasing P_2. Another way to minimize this risk is to increase θ_2. The higher θ_2 is, the longer a node will take to raise its reputation into the high level. In the extreme case, θ_2 can be set to 1 so that no node will be trusted during the course of operation unless its initial reputation is set to 1.

We can emulate a trustworthy environment by setting both θ_1 and θ_2 to 0 (Figure 1.b), where every node is trusted by every other node and no routing advertisements will be validated. Validation overhead can also be reduced by adjusting the reputation thresholds (Figure 1.a).

2.4 Sized Windows

Since there may be multiple informed nodes of an advertised route, a mechanism is required to decide how many nodes to involve in a consistency check. The more nodes consulted (which agree with the the advertised route), the higher the confidence acquired in the correctness of that route; but the network overhead will also be higher. We use a *sized window* as a mechanism for balancing the trade-off between security and efficiency. The size of the window is the number of the nodes consulted in a consistency check. The window size starts from 1. In other words, there is only one node in the window before the consistency check of an advertised route, which is the advertiser of that route. The window size grows by one, or an

additional node is consulted, if the computed confidence using Equation 5 in the correctness of that route is less than θ_2. The window size keeps growing for the advertised route until 1) an inconsistency occurs, i.e., a node reports conflicting information; or 2) all the nodes in the window agree upon the route, and 2.1) the computed confidence is greater than θ_2; or 2.2) all informed nodes have been involved. In case 1), the route fails the consistency check and is dropped. In case 2), the route succeeds the consistency check and is accepted.

3 Secure Routing Information Protocol (*S-RIP*)

3.1 Assumptions and Notation

Cryptographic Assumptions. We assume (A1) every router shares a secret key with every other router in a RIP domain. Pair-wise shared secret keys can prevent router impersonation. This assumption is an extension to the security requirement of RIP, where a secret key is required for each network link if MD5 authentication is applied [1]. We assume additionally (A2) every router shares with every other router in a RIP domain a secret key for each of the subnets directly attached to that router. This assumption is made as a countermeasure to subnet impersonation. Router i claiming a zero distance to a subnet (e.g., N_1) has to demonstrate to a recipient (j) the knowledge of the shared secret key between j and N_1.

In RIP (we means RIPv2 in the remainder of the paper), a destination could be a subnet or a host. For convenience purposes, we use the identification (e.g. address) of a router to represent all the subnets or hosts directly attached to that router. Such an abstraction is widely used in studies of routing protocols [28, 21]. With this abstraction, our cryptographic assumptions become pair-wise shared secret keys.

Network Assumptions. We assume (A3) each network link is bidirectional, and has an equal cost for each direction. We also assume (A4) that every node includes the immediate next hop in an advertised route. In RIP, the next hop field in a routing advertisement is used for route optimization but is not mandated. If it is set to 0, the originator of the advertisement will be used as the next hop by a recipient. If it is set to an IP address that is directly reachable by the recipient (i.e., on a subnet directly attached to the recipient), this address, instead of the originator, is used as the next hop by the recipient. We require that a node always includes the next hop when advertising a route. This allows a recipient to find informed nodes of, and perform consistency check for, that route by recursively requesting routing information from next hops. It also allows a recipient to make the decision whether the advertised next hop or the advertiser will be the next hop for that route from its viewpoint. It appears that in RIP such a decision is primarily made by a sender since the sender has the flexibility to set a next hop either to 0 or to an IP address. We believe that a recipient is in a better position to make such a decision since it can easily check if the IP address of the received next hop is directly reachable. A sender may not have such information. Besides route optimization, the next hop allows a router to construct a complete path to a destination. This is useful for diagnosis and detection of misconfiguration, e.g., routing loops or malicious packet dropping [27].

Notation. For an advertised route $[dest, dist, nh]$, we use v_0, v_1, and v_n to represent the recipient, the advertiser, and the ultimate destination respectively. To be more specific, we use $dist(v_1, v_n)$ and $nh(v_1, v_n)$ to represent the distance and the next hop respectively from v_1 to v_n for this particular route.

3.2 *S-RIP*

When router v_0 receives from v_1 an advertised route $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$, v_0 validates the route as required by RIP [1]. If the route passes the validation, and will be used to update v_0's routing table, *S-RIP* is triggered to perform additional validations. *S-RIP* will NOT be triggered if the advertised route does not indicate a route change or a topology change. Although the timer associated with this route will be re-initialized, there is no need to re-validate the route since such a validation should have been done when the

route was first installed in v_0's routing table. Highlights of *S-RIP* on validating $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$ are given immediately below. More details are presented in the remainder of this section.

1. Is the advertised route self-consistent? If not, drop the route.
2. If $dist(v_1, v_n) = 0$, v_0 performs entity authentication with v_1 on v_n. If v_1 successfully demonstrates the knowledge of the secret key, k_{v_0,v_n}, shared by v_0 and v_n, v_0 accepts the route. Otherwise, drops it.
3. If $1 \leq dist(v_1, v_n) < 15$, v_0 performs consistency checks on $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$. If the consistency check succeeds, v_0 accepts the route. Otherwise, drops it.
4. If $dist(v_1, v_n) \leq 16$, v_0 accepts the route without validating it.

Self-consistency Check. v_0 checks if the information contained in $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$ is self-consistent. 1) If v_n is not a legitimate entity, the route is dropped. v_n is legitimate only if v_0 shares a secret key with v_n. 2) If $dist(v_1, v_n) = 0$, $nh(v_1, v_n)$ should be v_1 itself since the advertised route is for v_1 or a subnet directly attached to v_1. 3) If $1 \leq dist(v_1, v_n) < 15$, the next hop must not be v_0 or v_1. v_1 should not advertise a valid route back to v_0 from which it learns that route. Otherwise, the problem of counting to infinity occurs. Although RIP recognizes this problem and proposes split horizon (or with poisoned reverse) for solving it, a misbehaving node may not follow the rule and intentionally create the problem. By validating the next hop, a recipient can always avoid the problem.

Entity Authentication. If $dist(v_1, v_n) = 0$, v_1 advertises to v_0 a route for itself or for a subnet which is directly attached to v_1, v_0 will ask v_1 to demonstrate the knowledge of the secret key, k_{v_0,v_n}, shared by v_0 and v_n. The messages exchanged by v_0 and v_1 are illustrated in Table 1, where r_0 is a random number, and $h()$ is a one-way hash function, e.g., MD5 [32]. If v_1 succeeds in the entity authentication, v_0 accepts the advertised route. Otherwise, discards it.

$v_0 \rightarrow v_1$	r_0
$v_0 \leftarrow v_1$	$h(r_0, v_0, k_{v_0,v_n})$

Table 1: Entity Authentication

$v_0 \rightarrow v_2$	$[v_n, *, *]$ $[v_1, *, *]$
$v_0 \leftarrow v_2$	$[v_n, dist(v_2, v_n), nh(v_2, v_n)]$ $[v_1, dist(v_2, v_1), nh(v_2, v_1)]$

Table 2: Routing Request and Response

Consistency Check. If $1 \leq dist(v_1, v_n) < 16$, v_1 advertises to v_0 a reachable route for v_n. v_0 will check the consistency of the advertised route with $nh(v_1, v_n)$, let's say v_2. v_0 will request from v_2 the routing information from v_2 to v_n and from v_2 to v_1. The message flows are given in Table 2, where * denotes information fields to be provided. The advertised route from v_1 for v_n is treated as consistent with v_2's routing information if $dist(v_1, v_n) = dist(v_2, v_n) + dist(v_2, v_1)$. Otherwise inconsistent.

If v_1 is consistent with v_2, v_0 will use Equation 5 to compute an accumulated confidence, $r_{v_0}(v_1, v_2)$. If $r_{v_0}(v_1, v_2) \geq \theta_2$, v_0 accepts the advertised route as correct. Otherwise, v_0 will consult with additional nodes based on the next hop information. Before v_0 sends a route request to node v_i, it checks if a network loop has been formed. A network loop is formed if the node (v_i) to be consulted has been consulted before. In the case that a loop is detected, v_0 drops the advertised route. Otherwise, the consistency check continues until one of the following three conditions holds: 1) $r_{v_0}(v[1..k]) \geq \theta_2$. In this case, the advertised route from v_1 is treated as correct by v_0. 2) $r_{v_0}(v[1..k-1]) < \theta_2$, and v_k disagrees with v_{k-1}, i.e., $dist(v_{k-1}, v_n) \neq dist(v_k, v_n) + dist(v_k, v_{k-1})$ In this case, v_0 treats the advertised route as inconsistent. 3) v_n has been consulted. If v_n disagrees with v_{n-1}, the advertised route from v_1 is treated as inconsistent. Otherwise, v_0

will ask v_n to demonstrate the knowledge of the secret key k_{v_0,v_n}. If v_n succeeds the entity authentication, the advertised route is treated as correct no matter what the value of $r_{v_0}(v[1..n])$ is. Otherwise, the advertised route is dropped and v_n is treated as providing cryptographically incorrect information.

It is possible that v_0 may not have a route to v_i in its routing table when it needs to send routing requests to v_i. For example, the route being validated by v_0 is for v_i itself (i.e., $v_n = v_i$). It is also possible that v_i may not have a route to v_0 when it needs to send back routing responses. Therefore, we require that v_0 should send the routing request directly to v_1 with v_i as the ultimate destination. v_1 should have a route for v_i since it advertises such a route to v_0. In addition, v_0 should enable the flag of routing record in the IP header of the routing request message. As a result, all the intermediate routers forwarding the routing request message will record their addresses in the packet. v_i will reverse the recorded route, and use source routing for sending back the routing response to v_0.

Infinity Route. If $dist(v_1, v_n) = 16$, v_1 advertises to v_0 an infinite route for v_n. v_0 does not validate an infinite or unreachable route since it is trivial for v_1 to make a valid route unreachable if it misbehaves, e.g., by disabling a network interface or dropping packets. The consequence of such possible misbehaviour is that v_0 will drop the route and will not forward packets to v_n through v_1. If there is only one route in the network from v_0 to v_n and it goes through v_1, v_0 will not be able to communicate with v_n. It seems to be hard to force a misbehaving node forward packets for others if it is determined not to do so. Therefore, we hope a network is designed with redundancy to accommodate a single point of failure. In that case, hopefully v_0 could find an alternative route to v_n, bypassing the misbehaving node v_1.

3.3 Threat Analysis

A node may misbehave in several ways: 1) advertising false routing information; 2) providing false routing information specifically during a consistency check; 3) dropping a validation request/reply message or not responding to a validation request. 4) manipulating a validation request/reply message originated from other nodes.

1) *Advertising false routing information.* Given a route $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$ advertised by node v_1 to v_0, v_1 may provide false information about v_n, $dist$, nh, or any combination.

1.1) *Destination Fraud.* v_1 may advertise a route for a nonexistent or unauthorized destination v_n. Under our proposal, such misbehaviour can be easily detected since v_0 does not share a secret key with v_n if it is not a legitimate entity in the network.

1.2) *Distance Fraud.* v_1 may advertise a fraudulent distance to a destination v_n, e.g., longer or shorter than the actual distance. If the advertised distance, $dist(v_1, v_n)$, is 0, but v_1 is actually one or more hops away from v_n, in our proposal, v_0 can detect this fraud by entity authentication since v_1 does not have the knowledge of the secret key shared by v_0 and v_n. Other shorter or longer distance fraud can be detected by cross checking consistency with those nodes which propagated the route in question. There are three scenarios in which a consistency in the corroborating group may not represent correctness: a) the nodes in the corroborating group are simultaneously misled by one or more misbehaving nodes; b) the nodes in the corroborating group are colluding; c) a subset of the corroborating group are colluding and mislead the rest of the nodes. Our idea is that by increasing the size of the corroborating group, it is increasingly unlikely that these scenarios will not be detected.

1.3) *Next Hop Fraud.* Node v_1 may provide a fraudulent next hop to support its claim of a longer or shorter distance. First, v_1 may use fictional nodes as next hops. v_1 then intercepts from v_0 the subsequent validation requests to these nodes and send back false responses on behalf of them. In our scheme, a fictional node can be detected since v_0 does not share a prior secret with it. Second, v_1 may use a remote node (i.e., a node not directly connected to v_1) as the next hop. For example, suppose v_1 is 5 hops away from v_n. If v_1 learns that v_m is one hop away from v_n, it may claim to be two hops away from v_n and use v_m as the next hop. Unless v_m is willing to provide false information (e.g., $dist(v_m, v_1) = 1$) to cover v_1's misbehaviour,

v_0 will be able to detect this fraud. In the case that v_m is willing to collude with v_1, we treat it as the case that v_1 establishes a virtual link (e.g., TCP connection) with v_m, and forwards packets over the virtual link to each other. This misbehaviour is similar to the *wormhole* attack (cf. Hu et al. in [18]).

2) *Providing false routing information* in a consistency check. The fraud could be on distance or next hop. When the false information cause inconsistency, the consequences are: 2.1) correct routing advertisements may be disregarded by well-behaved nodes. We think it is not to the advantage of a misbehaving node to mislead another node by this type of misbehaviour since it may be best to avoid a "valid" route through a misbehaving node in any case. By dropping a route involving a misbehaving node, the validation node may take an alternative good route, albeit possibly suboptimal. 2.2) the reputation of a well-behaved node may be decreased as a result of false information arising from a misbehaving node. In the worst case, if node v_0's rating of node v_1's reputation is decreased to the low range, v_0 will disregard v_1's routing advertisements for a certain period of time. Since consistency check occur only on route changes, a misbehaving node, v_m, may only damage the reputation of v_1's reputation when there is a route change which involves both v_m and v_1 in a consistency check. v_m's own reputation may also be decreased if it provides false information. Therefore, v_m is unable to damage another node's reputation at its will. On the other hand, v_1 has other chances to increase its reputation when it advertises good routes (without going through v_m) to v_0. So the effect of the type of misbehaviour depends on the network topology and the location of the misbehaving nodes. If one or more misbehaving nodes are located on the links which can form a network-cut, they may be able to completely separate the network through collusion. It would appear no approaches resilient to such misbehaviour.

3) *Dropping a validation request/reply message or not responding to a validation request.* This misbehaviour can disrupt a validation process. As a result, the route being validated will be dropped. We do not consider this as a major drawback since dropping a route with misbehaving nodes en route allows an alternative route to be discovered. An adversary may launch this type of attack when it is not willing to forward packets for other nodes. As discussed before, a misbehaving node can avoid traffic by many other ways, e.g., dropping packets based on source or destination addresses, or simply disabling a network interface. We rely upon network redundancy and other mechanisms [27] to counter this type of misbehaviour.

4) *Manipulating a validation request/response message* originated from other nodes. If all routers are deployed with *SRIP* and use MD5 for message authentication, validation request/response messages cannot be manipulated en route. However, communication between a secured router and a remote nonsecure router is not authenticated. The consequences are: 4.1) A routing response sent back by a remote non-secured router can be modified by an adversary en route. The adversary may modify the routing response in such a way that it would confirm the consistency of a false advertised route; 4.2) An adversary may intercept routing requests sent to a non-secured router, and produce false responses on behave of that router. This vulnerability can be addressed by IP layer security. For example, if IPSec is available, an adversary would not be able to manipulate or intercept routing requests or responses between two remote nodes. It can also be mitigated if we assume that an adversary does not have the capability to launch attacks in packet level. It is easy for an adversary to manipulate a routing table to make a router to broadcast fraudulent routing information. It may not be that easy to manipulate packets transmitted through a router if the adversary does not have sufficient control over that router, e.g., modify and compile source codes, install malicious software, etc.

One characteristic of *S-RIP* is that it does not guarantee that a validated route is optimal. In fact, *S-RIP* proposed in this paper only validate route consistency, without considering the cost. *S-RIP* always accepts a consistent route and disregards an inconsistent one regardless of its cost. Therefore, optimal route involving a misbehaving node may not be used. We consider this as a good tradeoff between routing security and efficiency.

3.4 Efficiency Analysis

Suppose there are n routers and m subnets in a network. The average length of a route is $l + 1$ hops. For maximum security, every router would validate every route with all other routers on that route. For a single route with a length of $l + 1$ hops, the number of messages required for a consistency check, including requests and responses, is $2 \cdot l$. Each message will travel a number of hops. The first request message is sent to the node in two hops, and will travel 2 hops. The last request message is sent to the node in $l + 1$ hops, and will travel $l + 1$ hops. A response message will travel the same number of hops as the corresponding request message assuming they travel at the opposite direction of a same route. Therefore, the total number of hops (message transmissions) traveled by both request and response messages is $2 \cdot [2+3+\cdots+(l+1)] = (1+l) \cdot l$. Assume every router keeps a route for every subnet in the network. Each router would need $(1 + l) \cdot l \cdot m$ message transmissions for validating every route. Over the whole network, the total number of message transmissions in the most secure case is $(1 + l) \cdot l \cdot m \cdot n$.

Suppose we use RIP message for route request and response. Each route request would need two route entries, one for the routing information from the recipient to the ultimate destination, and one for the routing information from the recipient to its predecessor node on that route. The RIP message header is (24 bytes including authentication data), and each route entry is 20 bytes. Thus, one route request or response is 64 bytes. Plus the UDP header (8 bytes) and IP header (20 bytes), a packet carry a route request or response is 92 bytes. The total overhead of routing validation, in addition to the overhead of regular routing updates, in the most secure case is $92 \cdot (1 + l) \cdot l \cdot m \cdot n$ bytes.

Several route requests or responses may be transmitted in a single message. For example, if v_1 advertises to v_0 3 routes with a same next hop v_2. v_0 can send a single message with 4 route entries to v_2, one for each of three advertised destinations and one for v_1. The size of the packet carrying this message is 132 bytes, considerably less than 276 bytes, the total size of three standard packets with 92 bytes length each.

After a network becomes stable, *S-RIP* might only be triggered by a topology change, such as a link failure. After monitoring a production network with 61 routers and 142 subnets for 3 months, we observed 95 link failures, which is about one failure per day. Based on this, we suspect routing overhead generated by *S-RIP* would be light.

3.5 Incremental Deployment

A practical challenge of securing routing protocols is how to make the secured version interoperative with the existing infrastructure. Despite their technical merits, many proposed mechanisms for securing routing protocols are not widely deployed due to the fact that they require significant modifications to existing implementations and/or do not provide backward interoperability. Since it is unrealistic to expect that an existing routing infrastructure can be replaced by a secured version in a very short period of time, ideally a secured version should be compatible with the nonsecure protocols. It is also desirable that security can be increased progressively as more routers are deployed with the secured protocol.

To this end, *S-RIP* supports incremental deployment. We propose that messages exchanged in *S-RIP* conform to the message format defined in RIP. *S-RIP* can be implemented as a compatible upgrade to the existing RIP, and a *S-RIP* router performs the functions the same way as a RIP router. Therefore, deploying *S-RIP* on a router only requires a down time for the period of installation and rebooting of RIP processes. request from a non direct neighbor (remote node), a *S-RIP* router can successfully get information (albeit not authenticated) from a non-secured router for a consistency check. In other words, a RIP router can participate in a consistency check, but not initiate a consistency check. Thus, even before *S-RIP* is deployed on all routers, the routing table of a *S-RIP* router is partially protected as it is built from validated routing updates. The more routers deployed with *S-RIP*, the more reliable the routing tables in the network become. Therefore, we can say that security can be increased incrementally.

Although *S-RIP* can interoperate with RIP and supports incremental deployment, its security is limited

during the process of deployment. One vulnerability is that a RIP router does not participate in an entity authentication. *S-RIP* requires a router to demonstrate the knowledge of a secret key for itself or for a directly attached subnet when it claims a zero distance for that destination (see Table 1). Since RIP does have such functionality built in, an *S-RIP* router will not be able to validate a route with a zero distance claimed by a RIP router. To interoperate with a RIP router, an *S-RIP* router must accept zero distance routes advertised by a RIP router. Therefore, a compromised RIP router can successfully claim to be directly attached to non-existing subnets or impersonate subnets attached to other RIP routers. Another vulnerability is that a routing request/response sent to/from a remote RIP router is unauthenticated. RIP with MD5 extension [1] only provides message authentication for routing advertisements between two directly connected neighbors since a secret key is configured per network link. Therefore, routing requests/responses between two remote nodes are not authenticated. The consequences and countermeasures are discussed in §3.3.

4 Simulation

We study how security and efficiency of *S-RIP* are affected by different settings of the two thresholds (§2.3). We implemented *S-RIP* in the network simulator NS2 [10] as an extension to the distance vector routing protocol provided by NS2. The entry point of *S-RIP* is in the procedure *compute-routes*{} in the Tcl DV class. *S-RIP* is triggered if an advertised route is used to update a recipient's routing table.

4.1 Simulation Environment

Network Topology: we simulated a network with 50 routers and 82 network links. The network topology is based on a real network with 5 geographic locations, each location has two core routers. Each core router in one location is connected to one of the core routers in every other location. *Fraud*: we simulated misbehaving nodes which commit either or both shorter and longer distance fraud (§3.3). We randomly selected 5, 10, 15, 20, and 25 nodes to commit fraud in each run of the simulation. Note that 25 misbehaving nodes represent 50% of the total nodes. Each misbehaving node periodically (every 2.5 seconds) randomly selects a route from its routing table and makes its distance shorter or longer. *Simulation Scenarios*: we simulated 5 scenarios (Table 3) by adjusting the thresholds θ_1 and θ_2. Each simulation runs 180 seconds. A node reputation above θ_2 or below θ_1 times out after 2 seconds.

4.2 Metrics

Risk Window of Accepting a Malicious Route. We counted the total number of times (n_1) advertised routes are accepted for which consistency check is not performed due to the fact that the advertisers have a reputation higher than θ_2 (i.e., they are trusted). We also counted the total number of times (n_2) advertised routes are checked for their consistency. $\frac{n_1}{n_1+n_2}$ represents a probability that a malicious route may be accepted. It is used to measure the risk that a routing table may be poisoned by malicious updates.

S-RIP Overhead. To determine how much network overhead is generated by *S-RIP*, we compared the *S-RIP* overhead to the total routing overhead, which is calculated as the sum of *S-RIP* overhead and regular routing update overhead in RIP. Since the distance vector routing protocol provided by NS2 is not a strict implementation of RIP RFCs, we could not obtain network overhead directly from the NS2 trace file. We use $\frac{92x}{92x+632y}$ to calculate the ratio of *S-RIP* overhead and the total routing overhead, where x is the total number of *S-RIP* message transmissions, y is the total number of rounds of regular routing updates, 92 bytes is the size of the packet carrying a *S-RIP* message (see §3.4), and 632 bytes is the overhead generated by one router in one round of regular routing updates. x and y are derived from simulation outputs, which are used to generate Figure 3.

Maximally Secured	$\theta_1 = 0$	$\theta_2 = 1$
Partially Secured-1	$\theta_1 = 0.1$	$\theta_2 = 0.9$
Partially Secured-2	$\theta_1 = 0.2$	$\theta_2 = 0.8$
Partially Secured-3	$\theta_1 = 0.3$	$\theta_2 = 0.7$
Not Secured	$\theta_1 = 0$	$\theta_2 = 0$

Table 3: Simulation Scenarios

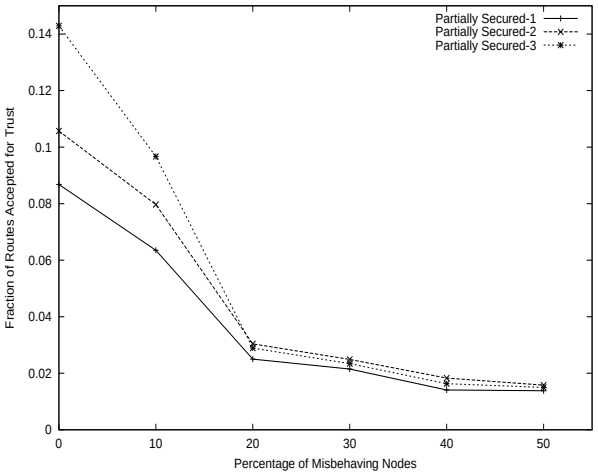


Figure 2: Fraction of routes accepted for which consistency checks are not performed.

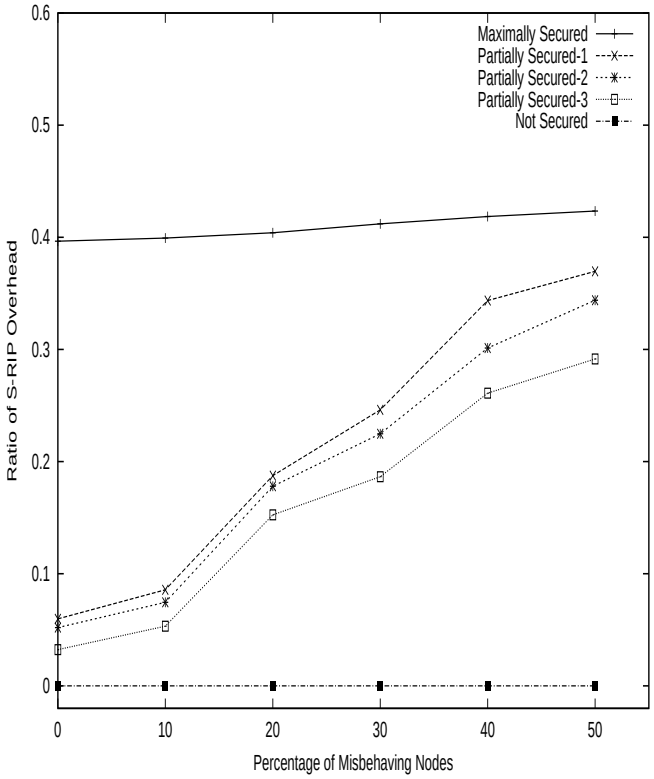


Figure 3: Network overhead generated by *S-RIP*.

4.3 Simulation Results

By looking at the output data from the simulation, we observed that an advertised malicious route can be successfully detected by a consistency check. This is precisely what we expected.

Risk Window of Accepting a Malicious Route. Figure 2 shows: 1) The lower the threshold2 (θ_2), the higher the risk of accepting a malicious route. This is because that the higher θ_2, the longer it takes a node to become trusted. 2) The less number of misbehaving nodes, the higher risk of accepting a malicious route. The reason is that when there are less number of misbehaving nodes in the network, more nodes will be trusted, i.e., having a reputation above θ_2. 3) When there are 20% or more misbehaving nodes, such risk is very low (less than 3%). The difference among three partially secured scenarios becomes insignificant since most nodes have a reputation less than 0.7, i.e., they are not trusted in all three scenarios. Although the risk of routing tables being poisoned becomes low, the risk that correct routes are not in routing tables becomes high. Therefore, data traffic may be dropped, though not routed to a misbehaving node.

S-RIP Overhead. Figure 3 compares the *S-RIP* overhead in different scenarios. 1) In a maximally secured network, *S-RIP* overhead is very high (about 40% of the total routing overhead). The *S-RIP* overhead stays relatively flat when the number of misbehaving nodes increases. This is because every node needs to validate every route with every other node on that route. In our implementation, a new route is not considered if the current route is being checked for consistency. Since it takes long time for a consistency check to complete, most new route changes (malicious or non-malicious) are not checked for their consistency. Therefore, overhead increased by new malicious updates is insignificant. This indicates that the speed of network convergence is significantly slowed down. We expect that it would make no difference in terms of overhead if we allow a new route to interrupt an ongoing consistency check as several uncompleted consistency checks would generate similar amount of *S-RIP* overhead as a completed one does. 2) In the

three partially secured scenarios, *S-RIP* overhead is relatively low (less than 8.6%) when there are only 10% of misbehaving nodes. *S-RIP* overhead increases significantly when the number of misbehaving nodes increases. Since the number of nodes involved in a consistency check is relatively low in these scenarios, it takes less time to complete. Thus more malicious updates will trigger more consistency checks and result in more *S-RIP* overhead. *S-RIP*overhead decreases when θ_1 and θ_2 are moved toward each other because: a) the number of routes accepted without being checked for consistency increases when there are less than 20% of misbehaving nodes (see Figure 2); b) the number of routes dropped without being checked for consistency increases when more than 20% of the nodes misbehave. 3) There is no *S-RIP* overhead in a non-secured network since *S-RIP* is never triggered.

5 Related Work

Significant work has been done in securing routing protocols. Perlman [28] is the first to recognize and to study the problem of securing routing infrastructures. Perlman classified router failures into two categories: *simple failures* and *byzantine failures*. A router with simple failure stops functioning completely. A router with byzantine failure may be functioning, but not properly. A byzantine failure could be caused by hardware faults, software bugs, misconfiguration, or malicious software. Perlman proposed to use public key signatures, resource reservation, hop by hop acknowledgments, and source routing, among other mechanisms, to achieve robust flooding and robust routing. The proposed solution can guarantee to find a non-faulty path (i.e., all intermediate links and routers on the path are non-faulty) from a non-faulty source to a non-faulty destination provided there is a such a path in the network. Our proposed approach differs in that we are targeting securing existing routing infrastructures with support of incremental deployment, while Perlman's solution is to build a new robust routing protocol. Our proposed *S-RIP* is also based on a distance vector routing protocol, while Perlman's solution is based on link state routing protocol.

Kumar [20] proposed to use digital signatures and sequence numbers to protect subverted network links. By gaining the control of a network link, an intruder can manipulate routing updates, e.g., modifying or replaying routing updates. Digital signatures can prevent unauthorized alternation of routing updates. Sequentially numbering routing updates can prevent replay attacks. Kumar also proposed to use retransmission and acknowledgments for improving reliability and security.

Smith et al. [35] proposed to use digital signatures, sequence numbers, and a loop-free path finding algorithm for securing distance vector routing protocols. By including an additional field, next-to-last-hop, in an advertised route, a recipient node can validate based on its local routing information if a routing loop can be formed. One disadvantage is that it cannot prevent longer or shorter distance fraud.

Zhang [39] considered that public-key based digital signatures are computationally inefficient for signing routing advertisements given the fact that both signature generation and verification processes must be done online using computational inefficient algorithms. Zhang proposed to use one-time digital signatures and one-way hash chains for signing routing updates. One disadvantage of this approach is that it significantly increases message overhead.

Goodrich [12] proposed a method, called leap-frog, for securing distance vector routing protocols. The proposed method can ensure that a node always chooses the minimum value as the cost for a destination among all the values received from its neighbors. This approach inexplicitly assume that one node will receive an advertised route for a destination from every of its neighbors, which may not be true. Another disadvantage is that it is vulnerable to replay attacks.

Mittal and Vigna [24] proposed to use intrusion detection for securing distance vector routing protocols. A precomputed master routing database contains paths and associated costs from every node to every other node. Sensors are installed on some or all subnets. Routing information related to a subnet is extracted automatically from the master routing database, and is distributed to the sensor installed on that subnet. A

sensor uses its portion of the master routing information to validate routing advertisements transmitted in its subnet. One notable advantage is that it does not require modifications to the routing protocol being secured. Thus, it allows incremental deployment. One disadvantage is that it cannot prevent fraudulent routing advertisements from poisoning others' routing tables, although it may be able to detect them.

Hu et al. [16, 17] proposed several efficient mechanisms using one-way hash chains and authentication trees as construction primitives for securing distance vector routing protocols. Their approach can prevent newer sequence number and shorter (or same) distance fraud, but not longer distance fraud. Another disadvantage is that it incurs significant message overhead since each advertised route must include a hash value to authenticate itself.

Goodell et al [13] proposed a protocol for improving the security and accuracy of Border Gateway Protocol (BGP) [30]. In their approach, each Autonomous System (AS) designates a server, namely Interdomain Routing Validator (IRV), as the authority of its routing information. An IRV maintains a routing database, which is separated from, but needs to be synchronized with, the routing tables of BGP speakers. An IRV can send queries to another IRV to validate the BGP routing information of the AS that IRV is responsible for. This approach is a second line of defense mechanism, which can detect incorrect BGP routing information, but cannot prevent it from propagating. One similarity of our proposed approach and IRV is that both propose to validate routing information. The difference is that our approach can be integrated with the existing routing protocol and validate routing information before being used to update the routing table, while IRV is separated from and parallel with the existing BGP infrastructure.

Many researchers have explored securing link state routing protocols (e.g., OSPF) [28, 29, 26, 5, 6, 3, 36] and BGP [34, 19, 13]. Securing wireless ad hoc networks has also attracted extensive interest [40, 22, 40, 4, 15, 16, 38, 18]. Reputation-based systems have been used to facilitate trust in electronic commerces [31, 37].

6 Concluding Remarks

We expect that our methodology of correlating routing information by cross checking their consistency could be applied to securing link state routing protocols (e.g., OSPF) and inter-domain routing protocols (e.g. BGP). The sized window based on Equation 5 could be used for balancing security and efficiency in other problems, e.g., detecting misbehaving nodes by data correlation in distributed sensor networks or wireless ad hoc networks.

We plan to make our simulation code publicly available. Our future researches include: 1) supplementing our simulations by a multinode testbed in our lab based on Linux and implementing *S-RIP* as an extension to *routed* (a daemon implementing RIP); 2) applying the framework to securing BGP.

Acknowledgment

The first author is supported in part by Alcatel Canada and OCIPEP (Office of Critical Infrastructure Protection and Emergency Preparedness, Government of Canada) Research Fellowship. The second author is supported in part by NSERC (Natural Sciences and Engineering Research Council of Canada) and MITACS (Mathematics of Information Technology and Complex Systems). The third author is Canada Research Chair in Network and Software Security, and is supported in part by a NSERC Discovery Grant, the Canada Research Chairs Program, and Alcatel Canada.

References

[1] F. Baker and R. Atkinson. RIP-II MD5 Authentication. RFC 2082 (Proposed Standard), January 1997.

[2] S.M. Bellovin. Security Problems in the TCP/IP Protocol Suite. *ACM Computer Communications Review*, 19(2): 32-48, April 1989.

[3] K.A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R.A. Olsson. Detecting Disruptive Routers: A Distributed Network Monitoring Approach. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 115-124, May 1998.

[4] S. Buchegger and J.Y. Le Boudec. Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes - Fairness In Dynamic Ad-hoc NeTworks) In *Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing*, (MobiHoc 2002), June 2002.

[5] S. Cheung. An Efficient Message Authentication Scheme for Link State Routing. In *Proceedings of the* 13^{th} *Annual Computer Security Applications Conference*, San Diego, California, USA, December 1997.

[6] S. Cheung and K. Levitt. Protecting routing infrastructure from denial of service using cooperative intrusion detection. In *Proc. New Security Paradigms Workshop*, Cumbria, UK, September 1997.

[7] S. Deering, S. Hares, C. Perkins, and R. Perlman. Overview of the 1998 IAB Routing Workshop (RFC 2902). August, 2000.

[8] A.P. Dempster. Upper and Lower Probabilities Induced by a Multivalued Mapping. *The Annals of Statistics*, 28: pages 325-339, 1967.

[9] D. Estrin, J. Postel, and Y. Rekhter. Routing Arbiter Architecture. *http://www.isi.edu/div7/ra/Publications/*, June 1994.

[10] K. Fall and K. Varadhan, editors. The *ns* Manual (formerly *ns* Notes and Documentation). April 14, 2002. http://www.isi.edu/nsnam/ns/doc/index.html

[11] J.J. Garcia-Luna-Aceves and S. Murthy. A Loop-Free Algorithm Based on Predecessor Information. In *Proceedings of IEEE INFOCOM* , Boston, MA, USA. April 1995.

[12] M.T. Goodrich. Efficient and Secure Network Routing Algorithms. *Provisional Patent Filing*, USA. January 2001.

[13] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. McDaniel, and A. Rubin. Working around BGP: An Incremental Approach to Improving Security and Accuracy in Interdomain Routing. In *Proceedings of 2003 Internet Society Symposium on Network and Distributed System Security (NDSS'03)*, San Diego, California, USA. February 2003.

[14] C. Hedrick. Routing Information Protocol. RFC 1058. June 1988.

[15] Y.C. Hu, A. Perrig, and D.B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. In *Proceedings of the Eighth ACM International Conference on Mobile Computing and Networking (MobiCom 2002)*, September 23-28, 2002.

[16] Y.C. Hu, D.B. Johnson, and A. Perrig. Secure Efficient Distance Vector Routing Protocol in Mobile wireless Ad Hoc Networks. In *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002)*, June 2002.

[17] Y.C. Hu, A. Perrig, and D.B. Johnson. Efficient Security Mechanisms for Routing Protocols. In *Proceedings of 2003 Internet Society Symposium on Network and Distributed System Security (NDSS'03)*, San Diego, California, USA. February 2003.

[18] Y.C. Hu, A. Perrig, and D.B. Johnson. Packet Leashes: A Defense against Wormhole Attacks in Wireless Networks. In *Proceedings of IEEE INFOCOM 2003*, San Francisco, California, USA. April 2003.

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

[20] B. Kumar. Integration of Security in Network Routing Protocols. In ACM SIGSAC Review, 11(2): 18-25, Spring 1993.

[21] G. Malkin. RIP Version 2. RFC 2453 (Standard). November 1998.

[22] S. Marti, T.J. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 2000)*, August 2000.

[23] A.J. Menezes, P.C. van Oorschot, and S. Vanstone. Handbook of Applied Cryptography. CRC Press, 1996.
[24] V. Mittal and G. Vigna. Sensor-Based Intrusion Detection for Intra-Domain Distance-Vector Routing. In *Proceedings of the 9th ACM Conferences on Computer and Communication Security (CCS'02)*, Washington, D.C., USA. November 18-22 2002.
[25] S.L. Murphy. Presentation in Panel on "Security Architecture for the Internet Infrastructure". *1995 Internet Society Symposium on Network and Distributed System Security (NDSS'95)*, San Diego, California, USA. April 1995.
[26] S.L. Murphy and M.R. Badger. Digital Signature Protection of the OSPF Routing Protocol. In *Proceedings of the 1995 Internet Society Symposium on Network and Distributed System Security (NDSS'96)*, San Diego, California, USA. April 1996.
[27] V.N. Padmanabhan and D.R. Simon. Secure Traceroute to Detect Faulty or Malicious Routing. *ACM SIGCOMM Workshop on Hot Topic in Networks (HotNets-I)*, Princeton, New Jersey, USA. October 2002.
[28] R. Perlman. Network Layer Protocols with Byzantine Robustness. PhD thesis, Massachusetts Institute of Technology, August 1988.
[29] R. Perlman. Interconnections: Bridges and Routers. Addison-Wesley, 1992.
[30] Y. Rekhter and T. Li. A Border Gateway Protocol 4 (BGP-4), RFC 1771, March 1995.
[31] P Resnick, R. Zeckhauser, E. Friedman, and K. Kuwabara. Reputation systems: Facilitating trust in Internet interactions. *Communications of the ACM*, 43(12): 45-48, 2000.
[32] R. Rivest. The MD5 Message-Digest Algorithm, RFC 1321, April 1992.
[33] G. Shafer. A Mathematical Theory of Evidence. Princeton, NJ, Princeton University Press, 1976.
[34] B.R. Smith and J.J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocol. In *Proceedings of Global Internet 1996.* London, UK. November 1996.
[35] B.R. Smith, S. Murphy, and J.J. Garcia-Luna-Aceves. Securing Distance-Vector Routing Protocols. In *Proceedings of 1997 Internet Society Symposium on Network and Distributed System Security (NDSS'97)*, San Diego, California, USA. February 1997.
[36] F.Y. Wang and F.S. Wu. On the Vulnerablity and Protection of OSPF Routing Protocol. In *Proceedings of IEEE Seventh International Conference on Computer Communications and Networks*, Lafayette, LA, USA. Oct 12-15, 1998.
[37] B. Yu and M.P. Singh. Distributed Reputation Management for Electronic Commerce. In *Computational Intelligence*, 18(4): 535-549, 2002.
[38] M.G. Zapata and N. Asokan. Securing Ad Hoc Routing Protocols. In *Proceedings of the ACM Workshop on Wireless Security (WiSe 2002)*, September 2002.
[39] Kan Zhang. Efficient Protocols for Signing Routing Messages. In *Proceedings of 1998 Internet Society Symposium on Network and Distributed System Security (NDSS'98)*, San Diego, California, USA, March 1998.
[40] Y.G. Zhang, W. Lee and Y.A. Huang. Intrusion Detection in Wireless Ad-Hoc Networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 2000)*, August 2000.
[41] L. Zhou and Z.J. Haas. Securing Ad Hoc Networks. *IEEE Network Magazine*, 13(6), November/December 1999.
[42] L. Eschenauer and V. Gligor. A Key-Management Scheme for Distributed Sensor Networks. In *Proceedings of the 9th ACM Conferences on Computer and Communication Security (CCS'02)*, Washington, D.C., USA. November 18-22 2002.

Appendix 1 Routing Information Protocol

Overview

RIP is an intra-domain routing protocol. Although it has certain limitations, e.g., the maximum distance between two nodes is limited to 15 hops, RIP is still widely used by many small and medium size organizations.

A RIP routing table consists of a number of entries, one for each destination in the network. Each route entry contains at least the following information: *destination* - the IP address and the subnet mask of the destination; *distance* - number of hops from this router to the destination; *next hop* - the IP address of the next router along the path to the destination; *timers* - a number of timers are associated with each route entry. One timer is set to 180 seconds. If no routing update about this route is received within three minutes, the distance of this route is set to 16 which designates infinity in RIP. Whenever a routing update is received about this destination, the timer is re-initialized to 180 seconds.

A router speaking RIP periodically (every 30 seconds) broadcasts routing update messages to its neighbors to advertise its routing information. A routing update message consists of up to 25 routes. Each route contains the information of destination (IP and subnet mask), distance, and the next hop. The next hop is only useful if it is directly reachable to a recipient.

A routing update message may also be triggered by a route distance change. A router or a host can also solicit routing update messages by sending out a route update request. For example, after a router reboots, it will send out a route update request to collect routing information to initialize its routing table. A route request may also be sent out for a diagnosis purpose.

RIP does not require that a routing update request is from a direct neighbor, although it requires a routing update response must be from a direct neighbor. Therefore, a route update response message may be sent to a remote node (i.e., two or more hops away). Some proposed methods [17] for securing distance vector routing protocols that inexplicitly assume that routing update messages are only sent to direct neighbors may need additional mechanisms to take this into account.

An advertised route may lead to the update of a recipient's routing table if 1) it is a new route; 2) it is better than the existing route; or 3) it is from the originator of the existing route. In case 3), the the existing route will always be updated no matter whether the new route is shorter or longer than the existing one. If the advertised route is the same as the existing one, only the time-out timer associated with that route is reinitialized.

Although it is understood that it is important to validate a routing update response message carefully before using it to update one's routing table, RIPv2 only performs rudimentary checks (e.g., the source IP address and port number). Our proposed approach extends the validation by cross checking route consistency.

Since RIP does not keep the complete path information to a destination, it is possible that one router advertises a route to another router from which it learns that route. This will lead to the problem of counting to infinity. RIP adopts an approach, namely *Split Horizon*, to solve the problem. One router does not advertise a route back to the router from which it learned the route. An extension of split horizon is to advertise the route back to the router where it learned it from , but with a distance of infinity (16). This is called *Split Horizon with Poisoned Reverse*.

Although split horizon can break a loop between two nodes, it cannot break a loop among three or more nodes. *Triggered Updates* can speed the network convergence in that whenever the metric of a route is changed, a routing update is triggered immediately without waiting for the normal periodic routing update.

Security Vulnerabilities

RIP has several known security vulnerabilities. RIPv1 does not provide any authentication service. RIPv2 only uses a cleartext password for authenticating peers. Since a cleartext password can be easily captured,

it provides only marginal additional security in practice. Keyed MD5 has been proposed [1] to replace the password-based authentication mechanism, but it does not prevent a compromised node or a malicious legitimate node (i.e. a questionable node) from advertising fraudulent routing information about distance or next hop.

First, a questionable node can claim a zero distance to a non-directly connected network or a nonexistent network. The proposed MD5 authentication requires a secret key(s) is shared by a pair of nodes for each directly connected network link. This can prevent router impersonation, but cannot prevent network impersonation. Second, a questionable node may claim a distance shorter than the actual distance for a destination. For example, a node i which is 5 hops away from subnet N_i could advertise a route for N_i with a distance of 3 hops. Third, a questionable node can claim a distance longer than the actual distance for a destination. Fourth, a questionable node can provide false information on a next hop. For example, suppose i advertises to j a route for N_i with node k as the next hop. If j and k are in a same subnet, j would set k, instead of j, as the next hop to N_i. If the actual next hop from i to N_i is not in the same subnet as j, but i intentionally makes it in the same subnet as j, j would be misled.

Appendix 2: Secure Routing Information Protocol (*S-RIP*)

Algorithm 1 Secure Routing Information Protocol (*S-RIP*)

INPUT: $v_0, v_1, [v_n, dist(v_1, v_n), nh(v_1, v_n)], \theta_1, \theta_2$
OUTPUT: accept or reject $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$
if $r_{v_0}(v_1) > \theta_2 \parallel dist(v_1, v_n) = 16$ **then**
    Accept $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$; goto: END
end if
Entity Authentication:
if $dist(v_1, v_n) = 0$ **then**
    Perform Entity Authentication of v_n.
    if v_1 demonstrates the knowledge of secret k_{v_0,v_n} **then**
        Accept $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$
    else
        Reject $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$
    end if
    goto: END
end if
$i = 1$
while TRUE **do**
    $i = i + 1$; $v_i = nh(v_{i-1}, v_n)$;
    Request from v_i: $[v_n, *, *]$ and $[v_{i-1}, *, *]$
    Wait until receiving $[v_n, dist(v_i, v_n), nh(v_i, v_n)]$ and $[v_{i-1}, dist(v_i, v_{i-1}), nh(v_i, v_{i-1})]$
    if $nh(v_i, v_n) \in \{v_j\}$ $(1 \leq j \leq i-1)$ **then**
        Reject $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$; goto: END
    end if
    if $dist(v_{i-1}, v_n) = dist(v_i, v_n) + dist(v_i, v_{i-1})$ **then**
        calculate $r_{v_0}(v[1..i])$
        if $r_{v_0}(v[1..i]) > \theta_2$ **then**
            Accept $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$; goto: END
        else if $dist(v_i, v_n) = 0$ **then**
            goto: Entity Authentication
        end if
    else
        Reject $[v_n, dist(v_1, v_n), nh(v_1, v_n)]$; goto: END
    end if
end while
END

Appendix 3: Overview of Dempster-Shafer Theory

Overview

In Dempster-Shafer theory, a Universe of Discourse, denoted by U, is a set of mutually exclusive alternatives, e.g., $U = \{s_1, s_2, \ldots, s_n\}$. There are three concepts associated with each set $S (S \subseteq U)$, basic probability assignment, belief, and plausible belief.

- *Basic Probability Assignment,* denoted by $m(S)$, is the strength of evidence supporting S, where $0 \leq m(S) \leq 1$. The sum of the basic probability assignments of all subsets of U is equal to 1, i.e., $\sum_{S \subseteq U}(m(S)) = 1$. The basic probability assignment in DST is different from the probability assignment in classicial probability theory in that its domain is the powerset of U instead of the elements of U. Therefore, $m(S)$ should not be interpreted as the precise probability that S is true as in the classical probability theory. $m(S)$ just represents one piece of evidence that will contribute to our final belief in S.

  If there is only one element in S, e.g., $S = \{s_1\}$, it is clear that $m(S)$ represents the strength of the evidence that supports s_1. If there are more than one element in S, e.g., $S = \{s_1, s_2\}$, we interpret $m(S)$ or $m(\{s_1, s_2\})$ as a piece of evidence that will contribute to our final belief in s_1 by the amount of $m(S)$, or in s_2 by the amount of $m(S)$, or in both by the amount of x and y respectively with $x + y = m(\{s_1, s_2\})$. $m(\{s_1, s_2\})$ characterizes the uncertainty of the evidence as it is not clear how much the evidene will contribute exactly to our final belief in each of its elements (e.g., s_1, s_2). With additional evidence from indepedent sources, the uncertainty will be reduced. Let $|S|$ denote the number of elements in S, or the size of S. If there is no uncertainty, then for all S, where $S \subseteq U$ and $|S| > 1$, $m(S) = 0$. Since $\sum_{S \subseteq U}(m(S)) = 1$, we obtain $\sum_{i=1}^{n}(m(\{s_i\})) = 1$, which is equavelent to the probability assignment in classical probability theory. Therefore, we can say that DST is a generalization of the classical probability theory.

- *Belief,* denoted by $b(S)$ $(0 \leq b(S) \leq 1)$, is defined by the sum of the basic probability assignments of all the subsets of S, i.e., $b(S) = \sum_{X \subseteq S}(m(X))$. $b(S)$ can be interpreted as the belief in S drawn from all the known evidence which supports S.

  If $|S| = 1$, e.g., $S = \{s_1\}$, $b(S)$ is equavelent to the probability that s_1 is true given the known evidence. If $|S| > 1$, e.g., $S = \{s_1, s_2\}$, $b(S)$ should not be interpreted as the probability that both s_1 and s_2 are true. $b(S)$ represents the accumulated uncertainty of evidence which cannot be precisely assigned to each of the elements of S.

- *Plausible Belief,* denoted by $pl(S)$, is defined by $1 - b(\bar{S})$. $pl(S)$ is in the range $[0, 1]$, and can be interpreted as the belief in S if all the unknown evidence turns out to be supportive to S or to be against $\bar{S}$.

Since there is always evidence which is unknown to us at the moment when we make judgement on a subject (e.g., S), the true belief in S, $tb(S)$, is always in the interval of the belief, $b(S)$, and the plausible belief, $pl(S)$, as defined in DST. If all the unknown facts turn out to be supportive to S, then $tb(S)$, $b(S)$, and $pl(S)$ are all equal. If all the unknown facts turn out to be against S, then $tb(S)$ is equal to $b(S)$, and both of them are less than $pl(S)$.

Evidences from two independent sources, represented by the basic probability assignments $m_1(S)$ and $m_2(S)$ respectively, can be combined to yield a new basic probability assignment, $m_3(S)$, using Equation 6.

$$m_3(S) = \frac{\sum_{X \cap Y = S} m_1(X) \cdot m_2(Y)}{1 - \sum_{X \cap Y = \phi} m_1(X) \cdot m_2(Y)} \qquad (X, Y \subseteq U) \tag{6}$$

Proof of Equation 5 Using DST

Let s be a proposition (e.g., a route advertisement), $\bar{s}$ be the complementary of s, and $U = \{s, \bar{s}\}$. For a corroborating group $(v_1, v_2, \ldots, v_n)$ which agree upon s, each of $r_x(v_1), r_x(v_2), \ldots, r_x(v_n)$ can be treated as a piece of independent evidence supporting s. Therefore, node x has n basic probability assignments for U, which is listed in Table 4. In the rest of this section, the following notations are used:

- The subscript x is omitted for the purpose of convenience.
- $m_{[1,2]}$: the basic probability assignment combined from m_1 and m_2.
- $m_{[1..k]}$: the basic probability assignment combined from m_i $(1 \leq i \leq k)$.
- $Bel_{[1,2]}$: the belief computed from the basic probability assignment $m_{[1,2]}$.
- $Bel_{[1..k]}$: the belief computed from the basic probability assignment $m_{[1..k]}$.

	$\{s\}$	$\{\bar{s}\}$	$\{s,\bar{s}\}$
m_1	$r(v_1)$	0	$1 - r(v_1)$
m_2	$r(v_2)$	0	$1 - r(v_2)$
$\ldots$	$\ldots$	$\ldots$	$\ldots$
m_n	$r(v_n)$	0	$1 - r(v_n)$

Table 4: n Basic Probability Assignments for U, $U = \{s, \bar{s}\}$

We want to show that the accumulated confidence, $r(v[1..n])$, in the corroborating group $\{v_1, v_2, \ldots, v_n\}$ calculated by Equation 5 is consistent with the combined belief $Bel_{[1..n]}(\{s\})$ computed from n independent basic probability assignments by DST. Since $Bel_{[1..n]}(\{s\}) = \sum_{X \subseteq \{s\}}(m_{[1..n]}(X)) = m_{[1..n]}(\{s\}) + m_{[1..n]}(\phi) = m_{[1..n]}(\{s\})$, we only need to show that $r(v[1..n]) = m_{[1..n]}(\{s\})$.

Step 1: When $n = 1$, we know from Table 4 that $m_1(\{s\}) = r(v_1)$.

Step 2: Assume when $n = k$, $m_{[1..k]}(\{s\}) = r(v[1..k])$. Since none of the nodes in the corroborating group $\{v_1, v_2, \ldots, v_k\}$ supports $\{\bar{s}\}$, we know that $m_{[1..k]}(\{\bar{s}\}) = 0$, and $m_{[1..k]}(\{s, \bar{s}\}) = 1 - r(v[1..k])$.

Step 3: When $n = k + 1$, $m_{[1..k+1]}(\{s\})$ can be computed from the basic probability assignments $m_{[1..k]}$ and m_{k+1} using Equation 6 as follows:

$$
\begin{aligned}
m_{[1..k+1]}(\{s\}) &= \frac{\sum_{X \cap Y = S} m_{[1..k]}(X) \cdot m_{k+1}(Y)}{1 - \sum_{X \cap Y = \phi} m_{[1..k]}(X) \cdot m_{k+1}(Y)} \\
&= \frac{m_{[1..k]}(\{s\}) \cdot m_{k+1}(\{s\}) + m_{[1..k]}(\{s\}) \cdot m_{k+1}(\{s, \bar{s}\}) + m_{[1..k]}(\{s, \bar{s}\}) \cdot m_{k+1}(\{s\})}{1 - \left[m_{[1..k]}(\{s\}) \cdot m_{k+1}(\{\bar{s}\}) + m_{[1..k]}(\{\bar{s}\}) \cdot m_{k+1}(\{s\})\right]} \\
&= \frac{r(v_{[1..k]}) \cdot r(v_{k+1}) + r(v_{[1..k]}) \cdot \left[1 - r(v_{k+1})\right] + \left[1 - r(v_{[1..k]})\right] \cdot r(v_{k+1})}{1 - \left[r(v_{[1..k]}) \cdot 0 + 0 \cdot r(v_{k+1})\right]} \\
&= r(v_{[1..k]}) + \left[1 - r(v_{[1..k]})\right] r(v_{k+1})
\end{aligned}
$$

The induction proof shows that the accumulated confidence in s, $r(v_{[1..n]})$ calculated using Equation 5 from the colloborating group $\{r_1, r_2, \ldots, r_n\}$ is consistent with the combined belief in s, computed using

DST from n independent basic probability assignments. The advantage of Equation 5 is that it is simple and intuitive. It allows one to compute a new accumulated confidence from a previous value when the size of the corrobarating group grows. The computation is also efficient. Although Dempster-Shafer theory is more general, e.g., it can handle conflicting information, it is computationally inefficient since it involves set operations.