Attack Propagation in Networks[*]

Sotiris Nikoletseas Grigorios Prasinos Paul Spirakis Christos Zaroliagis
Computer Technology Institute, P.O. Box 1122, 26110 Patras, Greece
Department of Computer Engineering & Informatics, University of Patras, 26500 Patras, Greece
Emails: {nikole@cti.gr, green@ceid.upatras.gr, spirakis@cti.gr, zaro@ceid.upatras.gr}

ABSTRACT

A new model for intrusion and its propagation through various attack schemes in networks is considered. The model is characterized by the number of network nodes n, and two parameters f and g. Parameter f represents the probability of failure of an attack to a node and is a gross measure of the level of security of the attacked system and perhaps of the intruder's skills; g represents a limit on the number of attacks that the intrusion software can ever try, when it issues them from a particular (broken) network node, due to the danger to be discovered. The success of the attack scheme is characterized by two factors: the number of nodes captured (the spread factor) and the number of virtual links that a defense mechanism has to trace from any node where the attack is active to the origin of the intrusion (the traceability factor). The goal of an intruder is to maximize both factors. In our model, we present four different ways (attack schemes) by which an intruder can organize his attacks. Using analytic and experimental methods, we first show that for any $0 < f < 1$, there exists a constant g for which any of our attack schemes can achieve a $\Theta(n)$ spread and traceability factor with high probability, given sufficient propagation time. We also show for three of our attack schemes that the spread and the traceability factors are, with high probability, linearly related during the whole duration of the attack propagation. This implies that it will not be easy for a detection mechanism to trace the origin of the intrusion, since it will have to trace a number of links proportional to the nodes captured.

1. INTRODUCTION

Attacks in computer networks pose several key problems regarding intrusion propagation and detection [1, 3, 7, 8]. Various models have been proposed under which researchers mainly study the effective detection and defeat of attacks assuming a very powerful intruder; see e.g., [6]. In this setting, intrusion propagation (the process of spread of such attacks) has mostly been investigated under gossip or epidemiological models [4]. However, the fear of malicious attacks along with the development of advanced cryptographic techniques has considerably increased the security level of nowadays computer systems. Contrary to previous studies, we are interested here in studying intrusion propagation assuming that the intruder has a rather limited power and would like to investigate how intrusion can propagate in a perhaps highly secure network. To this end, we introduce a general model for such an intrusion and its propagation in networks.

Let $\mathcal{N}$ be an n-node network, e.g., a computer network, with some physical infrastructure underlying it and whose specific topology is not a concern. By the term "network", we do not necessarily mean that communication is done point-to-point. We rather view a network as a collection of host systems each one having its own logical address (e.g., the internet), and whose underlying physical infrastructure uses advanced communication and interconnection technology such as buses, optical links, wireless communication media, etc. Direct communication between two nodes of $\mathcal{N}$ is achieved by establishing a *virtual* channel through the physical infrastructure between these two nodes.

Assume that in such a network an intruder, starting from his own computer, would like to break as many other systems as possible. The intrusion consists of a collection of attacks. An *attack* is issued from some node in $\mathcal{N}$ and is an attempt to break the perimetric security of another node (host system) in $\mathcal{N}$. The intrusion is realized by an attack scheme. An *attack scheme* is a protocol for the organization of the attacks issued from specific nodes of $\mathcal{N}$. The intruder is a greedy one, i.e., does not have a specific target, and attacks computer systems equiprobably at random. An attack succeeds or fails, independently of other attacks, with a failure probability $0 < f < 1$ that represents the difficulty of breaking a system in $\mathcal{N}$; f is a gross measure of the security level of the attacked systems (e.g., of the average security or the perceived maximum security level of a system) and may also depend on the intruder's skills. Our model assumption about f is motivated by a large class of existing attacks; for example, attacks that are based on randomly sampling a set of possible passwords from a large password domain and then trying each of them. Many practical local attack software programs (e.g., a cracking password worm) work in

[*]This work was partially supported by the IST Programme of EU under contract no. IST-1999-14186 (ALCOM-FT), by the Human Potential Programme of EU under contracts no. HPRN-CT-1999-00104 (AMORE) and HPRN-CT-1999-00112 (ARACNE), and by the Greek GSRT project ALKAD.

this way. Furthermore, the probability of success of such a scheme in a node does not depend on previous successes at other nodes or on previous attempts at the same node. This is because the locally implemented set of passwords is perhaps different in each node and the set of passwords used by the local attack software is very small (for reasons of speed) compared to the password domain set.

If an attack does not fail, then some, randomly and equiprobably chosen, network node is returned. Because of that, it may happen that an already selected node (an already broken system) is chosen again. If the result of a non-failed attack is a node which has not been selected before, then the attack is considered *successful* and a *virtual link* (virtual channel) is established to that node. The random selection, with possible repetition, of a node in the case of a non-failed attack is motivated by the following pragmatic considerations: (i) if the local attack software (e.g., a worm) is successfully confronted, it should not reveal any information about broken nodes in the past; (ii) the local attack software may blindly extend attacks to hosts contained in tables of the newly broken systems which may include the already broken ones. The intruder tries to protect himself as much as possible from being traced: once a system is broken, his software tries from *that* system to attack (again equiprobably at random) another system by disguising itself as a user process of the broken system. Because of the danger to be discovered, the intruder's software can ever try only a limited (i.e., constant) number g of attacks from a specific node of the network. If a successful attack is issued before the limit g is reached, then the software enters a dormant phase and performs no action (for the purpose of not raising any suspicions). If at some node i the software exhausts the attack bound g, then it terminates execution at i and "backtracks" to a previously broken system j to continue from there its attacks, provided that there are still some attempts left at j. In such a case, the local software at j is reactivated and starts again to issue attacks. If at any time during the execution of the attack scheme, the intrusion is discovered by some system, we assume that the whole attack scheme to $\mathcal{N}$ terminates.

Two natural questions raised here are: (a) how long the intruder can go (i.e., how many computer systems can be successfully attacked) in $\mathcal{N}$ until he is discovered, and (b) how many virtual links a detection mechanism has to trace in order to find the origin of the intrusion. In particular, assume that the intrusion starts at time 0 with attack scheme S. At any time $t \geq 0$, let $n_S(t)$ be the number of nodes captured, called the *spread factor*, and let $\ell_S(t)$ be the shortest possible distance (in number of virtual links) from the currently active position of the intruder's software to the origin, called the *traceability factor*. Given a discovery (i.e., stopping) time T, we would like to estimate $n_S(T)$ and $\ell_S(T)$. We shall refer to this as the *attack propagation* problem. The goal, from the side of the intruder, is to employ an attack scheme which maximizes *both* factors. It is not at all obvious how an intrusion can be organized so that both $n_S(T)$ and $\ell_S(T)$ are large. In fact, to the best of our knowledge, almost all epidemiological (and gossip propagation) models have usually very small $\ell_S(T)$ compared to $n_S(T)$, because of their "radially" spreading nature.

The above process defines naturally a graph G whose vertices correspond to the nodes of the network and if a virtual link (i.e., a successful attack through some virtual channel) is established between two nodes i and j, then an edge between vertices i and j is added to G. In this setting, the spread factor $n_S(T)$ is the size of the obtained connected component in G, and the traceability factor $\ell_S(T)$ is the length (number of edges) of the path in this connected component from the current vertex (node) issuing attacks to the origin (the node from which the intrusion was started). We shall refer to this path as the *traceability path.* Hence, the attack propagation problem reduces in estimating the values of these two quantities in G.

Our work is centered around the attack propagation problem. We present four different protocols (attack schemes) by which an intruder can organize his attacks in the above model. Our starting point is an attack propagation protocol that organizes attacks along a single traceability path. This protocol, inspired from ideas developed in [5] for a different problem and setting, forms the basis for the development of three other attack schemes, called tree protocols. Using analytic and experimental methods, we first show that for *any* $0 < f < 1$, there exists a g for which any of our attack schemes will achieve a $\Theta(n)$ spread factor with high probability, provided T is sufficiently large. This means that if an intrusion is realized by any of our attack schemes, it will spread, regardless of the security level, to a big part of the network. We also show that the spread and the traceability factors are linearly related. Actually, for our tree protocols this linear relationship holds, with high probability, during the *whole* duration of the attack propagation. This implies that it will not be easy for a detection mechanism to trace the origin of the intruder, since at any time it will have to trace a number of links proportional to the number of nodes captured.

Another interesting issue is to investigate the possibility of a total failure of such attack schemes, namely the possibility that they eventually return to their starting point, not because the intruder is discovered but due to backtracking caused by the limited number of attempts from a specific node. Our combined analytic and experimental methods show that for all four protocols such a possibility is very small.

It is worth mentioning that our analytic and experimental methods are tied to each other. The analytic study of the protocols, where applicable, is rather complicated and gives only lower bounds that are (probably) not tight. Hence, we only have to resort to experiments to get insight as well as a basis of reasonable assumptions to further proceed with the analysis.

2. THE MODEL

We shall describe in more detail the model we consider in this work originating from ideas in [5]. We assume that we are given a network $\mathcal{N}$ consisting of a set V of n nodes. We further assume that some physical network infrastructure exists underlying $\mathcal{N}$ and its specific topology is not a concern of the model. The nodes of $\mathcal{N}$ can be in two states: *awake* and *sleeping.* An awake node may (or may not) possess a special token. Awake nodes possessing a token are called *active.* Only active nodes are allowed to perform attacks. An *attack* is an attempt from an active node to break the security of some other node in $\mathcal{N}$. The active nodes constitute the "frontier" of the awake nodes which issue attacks. Initially, exactly one node x_0 is awake and active (representing the original position of the intruder), and the rest are considered sleeping. Each node is allowed a maximum, but fixed, number g of attacks for establishing a virtual link with some other node of $\mathcal{N}$. Attacks from each active node are executed one-by-one and each may independently fail with probability $0 < f < 1$. If an attack does not fail, then a randomly chosen node v from V is returned. If v is sleeping, then a virtual link to v (through some virtual network channel) is established and the attack is considered successful. That is, established links represent successful attacks. Once a link is established, the sending node is notified and the receiving node (i.e., v) becomes awake. The model allows repetitions, i.e., the same node may be returned in two different attacks which did not fail (but obviously in such a case the attack is not considered successful).

An *attack scheme* in such a model is a distributed computation protocol which specifies how the active nodes are created as the intrusion proceeds. The protocol also specifies information exchange between active and awake nodes. Clearly, this information exchange is carried out along the established links. It is a duty of the protocol to maintain information about the currently established links.

3. THE ATTACK SCHEMES AND THEIR ANALYSIS

A protocol for attack propagation in a network $\mathcal{N}$ according to our model can be derived by extending ideas developed in [5]. In particular, attack propagation can be achieved by a suitable attack scheme which incrementally (link by link) extends and maintains a traceability path of attacked systems using a protocol that comes in two versions (for the sake of technical analysis). The first version assumes the existence of a special capability that restarts the protocol in the case of a total failure and which always succeeds. The second version rectifies this strong assumption by basically simulating this special capability and hence making it unnecessary. In our setting, we shall refer to these two versions as protocol P_1 and P_2 respectively.

We will assume for P_1 that there exists a special attack $A(x, U)$ which can be issued by any active node x to a subset $U \subseteq V$ of sleeping nodes and that always succeeds. We shall call $A(x, U)$ the *special successful attack*. After $A(x, U)$ has been issued, a node x' of U is selected randomly, is made active, and the protocol restarts execution in the subnetwork consisting of nodes in $U - \{x'\}$. Later we shall show how P_2 simulates this special successful attack and makes it unnecessary.

Both protocols assume that there is only one active node, i.e., there is only a single token possessed by some awake node, representing the current position from which attacks are issued. The particular awake node possessing the token is determined by the protocol. Protocol P_2 is the starting point of our study. Based on that, we develop three other variants. All protocols are described in the remainder of this section.

3.1 The basic protocol P_1

In protocol P_1 the g attacks per node are grouped into two equally sized sets, called the *green* and the *red* set, respectively; that is, $g = 2\lambda$ where λ is the cardinality of each set (green or red). The separation of the attacks into two sets is done only for the sake of technical analysis, to provide stochastic independence. The main idea of P_1 is to organize the attacks along the traceability path of attacked systems. The protocol tries initially to establish a (long) traceability path, link by link, using only the green attacks. Each attack is issued from the last node in the path which is the active node (i.e., it possesses the token). An attack is considered successful if a sleeping node is returned which will now become the new last node of the path and gets the token. When attack propagation, i.e., extension of the constructed path, using green attacks is not possible, then the red set of attacks is used. If extension to a new node is established, then P_1 passes the token to that node and continues from there using its green attacks. Otherwise, P_1 backtracks to the first node whose red attacks have not been used yet and (after passing the token) tries to extend the path from that node using the red attacks.

Less informally, P_1 performs a number of logical steps. At the end of the k-th logical step the protocol maintains a triple (Π_k, U_k, R_k), where Π_k is the traceability path constructed so far, U_k is the set of sleeping nodes, and R_k is the set of *red nodes*, i.e., of those awake nodes whose green attacks have all been used. Initially $k = 0$, $\Pi_0 = \{x_0\}$ (the first active node), $U_0 = V - \{x_0\}$, and $R_0 = \emptyset$. Let x_k (resp. z_k) be the first (resp. last) node of Π_k; z_k is assumed to hold the special token denoting the node issuing attacks. P_1 performs the following logical step until $U_k = \emptyset$ or all nodes have exhausted their attacks. There are two cases to consider depending on whether z_k belongs to R_k or not.

Case 1: $z_k \notin R_k$. Node z_k tries its green attacks one-by-one. Every such attack either fails or a random node v is returned. If $v \in U_k$, then the traceability path is extended by setting $z_{k+1} = v$, $\Pi_{k+1} = \Pi_k \cup \{z_{k+1}\}$, $U_{k+1} = U_k - \{z_{k+1}\}$, $R_{k+1} = R_k$, and the token is passed to z_{k+1}. Otherwise, $\Pi_{k+1} = \Pi_k$, $U_{k+1} = U_k$, $R_{k+1} = R_k \cup \{z_k\}$, and z_k (the node having the token) informs the nodes in the current path about the new triple $(\Pi_{k+1}, U_{k+1}, R_{k+1})$.

Case 2: $z_k \in R_k$. Node z_k tries its red attacks one-by-one. Every such attack either fails or a random node v is returned. If $v \in U_k$, then the traceability path is extended by setting $z_{k+1} = v$, $\Pi_{k+1} = \Pi_k \cup \{z_{k+1}\}$, $U_{k+1} = U_k - \{z_{k+1}\}$, $R_{k+1} = R_k$, and the token is passed to z_{k+1}. Otherwise, z_k sends a message backwards in Π_k to find the first node whose red attacks have not been tried yet. Let us call this node z_{k+1}; this node also receives the token from z_k. Now, Π_{k+1} is the subpath of Π_k from x_k up to z_{k+1}, $U_{k+1} = U_k$, $R_{k+1} = R_k \cup \{z_{k+1}\}$, and z_{k+1} informs the nodes in the current path about the new triple $(\Pi_{k+1}, U_{k+1}, R_{k+1})$. In the special case where $z_{k+1} = x_k$ (i.e., when the traceability path has shrunk to a single node) and all of its red attacks have been unsuccessfully tried, then z_{k+1} sends the special attack $A(z_{k+1}, U_k)$. Subsequently, P_1 starts its execution from a new active node x_{k+1}, i.e., $\Pi_{k+1} = \{x_{k+1}\}$, $U_{k+1} = U_k - \{x_{k+1}\}$, and $R_{k+1} = R_k$.

3.2 Structural properties of P_1

We start with the structural properties of P_1, as it will form the basis of our analysis regarding the rest of the protocols. We shall prove that P_1 can achieve a $\Theta(n)$ spread and traceability factor and that the expected number of special successful attacks to restart the protocol is constant.

Let $\ell_k = |\Pi_k| - 1$ (length of Π_k), $u_k = |U_k|$, and $r_k = |R_k|$. Clearly, $\ell_0 = 0$, $u_0 = n - 1$, and $r_0 = 0$, which implies that $n - u_0 + r_0 = 1$. Notice that for P_1, we have $n_{P_1}(k) = n - u_k$ and $\ell_{P_1}(k) = \ell_k$. In each logical step k, P_1 increases either

$n - u_k$ or r_k by at most one. Consequently, $(n - u_k) + r_k \le (n - u_{k-1}) + r_{k-1} + 1$ and inductively we get

$$(n - u_k) + r_k \le k + 1. \tag{1}$$

From the description of the protocol, it is clear that

$$V - U_k - \{v : v \in \Pi_k\} \subseteq R_k$$

which implies that $\ell_k = |\Pi_k| - 1 \ge n - u_k - r_k - 1 \ge n - u_k - (k + 1 - (n - u_k)) - 1$, giving that

$$\ell_k \ge 2(n - u_k) - k - 2. \tag{2}$$

Let $\mathcal{A}_i$ be the set of attacks issued at step i. Then, $H_k = \{\Pi_i, U_i, R_i, \mathcal{A}_i\}_{i=0}^{k}$ represents the history of the protocol up to step k. The probability of failing to extend the path is

$$\begin{aligned}\Pr[u_{k+1} = j | u_k = j, H_k = H] &= \\ &= \sum_{t=0}^{\lambda} \binom{\lambda}{t} \left(\frac{u_k}{n}\right)^t f^t \left(1 - \frac{u_k}{n}\right)^{\lambda - t} \\ &= \left(1 - \frac{u_k}{n}(1 - f)\right)^{\lambda} \end{aligned} \tag{3}$$

for all k, j and H. Since this probability depends only on u_k, and since u_{k+1} is either u_k or $u_k - 1$, it follows that the sequence $\{u_k\}$ is a Markov chain. By considering the sojourn times of u_k in each state of the chain and by working as in [5], we can prove the following.

THEOREM 1. *Let $\lambda(1 - f) \ge 4 \ln 2$ and $r = \lambda(1 - f)/n$. Then for protocol P_1 $\exists$ time $t_0 = n - e^{-rn}/r + o(n)$ such that if $T \ge t_0$, then $n_{P_1}(T) \ge n(1 - 2\ln 2/g(1 - f))$ and $\ell_{P_1}(T) \ge n(1 - 4 \ln 2/g(1 - f))$ with probability tending to 1.*

In order to find the expected number of special successful attacks needed by protocol P_1 to restart, we model the stochastic process ℓ_k as a random walk in one dimension. Let p be the probability that ℓ_k is extended by 1 (via green attacks) and let $q = 1 - p$ be the probability that ℓ_k fails to extend and shrinks to the previous node (of unused red attacks). Then, by Eq. (3) $p = 1 - (1 - \frac{u_k}{n}(1 - f))^{\lambda}$. Let a *round* of P_1 be the time period until the protocol issues a special successful attack in order to restart.

Consider the first $\tau_0 = \alpha \log n$ (α is a constant) green attacks of P_1, and let T_0 be the time interval of these attacks. We can prove the following.

LEMMA 1. *During the time interval T_0, the established traceability path has length at least $(1 - \gamma)\frac{7}{8}\alpha \log n$ ($\alpha \ge 2$) with probability at least $1 - n^{-\frac{\gamma^2}{2}\frac{7}{8}\alpha}$ for any $\gamma \in (0, 1)$.*

PROOF. Since $u_0 = n - 1$ it follows, even if all green attacks were successful, that $u_{\tau_0} \ge n - 1 - \alpha \log n$, i.e., $u_{\tau_0} \ge \beta n$ for some $\beta < 1$. For the time interval T_0, we then have

$$\begin{aligned} p = 1 - \left(1 - \frac{u_k}{n}(1 - f)\right)^{\lambda} \\ \ge \; 1 - (1 - \beta(1 - f))^{\lambda} \ge 1 - \exp(-\beta\lambda(1 - f)) \end{aligned}$$

Then the length of the path established at time T_0 is at least the number of successes in the binomial distribution $B(\tau_0, p)$, i.e., at least $(1 - \gamma)\tau_0 p$ with probability at least $1 - \exp(-\frac{\gamma^2}{2}\tau_0 p)$ for any $\gamma \in (0, 1)$ (due to the Chernoff bound). But $\tau_0 p \ge \alpha \log n \cdot (1 - \exp(-\beta\lambda(1 - f))$ and since $\lambda(1 - f) \ge 4 \ln 2$, we have $\tau_0 p \ge \alpha \log n \cdot (1 - 2^{-4\beta})$ for some $\beta < 1$, i.e., $\tau_0 p \ge \frac{7}{8}\alpha \log n$. $\square$

Let E_1 be the event that the path length established at T_0 is at least $l_0 = (1 - \gamma)\frac{7}{8}\alpha \log n$. Consider the sequence of rounds, R, at which $u_k \ge \frac{n}{2}$. Then, for these rounds

$$q = \left(1 - \frac{u_k}{n}(1 - f)\right)^{\lambda} \le \left(1 - \frac{1 - f}{2}\right)^{\lambda} \le \exp\left(-\frac{\lambda(1 - f)}{2}\right)$$

Since $\lambda(1 - f) \ge 4 \ln 2$, we get $q \le 2^{-2} = \frac{1}{4} \Rightarrow p \ge \frac{3}{4}$.

Let us now condition on the event E_1. For any path length $l \ge l_0$ consider the path of length l' derived by "shortcutting" all nodes whose red attacks have been tried (i.e., skip each such node and connect the node immediately before and after it). From the above, the length l' is at least the number of successes in the binomial distribution $B(l, \frac{3}{4})$ which, for any $\gamma' \in (0, 1)$, is at least $(1 - \gamma')\frac{3}{4}l$ with probability at least $1 - \exp\left(-\frac{\gamma'^2}{2}\frac{3}{4}l\right)$. This implies the following.

LEMMA 2. *For the sequence of rounds R and given the event E_1, the length l' is at least $(1 - \gamma')\frac{3}{4}l$ with probability at least $1 - \exp\left(-\frac{\gamma'^2}{2}\frac{3}{4}(1 - \gamma)\frac{7}{8}\alpha \log n\right) \ge 1 - n^{-\alpha'}$, $\alpha' \ge 3$, for any $\gamma' \in (0, 1)$.*

Let E_2 be the event that, conditioned on E_1, $l' \ge (1 - \gamma')\frac{3}{4}l$. Now let us condition on both E_1 and E_2. Then, for the sequence R of rounds and conditioned on events E_1, E_2, the process of the path length dominates stochastically a random walk W of length w in the integers $[0, \infty)$. Initially, $w = 0$ and the probability of a unit move to the right equals $\frac{3}{4}$ while the probability of a unit move to the left equals $\frac{1}{4}$. Stochastic dominance implies that $\forall x \ge 0$, $\Pr\{l' \ge x\} \ge \Pr\{w \ge x\}$. From [2], it follows that the random walk W in $\frac{n}{2}$ steps achieves $\Theta\left(\frac{n}{2}\right)$ length and its number of returns to the origin is bounded above by a constant, with probability at least $1 - \frac{1}{n}$. Thus:

LEMMA 3. *Given events E_1, E_2 and for the sequence R, the number of times P_1 will return to the origin of the established traceability path is bounded above by a constant c, with probability at least $1 - \frac{1}{n}$.*

However, the expected number of special successful attacks of P_1 is the expected number, y, of returns to the origin. By Lemmata 2 and 3, we get that

$$\begin{aligned} E(y) \le c\left(1 - \frac{1}{n}\right)\Pr\{E_2|E_1\}\Pr\{E_1\} + \\ + \;\; \frac{n}{2}\,\frac{1}{n}\left(\Pr\{\bar{E_1}\} + \Pr\{\bar{E_2}\}\right) = O(1) \end{aligned}$$

The above discussion is summarized as follows.

THEOREM 2. *The expected number of special successful attacks performed by P_1 in order to restart is bounded above by a constant, provided that $\lambda(1 - f) \ge 4 \ln 2$.*

3.3 The protocol P_2

The result of Theorem 2 allows the development of another protocol P_2 which does not need the special successful attack to restart the protocol. P_2 simply *simulates* this special attack using a constant number of attacks per node.

Protocol P_2 is basically P_1 with the following modifications. It groups the g attacks per node into three equally sized sets, i.e., it now holds that $g = 3\lambda$. The third batch of λ attacks is kept for restarting purposes. The node having the token always stores the maximum (in length) traceability path of attacked systems constructed so far by the protocol. When the path shrinks to a single node, then that node notifies all nodes of the maximum traceability path seen in the past to try to use their third batch of attacks in order to restart the protocol. The properties of P_2 are summarized in the next theorem.

THEOREM 3. *Let $\lambda(1-f) \geq 4\ln 2$. Then: (i) For protocol P_2, $\exists$ time t_0 such that if $T \geq t_0$, then $n_{P_2}(T) \geq n(1 - 6\ln 2/g(1-f))$ and $\ell_{P_1}(T) \geq n(1 - 12\ln 2/g(1-f))$ with probability tending to 1 polynomially fast. (ii) The total failure probability to restart P_2 is $o(1/n)$.*

Notice that the stochastic process by which u_k changes in P_2 is the same as that of protocol P_1, since at each attempt to extend from a node always a batch of λ attacks is used. Hence, part (i) of the above theorem follows by Theorem 1 and by the condition for polynomially fast convergence imposed by Lemmata 1–3. Since this condition imposes that $g(1-f) \geq 12\ln 2$, Theorem 3(i) implies that for any $g > g_o = 12\ln 2/(1-f)$ the traceability factor achieved by P_2 is $\Theta(n)$, with high probability. For the proof of part (ii), observe that the total number of attacks that can be used for restart purposes is $\lambda \cdot L$, where L is the length of the maximum traceability path. By Lemma 1, this is with high probability at least $\delta \log n$ for any constant δ above a certain value. The probability of failure to restart the protocol is then just the probability of these extra attacks all failing to hit U_k. But this is at most

$$\left(1 - \frac{u_k}{n}(1-f)\right)^{\delta \log n} \leq e^{-\frac{u_k}{n}(1-f)\delta \log n}$$

Since at time k we have that $u_k \geq n - k$, then for $k = t_0$ (cf. Theorem 1), we get that

$$u_k \geq \frac{e^{-rn}}{r} - o(n) \geq \frac{n}{2\lambda(1-f)e^{\lambda(1-f)}}$$

Hence, the failure probability to restart the protocol is less than

$$e^{-(\delta \log n)/(2\lambda e^{\lambda(1-f)})} \leq n^{-\delta/(2\lambda e^{\lambda(1-f)} \ln 2)}$$

which is at most $\frac{1}{n^2}$ by choosing $\delta > 4\lambda e^{\lambda(1-f)} \ln 2$. Consequently, the total failure probability to restart the protocol is $o\left(\frac{1}{n}\right)$.

To implement P_2 a copy of the maximum traceability path seen so far is maintained at all times. Upon a successful attack the maximum traceability path is copied back to the original path and the protocol continues. We shall refer to the implementation of P_2 as `Original`.

3.4 Three new variants – the tree protocols

Instead of keeping the third batch of λ attacks just for restarting P_1 – and thus getting P_2 – we could use them during the path extension process in a more sophisticated way.

Let us first look more carefully on the execution of protocol P_2. P_2 constructs incrementally the graph G whose vertex set is V and whose edges correspond to successful attacks between nodes. The graph G consists of isolated vertices (corresponding to the sleeping nodes), and of a single connected component Δ whose vertices are the awake nodes and whose edges are the edges of G. Since the edges in G represent successful attacks, i.e., attacks from an active to a sleeping node, Δ is clearly a tree. We designate x_0 as the root of Δ.

The new variants are based on the idea that instead of keeping in P_2 only the maximum traceability path constructed, just store the whole tree Δ. Then, consider various orders of the nodes in Δ for path extension using their third batch of attacks. The different orders will specify the different ways the intruder can use to organize his attacks. Clearly, the maximum traceability path constructed by P_2 is the path of maximum depth in Δ. Under this perspective, P_2 can be viewed as simply considering, for path extension, only the nodes of this maximum depth path. If extensions from those nodes fail, then P_2 stops. Hence, it is natural to try to exploit possible expansions from all nodes in Δ, because in such a case the attack has a bigger front to expand compared to that of a single path.

Our three new variants start (like P_2) by emulating P_1 up to the point where all nodes in the current path have exhausted their green and red attacks (the path has shrunk to a single node) and by constructing on the fly the tree Δ. The difference among them and with P_2 lies in the order the nodes of Δ are considered for path extension.

Our first variant, called `rDFS1`, performs a kind of "reverse" DFS on Δ starting with a path of maximum depth and tries to extend the path from its leaf z using the third batch of attacks. If an attack is successful, that is, the returned node v belongs to U_k, then the protocol proceeds as in P_1: the path is extended, v is equipped with g attacks and takes the token. If all attacks fail, then `rDFS1` backtracks to the parent u of z and repeats the process (i.e., tries to extend the path from u). If during this process all nodes in the path (except for x_0) exhaust their third batch of attacks, then the protocol is restarted from the next largest path in Δ. The whole process is repeated until either $U_k = \emptyset$ or all nodes have exhausted their attacks. At any time during the execution of the protocol the tree Δ is maintained.

Note that as protocol `rDFS1` moves backwards on a path from a leaf z to x_0 and fails to extend the path from an internal node v, the nodes in the subtree rooted at v will be considered at some later point of time by the protocol depending on the overall length of the path to which they belong. An alternative order to consider these nodes is provided by our second variant, called `rDFS2`. It is similar to `rDFS1` except that when the protocol moves backwards to an internal node v, it doesn't try to extend the path from v but instead moves to the leaf z' belonging to a maximum depth path in the subtree of Δ rooted at v and tries to extend the path from z'.

Our third variant, called `rBFS`, is similar in spirit with the above two, but it employs a different strategy in the way the nodes of Δ are considered for extending the path. Protocol `rBFS` performs a kind of "reverse" BFS on Δ starting with the nodes of maximum depth d. It tries to extend the path by considering all such nodes one-by-one (each node uses its third batch of attacks). If an attack is successful, then the protocol proceeds as in P_1: the path is extended, d is increased by one, the new node returned is equipped with g attacks and takes the token. If all attacks in all nodes at level d fail, then `rBFS` repeats the process with all nodes at the tree level $d-1$. The whole process is repeated until either $U_k = \emptyset$ or all nodes in Δ have exhausted their attacks.

In the following, we shall refer to the above protocols as *tree protocols*. Notice that all protocols can be easily implemented in a distributed way. If the node z_k having the token succeeds, then it passes it along with Δ to the next node z_{k+1}. Otherwise, it passes the token and Δ to its parent, say u. Along with Δ, z_k passes information about those nodes which had the token in the past. Depending on the protocol which is currently executed, u either starts issuing attacks, or simply forwards the token and Δ to the appropriate node.

Despite their apparent similarity, we investigate in the remainder of the paper all tree protocols, since it turns out that they either differ in terms of analysis or in terms of experimental behaviour.

3.5 Structural analysis of the tree protocols

We now turn to the analysis of the tree protocols. Notice that the stochastic process by which u_k changes in a tree protocol S is again the same as that of protocol P_1, since at each attempt to extend from a node always a batch of λ attacks is used. Hence, $n_S(k) = n - u_k = |\Delta_k|$, where Δ_k is Δ at time k. Also, $\ell_S(k) = \ell_k$, where ℓ_k is the length of the traceability path. Analogously to the analysis of P_1 (Section 3.2), we would like to achieve a lower bound on ℓ_k as a function of n, k and u_k.

Let Y_k be the subset of red nodes that are reused to extend the path via their third batch of attacks in the tree protocols (i.e., each node in Y_k has already used all of its red attacks), and let $y_k = |Y_k|$. Arguing as before, we get $V - U_k - \{v : v \in \Pi_k\} \subseteq R_k - Y_k$ which implies that $\ell_k \geq n - u_k - r_k - 1 + y_k$. Combining this with Eq. (1), we obtain

$$\ell_k \geq 2(n - u_k) - (k+2) + y_k \qquad (4)$$

Comparing the above equation with (2), we see that the tree protocols add (at least) the quantity y_k to the path length achieved. Obtaining a precise estimation for y_k is very hard, but fortunately we can proceed in an alternative way. Recall that one of our interests is to study the relationship between ℓ_k and $|\Delta_k|$. Before diving into an analytic study, we performed a series of experiments to obtain it experimentally. This would give us an indication and, as we shall see later, provide us with a basis for some reasonable assumptions required by the analysis.

Our experiments clearly indicated that there is a constant $1/2 < a < 1$ (actually $a \geq 0.68$) such that $\ell_k \geq a|\Delta_k|$ (see Figure 1), provided that g satisfies $g(1-f) \geq 12 \ln 2$ as required by the analysis[1]. This experimental result can be verified analytically in the case of rDFS1 as we show next.

We first consider the situation where rDFS1 proceeds successfully (i.e., emulates P_1) without using any of the third batch of attacks. Let a *green node* be an awake node whose green set of attacks has not been exhausted. By working in a way similar to that of the proof of Lemma 1, we can establish the following.

LEMMA 4. *Starting from a green node and emulating P_1, protocol* rDFS1 *will extend the traceability path to an additional length $\ell^* = \Theta(\log n)$, without using any of the third batch of attacks, with probability at least $1 - n^{-c}$ (for some constant $c \geq 2$), provided that $\lambda(1-f) \geq 4 \ln 2$.*

[1]We have observed, however, that $a > 1/2$ even for smaller values of g.

Let us now consider what happens when the protocol is forced to use the third batch of attacks. Then, the current traceability path will "backtrack" (because of failures in the third batch) until a node extends again to another direction. The backtracking probability is at most

$$q_k = \left(1 - \frac{u_k}{n}(1-f)\right)^\lambda$$

which is a constant q (independent of n) as far as $u_k = \Theta(n)$. But in such a case the backtracking process is dominated by a geometric process of a constant backtracking probability q for each backtracking step. The probability that the additional path ℓ^* (gained by the sequence of extensions) will then drop to $\ell^*/2$ is at most $q^{\ell^*/2} = n^{-c_1}$ (c_1 can be selected to be at least 2 by choosing constants as in the proof of Lemma 1).

We shall use the above fact along with Lemma 4 to establish the desired relationship between ℓ_k and $|\Delta_k|$. Let us condition on all times during which $u_k = \Theta(n)$ and let us consider the i-th part of the process of extending the path to an additional length ℓ^* and then backtrack. Call this i-th part π_i. Let A_i be the event "the extension to an additional length ℓ^* gives $\ell^* = \Theta(\log n)$ and the backtracking reduces it to at most $\ell^*/2$". Then, we have that

$$\Pr[A_i] = (1 - n^{-c})(1 - n^{-c_1}) \geq 1 - 2n^{-c_2}$$

where $c_2 \geq 2$. Now, consider $n/2\log n$ π_i's. The probability that at least one A_i does not hold in this sequence is

$$\Pr[\exists \overline{A_i}] \leq \sum_i \Pr[\overline{A_i}] \leq \left(\frac{n}{2\log n}\right)(2n^{-c_2}) \leq \frac{1}{n}.$$

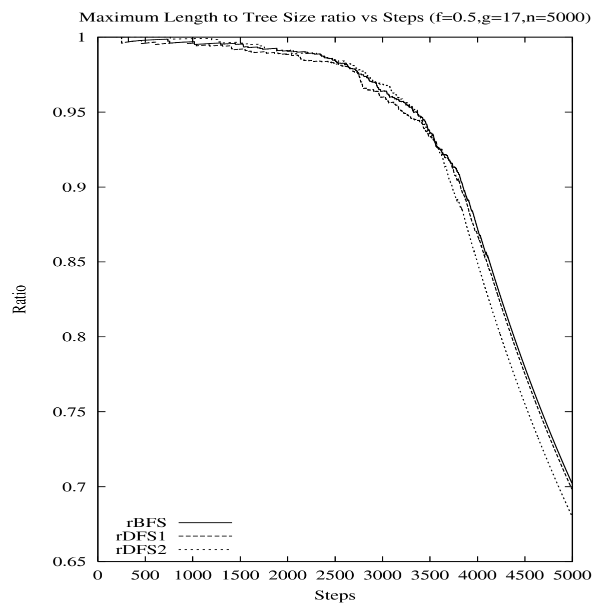
Hence, all events A_i hold simultaneously with probability at least $1 - 1/n$, i.e., the backtracking in each π_i at most halves the extension of the path achieved within π_i. Clearly, the cumulative path extensions in all π_i's equals $|\Delta_k|$. Since the cumulative backtrackings in all π_i's reduce $|\Delta_k|$ by at most a half, we conclude that $\ell_k \geq |\Delta_k|/2$. The preceding discussion establishes the following.

THEOREM 4. *Let $\lambda(1-f) \geq 4 \ln 2$. Then, with probability at least $1 - 1/n$, $\exists$ a time step t_0 for protocol* rDFS1 *such that: (i) $\forall 0 \leq k \leq t_0$, $\ell_{\texttt{rDFS1}}(k)/n_{\texttt{rDFS1}}(k) \geq 1/2$; (ii) $\forall T \geq t_0$, $n_{\texttt{rDFS1}}(T) = \Theta(n) = \ell_{\texttt{rDFS1}}(T)$.*

Unfortunately, the above analysis does not carry over to protocols rDFS2 and rBFS. However, our experimental results (cf. Figure 1) indicate that they perform at least as good as rDFS1. Hence, our experimental evidence and the above theoretical analysis lead us to the following conclusion.

EXPERIMENTAL CONCLUSION 1. *With high probability, $\exists$ a time step t_0 for any tree protocol S such that: (i) there is a constant a, $1/2 < a < 1$, such that $\forall 0 \leq k \leq t_0$, $\ell_S(k)/n_S(k) \geq a$; (ii) $\forall T \geq t_0$, $n_S(T) = \Theta(n) = \ell_S(T)$.*

We shall use the above conclusion as a (reasonable) hypothesis in order to investigate analytically the relationship between the traceability factor achieved by the tree protocols and the one achieved by P_1 or P_2.



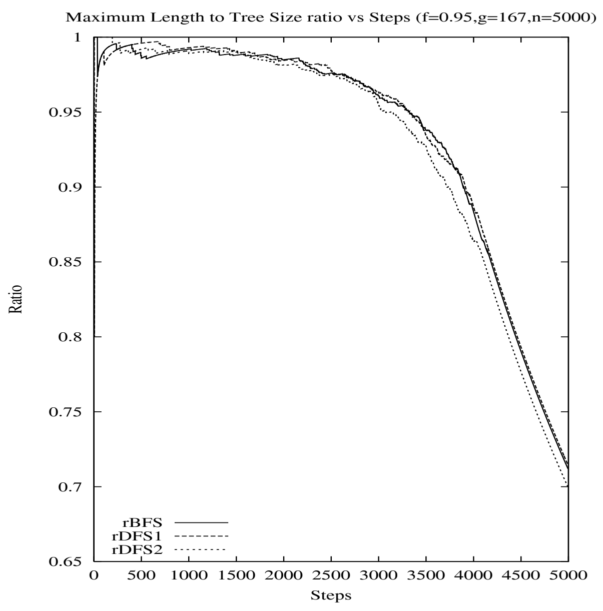


Figure 1: The ratio $\ell_k/|\Delta_k|$ versus the number of steps for which path extension is achieved.

Recall that $n_S(k) = n - u_k$ and that the stochastic process by which u_k changes is the same as that of protocol P_1. We will now condition on the event "$\forall k, \ell_k \geq a(n-u_k)$" and provide evidence that each of the tree protocols achieves slightly larger traceability factors than that of P_1 and P_2. This is also verified by our experiments.

THEOREM 5. *The tree protocols achieve a traceability factor which is at least as large as the one achieved by protocols P_1 and P_2.*

PROOF. Let $\tilde{u}_k$ be any lower bound on u_k holding with high probability. We only have to compare the progress of two expressions, namely $e_1(k) = 2(n-\tilde{u}_k) - (k+2)$ for P_1, P_2, and $e_2(k) = a(n-\tilde{u}_k)$, for the tree protocols.

Let $\lambda(1-f) \geq 4\ln 2$, $r = \frac{\lambda}{n}(1-f)$, and $j = \lceil n \ln 2/\lambda(1-f)\rceil$. Then, we have from Theorem 1 that the first time step k at which the path length becomes $\Theta(n)$ is, with high probability, $n - e^{-rn}/r + o(n)$. For this value of k, the above expressions become

$$e_1(k) = 2\left(n - n\ln 2/\lambda(1-f)\right) - \left(n - \frac{e^{-rn}}{r} + o(n)\right)$$

and

$$e_2(k) = a\left(n - n\ln 2/\lambda(1-f)\right).$$

Since $\lambda(1-f) \geq 4\ln 2$, we get that $e_2(k) \geq e_1(k)$ for any $a > 1/2$. $\square$

Notice that this is only a lower bound comparison, since both our analysis for P_1 and its extension to the tree protocols are not giving a precise value for the traceability factor, but only lower bounds.

4. EXPERIMENTAL RESULTS

As it is customary with such protocols, our implementations were done on a simulation environment. In the implementations we tried to exploit the full strength of the protocols. The (pure) protocol statements discard the remains of batches in attacks which succeeded to extend the traceability path, in the case where a part of the batch was processed. In our implementation we used also these remains. The implementations of all protocols were written in `C++` and were run on a SUN Enterprise 450 running Solaris 2.7 and having 1.2GB of main memory.

We performed an extensive set of experiments using different types and sizes of inputs. Due to space limitations, we report results with representative values of n (500 and 5000), and of the critical parameters f and g. We consider two values for f, namely 0.5 (network of average security level), and 0.95 (highly secure network), and two values for g, namely 10 and 50. The conducted experiments confirmed our theoretical results and exhibited the practicality of the new protocols.

We measured the length L of the maximum traceability path achieved by some protocol either as a function of g given a fixed value for f (Figure 3), or as a function of f given some fixed value of g (Figure 4). The `Theory` curve shows the bound expected from Theorem 3. It is interesting to observe that for small values of g, all protocols achieve a much better path length than `Theory`, even when f gets larger. When f is relatively small (e.g., $f = 0.5$), all protocols achieve almost the same path length, mainly because the number of restarts is minuscule. When f gets larger, however, `rBFS` and (surprisingly) in some cases `Original` achieve a better path length and are the most robust (having much less variance than the other protocols).

We observe no big differences w.r.t. the maximum path length achieved by the protocols. This can be explained by the fact that in all experiments we performed the resulted tree Δ was almost always deep and lean. This implies that there are neither big differences among the various orders in which the protocols consider the nodes of Δ, nor big differences between `Original` and the tree protocols, and this carries over to the length of the maximum path. For very large values of f, e.g., $f = 0.95$, we observed a big variance. While it seems difficult to explain this theoretically, we sus-

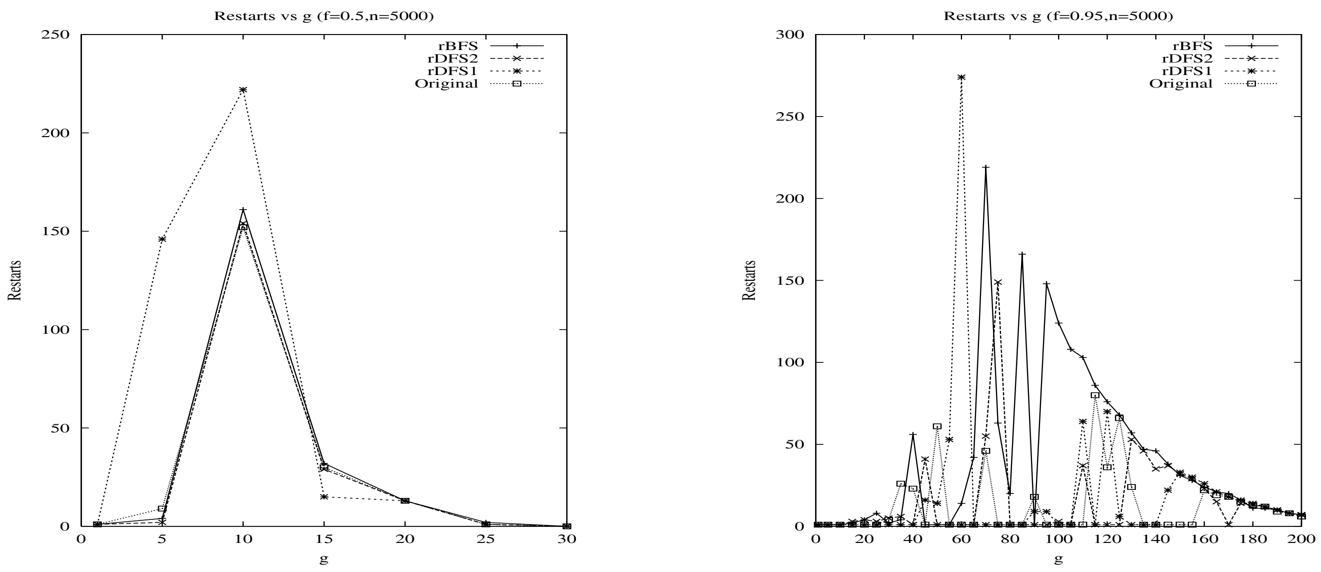


Figure 2: Number of restarts for $f = 0.5$ and $f = 0.95$ ($n = 5000$).

pect that it is due to the following. If the first few nodes fail to issue their attacks (which is highly probable in this case), then the length of the path will be very small, and as a consequence it is highly improbable that the protocol will even be able to restart. On the other hand, once a protocol manages to restart, the achieved path length is much better than the theoretically expected one – even if a relatively larger number of restarts is required (see Figure 2 and the discussion below regarding restarts). We also observe that in such cases the protocols need much less than g_o attacks to achieve a reasonable path length, thus demonstrating their good practical behaviour.

Regarding the number of restarts, our experiments showed that all protocols perform *zero* restarts for any value of n we considered ($n \in [50, 5000]$) when $g > 25$ and relatively small f (e.g., $f = 0.5$); see Figure 2 (left diagram). The number of restarts may vary almost linearly with n as g approaches g_o (see Figure 2), but drops rather quickly as soon as it passes g_o. The big variance observed in the number of restarts (cf. right diagram of Figure 2) for large values of f and when g is below g_o is due to the length of the maximum path achieved by the protocol prior to the first restart. If this length is small, then the protocols perform very few restarts, after which they stop (since all available attacks have been exhausted). Otherwise, the protocols, due to the high probability failure, perform many restarts in order to extend the path. This explains why e.g., `Original` has usually less restarts than `rBFS`.

5. CONCLUDING REMARKS

We investigated the analytic and experimental behaviour of four protocols for the attack propagation problem in networks under a new model for intrusion propagation introduced here. It would be interesting to investigate the spread and traceability factor for any time $t \geq 0$, as well as the cases where the values of f and g vary (under some pattern) with the time, the system, and/or the length of the traceability path achieved.

6. REFERENCES

[1] D. Denning, "Information Warfare and Security", Addison-Wesley, 1999.

[2] W. Feller, "An Introduction to Probability Theory and its Applications", Vol. I, John Wiley, New York, 1968.

[3] J. Howard, "An Analysis of Security Incidents on the Internet 1989-1995", Carnegie Mellon Univ., CERT/CC, 1997, `www.cert.org/research/JHThesis/Start.html`.

[4] J. Kephart and S. White, "Directed-Graph Epidemiological Models of Computer Viruses", IBM Research Report; also, in *Proc. IEEE Symp. on Security and Privacy*, 1991.

[5] S. Nikoletseas and P. Spirakis. "Efficient Communication Establishment in Adverse Communication Environments". In *Proc. ICALP Satellite Workshop on Approximation and Randomized Algorithms in Communication Networks*, 2000.

[6] R. Ostrovsky and M. Yung, "How to Withstand Mobile Virus Attacks", in *Proc. 10th ACM Symp. on Principles of Distributed Computing (PODC'91)*, 1991, pp.51-59.

[7] T. Shimonura and J. Markoff, "Takedown: The Pursuit and Capture of Kevin Mitnick, America's most Wanted Computer Outlaw", Hyperion, NY 1996.

[8] D. Safford, D. Shales, and D. Hess, "The TAMU Security Package: An Ongoing Response to Internet Intruders in an Academic Environment", in *Proc. UNIX Security Symposium IV*, 1993.

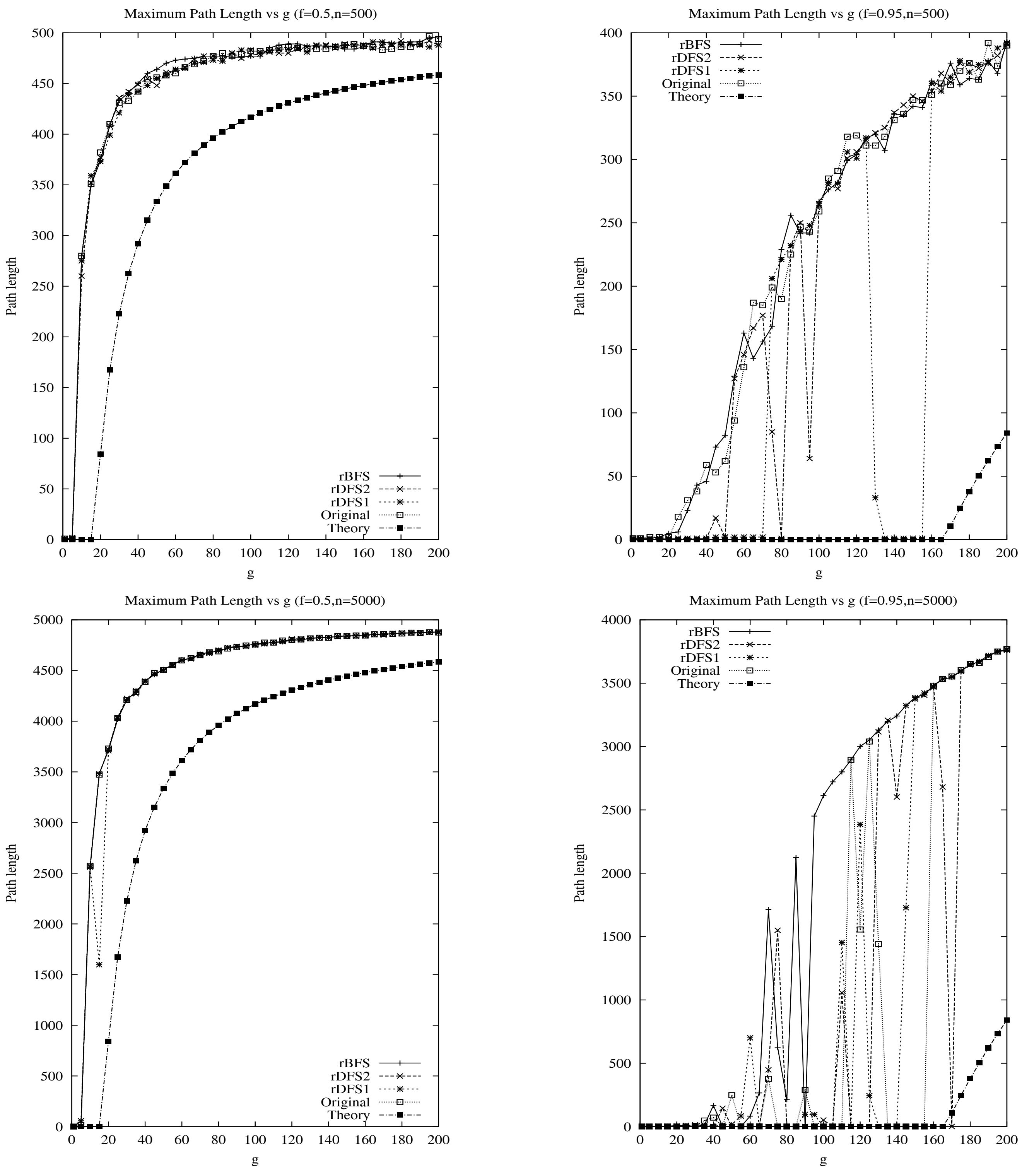


Figure 3: Maximum traceability path length as a function of g for $n = 500$ and $n = 5000$. The granularity of the g-axis is 5.

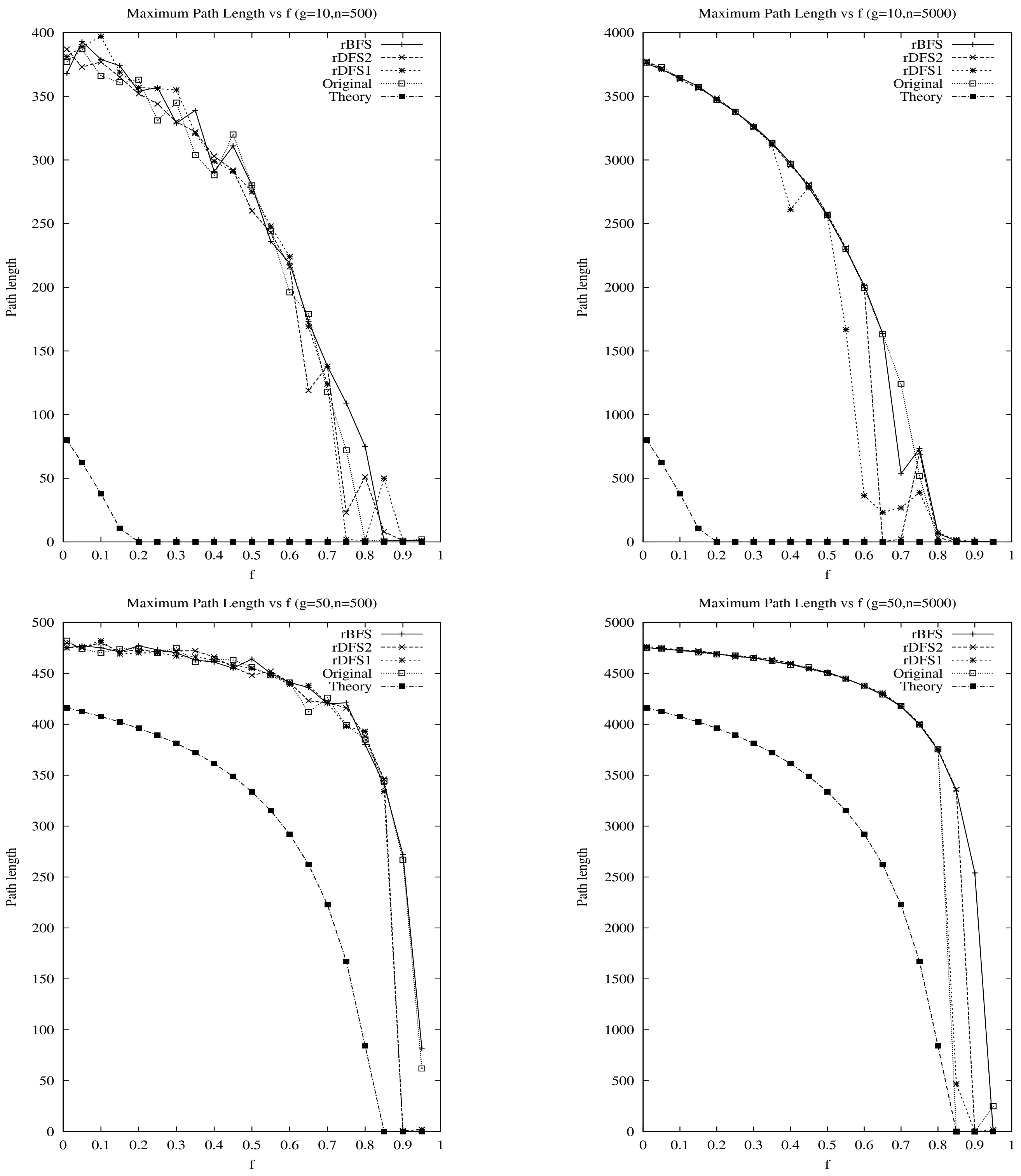


Figure 4: Maximum traceability path length as a function of f. The granularity of the f-axis is 0.05.