# Bayesian-Boolean Logic Security Assessment Model for Malware-Free Intrusions

Aaron Zimba, Hongsong Chen, Zhaoshun Wang
*(Corresponding author: Aaron Zimba)*
Department of Computer Science and Technology, University of Science and Technology Beijing
Haidian District 100083, Beijing, China
(Email: gvsfif@gmail.com)
*(Received Feb. 22, 2017; revised and accepted June 25, 2017)*

## Abstract

Attackers have come to leverage exploits precipitated by system vulnerabilities and lapses by using malware which otherwise tends to be noisy as it generates unusual network traffic and system calls. Such noise is usually captured by intrusion detection systems. Therefore, malware-free intrusions which generate little noise if any at all, are especially attractive to APT actors because they covertly use normal applications making it hard for intrusion detection systems. In this paper, we consider malware-free intrusions by formulating representations of system security states using Boolean logic in the scenario of a backdoor attack utilizing system implementation of pre-authentication services. We further derive, from the generated attack scenarios, a Bayesian security assessment model based on the environmental parameters of the experimental test-bed based on the backdoor attack via RDP-based remote access. The malware-free intrusion based on RDP backdoor attack is successfully run on five different versions of operating systems.

*Keywords: Advanced Persistent Threat (APT); Bayesian Network; Boolean Logic; Malware-Free Intrusion*

## 1 Introduction

Cyber networks today, the Internet inclusive, are plagued with a myriad of attacks all directed against Confidentiality, Integrity and Availability aspects of security. Attacks against these tenets of security, perpetrated by threat actors in the form of insider or outsider attackers [14], are categorized as either targeted or untargeted [5]. Advanced Persistent Threats (APT) belong to the latter classification and are thus carefully crafted owing to their nature since substantial knowledge of the victim is required prior to a successful attack. APT actors seek to maintain a long stealthy presence to further their attacks. Since the ultimate goal is not just to compromise a system and vacate in the shortest time possible but rather uphold the undetected presence feature, mechanisms and techniques employed to achieve the desired intrusion play a vital role. Attackers therefore use complex techniques to compromise systems which include leveraging vulnerabilities in system software, flaw in design and implementation of a security system as well as ignorance of a benign user. APT attackers use special malware with characteristics different from conventional malware in that such malware may hibernate and remain dormant for long periods before initiating the actual attack or beacon back to the Command and Control (C2) for further directives [22]. But the presence of Intrusion Detection Systems (IDS) present a higher chance of discovery against such malware even as its signature and activities might not escape the eye of the IDS. Since the noise generated by malware in form of network activity, issuance of system calls and signature of the malware itself form the basis upon which IDSs detect such malware [12, 16, 18], an ideal intrusion appealing to the attacker would be one that operates outside the realms of the aforementioned detection parameters.

Malware-free intrusions fit well in the above desirable intrusion requirement because they utilize normal system files and processes to covertly achieve their goal. The network activity generated by normal system processes, associated system calls and even their file signature values all fall under the threshold for normal file classification of the IDS. Attackers have therefore come to exploit shortfalls in the security implementation of systems without using malware to attain the desired malware-free intrusion. One such leveraged security shortfall which has seen considerable usage by various APT actors is the accessibility services backdoor [4, 15]. The attack pursued via this route avails the attacker system level access without logging in at all. Access is achieved by invoking corresponding accessibility keystrokes on a compromised system and this access is also possible over the network through RDP-based remote access. We explore two attack vectors emanating from the attack space cast by the aforementioned

backdoor attack. We employ a conceptual finite state automaton to deduce state equations representative of the various system security states relative to the pursued attack vector and use attack tree analysis for formulation of the security assessment model based on Bayesian inference. We carry out experimental attack tests on an IP network with different versions of operating systems and associated security implementations.

This paper is organized as follows: the second section encompasses state model formulation and analysis whilst the experimental test-bed is discussed in the third section. The security assessment model is derived in section four and we conclude the paper in the fifth section.

# 2 Model Formulation and Analysis

The accessibility backdoor considered in this paper affects Windows operating systems from Windows XP to Windows 10. Though we do not include server versions of Windows operating system, we contend that the techniques employed herein are applicable thereto provided the server operating system in contention ships with accessibility services and RDP-based remote access, which is the case by default.

## 2.1 Transitions of System Security States

We construct a conceptual model, as depicted in Figure 1, by employing a finite state machine where the security status of the given system is defined by the state of three attack vectors. The first two attack vectors pursued henceforth are the backdoor implantation variants through which system level access is made available at pre-login before authentication while the third vector is activation of remote access using the system inbuilt Terminal Services.

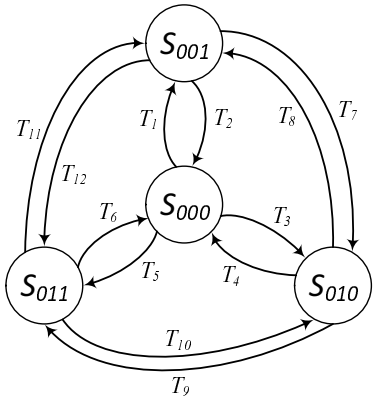


Figure 1: State transition diagram of system security states

The four states of the system are denoted by $S_n$, where $n$ increments by a single binary unit upon successful implementation of a given attack vector. Transitions from one state to another are denoted by $T_n$ and we use binary state encoding for security state assignment and thus denote the states as follows:

- $S_{000}$ - the initial secure state of the system in the absence of pursuance of all the three attack vectors.
- $S_{001}$ - the state of the system when only one of the three attack vectors is successfully pursued.
- $S_{010}$ - the state of the system when any two of the three attack vectors are successfully pursued.
- $S_{011}$ - the state of the system when all the three attack vectors are successfully pursued.

It is evident from Figure 1 that transitions $T_1$, $T_3$ and $T_5$ are induced by successful pursuance of the associated attack vectors thereby inducing state transitions from $S_{000}$ to $S_{001}$, $S_{010}$ and $S_{011}$ respectively. On the contrary $T_2$, $T_4$ and $T_6$ are as a result of mitigating the associated security breaches emanating from the corresponding attack vectors. Transitions $T_7$, $T_9$ and $T_{12}$ are perturbed by an increment in the number of successfully pursued attack vectors whereas transition $T_8$, $T_{10}$ and $T_{11}$ reflect the opposite. It should be noted however that the order in which the attack vectors are pursued, hence the corresponding state transition, is not relevant but rather the fact that the attack vector is pursued unto completion.

Let the result of backdoor implantation by the first attack vector via replacement of accessibility executable binary in the %systemroot% \system32 directory be denoted by the binary variable $\alpha$. And let the result of backdoor implantation by the second attack vector via the system registry be denoted by the binary variable $\beta$. Lastly, let the result of activation of RDP-based remote access via system registry alteration be denoted by the binary variable $\gamma$. Thus with regards these three binary variables, the security status of the system at any given instance can be formulated as a Boolean function:

$$S_n(\alpha, \beta, \gamma),\ where\ \alpha, \beta, \gamma \in \mathbb{N}_2$$

Consequently since $\alpha, \beta, \gamma \in \{0, 1\}$, it follows henceforth that the false values (binary 0) i.e. when the result of backdoor implantation and RDP-based remote access are unsuccessful thereby inducing no security breach are denoted by way of complementation, hence yielding the variables $\bar{\alpha}$, $\bar{\beta}$ and $\bar{\gamma}$. The possible system security states are given by:

$$q = log_2 n \tag{1}$$

where $q, n \in \mathbb{N}^+$ are the state variables and number of states respectively. Therefore the cardinality $S_n$ for the binary variables $\alpha$, $\beta$ and $\gamma$ using Equation (1) is;

$$|S_n| = 8\ where\ n\ is\ \{n : n \in \mathbb{N}^+,\ 0 \leqslant n \leqslant 3\}$$

The 8 states are spread across $S_{000}$, $S_{001}$, $S_{010}$ and $S_{011}$ and we use these to construct the corresponding truth table and k-maps to derive state binary equations representative of the various system security states. Since the input combinations of all states are identical, we use one integrated table instead of four where we only differentiate the output. The resulting truth table of all possible states at any instance is shown in Table 1.

Table 1: Truth table for possible security states

| **Input Bin. Variables** | | | *Output State* $S_n$ |
|---|---|---|---|
| $Var_1\alpha$ | $Var_2\beta$ | $Var_3\gamma$ | $S_{000}/S_{001}/S_{010}/S_{011}$ |
| $F$ | $F$ | $F$ | *1/0/0/0* |
| $F$ | $F$ | $T$ | *0/1/0/0* |
| $F$ | $T$ | $F$ | *0/1/0/0* |
| $F$ | $T$ | $T$ | *0/0/1/0* |
| $T$ | $F$ | $F$ | *0/1/0/0* |
| $T$ | $F$ | $T$ | *0/0/1/0* |
| $T$ | $T$ | $F$ | *0/0/1/0* |
| $T$ | $T$ | $T$ | *0/0/0/1* |

The security requirement of the initial state $S_{000}$ dictates that there be no breach in the system implying that all input variables be false. This implies a conjunctive Boolean AND operation on all the three complemented input variables. To derive the state equation, we employ the K-map in Figure 2 below and apply the Product of Sums (POS) on the dotted groups and Sum Of Products (SOP) on the solid group for equation validation.

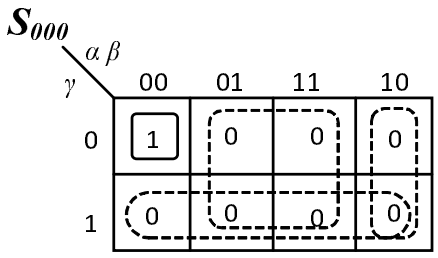


Figure 2: K-map for the single state $S_{000}$

We use the canonical disjunctive normal form of the maxterms corresponding to the dotted groups of *0*s and thus derive the complemented equation for the initial state:

$$\bar{S}_{000}(\alpha,\beta,\gamma) = \alpha\cdot\bar{\beta}+\beta+\gamma$$

Applying the De Morgan theorem and applicable Boolean identities, we derive the Equation (2) representative of this first secure state:

$$\therefore S_{000}(\alpha,\beta,\gamma) = \bar{\alpha}\cdot\bar{\beta}\cdot\bar{\gamma}\ where\ \alpha,\beta,\gamma\in\mathbb{N}_2 \qquad (2)$$

The state Equation (2) validates with one obtained via SOP of the minterms of the solid group.

When one and only one of the three attack vectors is pursued to fruition in the initial state, the security status of the system transitions via $T_1$ from state $S_{000}$ to 3 possible states of $S_{001}$. We use the K-map below in Figure 3 derived from the integrated truth Table 1 to deduce the state equations representative of the new system state.

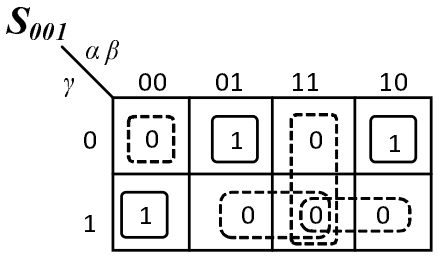


Figure 3: K-map for three possible $S_{001}$ states

We order the $(\alpha,\beta)$ variable pair in our K-maps in the sequence $01\rightarrow 11$ and not $01\rightarrow 10$ so as to avoid race conditions and static hazards. Therefore, in view of the aforementioned, the isolated groups of *1*s and *0*s in our K-maps do not denote "don't care"entries but are rather inputs of the SOP and POS minterms and maxterms respectively. The equations representative of the new state are thus minimized via SOP as:

$$S_{001}(\alpha,\beta,\gamma) = \bar{\alpha}\cdot\beta\cdot\bar{\gamma}+\bar{\beta}\ (\alpha\oplus\gamma)$$

$$S_{001}(\alpha,\beta,\gamma) = \alpha\cdot\bar{\beta}\cdot\bar{\gamma}+\bar{\alpha}\ (\beta\oplus\gamma)$$

$$S_{001}(\alpha,\beta,\gamma) = \bar{\alpha}\cdot\bar{\beta}\cdot\gamma+\bar{\gamma}\ (\alpha\oplus\beta)$$

$$\therefore S_{001}(\alpha,\beta,\gamma) = \{\alpha,\beta,\gamma \mid \bar{\alpha}\cdot\beta\cdot\bar{\gamma}+\alpha\cdot\bar{\beta}\cdot\bar{\gamma}+\bar{\alpha}\cdot\bar{\beta}\cdot\gamma\} \quad (3)$$

The state Equation (3), depicting the three possible states of $S_{001}$, correlates with canonical disjunctive normal form of the POS where the maxterms correspond to the dotted groups of *0*s in Figure 3, hence the validation.

If another attack vector of the remaining two is pursued to completion whilst in state $S_{001}$, the system transitions via $T_7$ to any of the three possible states of $S_{010}$ bringing the sum of successfully pursued attack vectors to 2. The K-map for these three new possible states is shown below in Figure 4.

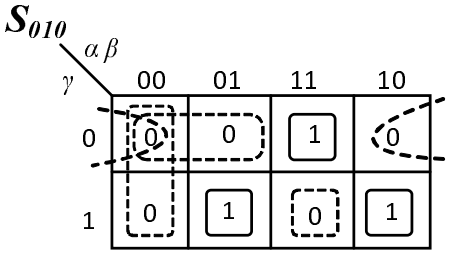


Figure 4: K-map for three possible $S_{010}$ states

The sequence for variable pair ordering on the horizontal axis in Figure 4 likewise follows suit as that of Figure 3. We therefore employ the SOP based on this K-map

to derive the equations representative of the three new possible states:

$$S_{010}(\alpha,\beta,\gamma) = \bar{\alpha}\cdot\beta\cdot\gamma + \alpha\ (\beta\oplus\gamma)$$

$$S_{010}(\alpha,\beta,\gamma) = \alpha\cdot\bar{\beta}\cdot\gamma + \beta\ (\alpha\oplus\gamma)$$

$$S_{010}(\alpha,\beta,\gamma) = \alpha\cdot\beta\cdot\bar{\gamma} + \gamma\ (\alpha\oplus\beta)$$

$$\therefore S_{010}(\alpha,\beta,\gamma) = \{\alpha,\beta,\gamma \mid \bar{\alpha}\cdot\beta\cdot\gamma + \alpha\cdot\bar{\beta}\cdot\gamma + \alpha\cdot\beta\cdot\bar{\gamma}\} \quad (4)$$

The Equation (4) depicts the three possible states with only two attack vectors yielding fruition. Likewise it correlates with its dual obtained by the canonical disjunctive normal form of the POS maxterms, hence the validation.

Now that the system is in a state with two attack vectors pursued to completion hence two system breaches, there only remains one attack vector to be pursued which transitions the system into the final state $S_{011}$. We yet again use a K-map, Figure 5, to derive the state equation representative of this new state.

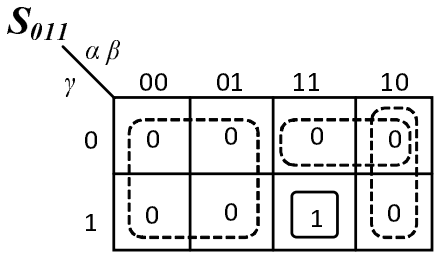


Figure 5: K-map for the final state $S_{011}$

We this time employ POS of the maxterms of the canonical disjunctive normal form of the dotted groups yielding:

$$\bar{S}_{011}(\alpha,\beta,\gamma) = \alpha\cdot\bar{\beta} + \bar{\alpha} + \bar{\gamma}$$

We further minimize the equation with Boolean identities to find the final compact status equation as:

$$S_{011}(\alpha,\beta,\gamma) = \alpha\cdot\beta\cdot\gamma \ where\ \alpha,\beta,\gamma \in \mathbb{N}_2 \quad (5)$$

This final equation is validated from the K-map by computing the SOP of the single solid group of *1*s. It is apparent from Equation (5) that the system cannot transition into any less secure state than $S_{011}$ because all the attack vectors have been exhausted and the equation itself is a Boolean conjunctive AND operation on all the three variables.

According to the derived four state equations, the eight states from Equation (1) are partitioned as follows:

- One state in $S_{000}$ denoted by Equation (2);
- Three states in $S_{001}$ denoted by Equation (3);
- Three states in $S_{010}$ denoted by Equation (4);
- One state in $S_{011}$ denoted by Equation (5).

Notwithstanding the aforementioned, not all the eight states represent a successful attack though they might imply the presence of a breach due to a specific pursued attack vector. We are now therefore tasked to find out which combinations of pursued attack vectors lead to a successful malware-free intrusion attack. We address this task in the proceeding sub-section where we describe the attack model.

## 2.2 Attack Modeling and Analysis

We approach attack modeling based on conceptual units [2,8,17] where such discrete units serve as the basic building blocks of the attack. The threat actor who is an attacking agent, executes a series of actions to obtain assets as pivots for reaching the final goal. Therefore our model comprises four units namely assets, actions, agents and goals.

### 2.2.1 Model Units Formulation

**Assets:** Assets are anything the attacker needs to acquire not only for optimal output but for actualization of the attack itself as well. In our context, assets include but are not limited to information about a victim host such as IP address, open ports and their associated protocols, underlying operating system, service banner information etc. This is the knowledge domain that the attacking agent has of the target in contention.

**Actions:** Actions are steps of sequential phases that constitute an attack. Actions have preconditions which foster acquisition of a sought after asset, that is to say that the outcome of an action is whether the asset is acquired or not. The actions in our setting include initiation of the pursuance of the earlier mentioned attack vectors, finding target hosts in the environment, invocation of console system level access via appropriate keystroke combinations etc.

**Agents:** Agents are the subject of any given attack scenario whose actions are directed towards a specified object. They can broadly be distinguished as human or software actors. The agent in our consideration is a highly skilled technical human actor with a considerable sophistication of stealthiness and non-traceability.

**Goals:** Goals represents a request knowing all action outcomes which in turn complete the associated assets. Goals are differentiated depending on the context, and in ours, goals include establishment of a remote access connection via the RDP protocol provided that the port scan action returned a value that Terminal Services are available on a target host.

Having defined the components of our model, we now integrate them into attack tress for modeling and analysis.

#### 2.2.2 Attack Tree Modeling Integration and Analysis

The units of the preceding subsection are integrated into an attack tree [13] for modeling and analysis. Here we describe the backdoor attack against a victim host where the nodes, represented by the model units, require complete execution of children nodes to reach the root node which is the sought after system level access via RDP-based remote access. The nodes are either conjunctive AND nodes or disjunctive OR nodes. All children of an AND node need to return a true value if the parent node is to execute successfully while only one or more of an OR node need to be true to accomplish the same. The resultant attack tree is shown in Figure 6 below.

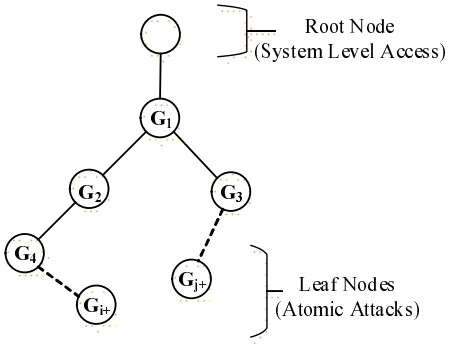


Figure 6: Attack Tree for backdoor attack

The root node is denoted by $G_0$ which is acquisition of system level access over the network. This is achieved by pursuing the sub-goals originating from the lower leaves. The rest of the nodes are denoted as follows: $G_1$ –RDP Remote Access Activation, $G_2$ –SystemRoot Bin. Replacement, $G_3$ –Registry Debugger Conf., $G_4$ –Local Privilege Escalation, $G_j+$ denotes a set of nodes decomposed into conjunctive AND or disjunctive OR representing attack model units needed to be engaged if the pursued attack vector of backdoor implantation is that of registry alteration while $G_i+$ denotes those where the backdoor is implanted via binary executable replacement. We deduce the adjacency square matrix $A_G$ of the 5th order from the graph after pruning out Gi+ and $G_j+$:

$$A_G = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \quad (6)$$

We now use Equation (6) of the above matrix rows and columns to derive attack scenarios corresponding to the following paths:

$$P_1 : \{G_4, G_2, G_1, G_0\}$$

$$P_2 : \{G_3, G_1, G_0\}$$

It is evident from the above paths that $P_1$ is longer than $P_2$ because pursuance of the former calls for addressing an additional unit of privilege escalation. However, this does not necessarily imply that $P_2$ is a better path than $P_1$ because the value of the weights of the edges might tell otherwise depending on the metrics used.

The edge $\{G_1, G_0\}$ represents the isthmus of graph implying that failure to actualize an action associated with this edge thwarts the backdoor attack. The action associated with this edge is activation of RDP-based remote access and this corresponds to the $\gamma$ binary variable of Equation (2), (3), (4) and (5). Since the value of $\gamma$ in Equation (2) is definitely false, it follows that the backdoor attack is not feasible and we therefore drop this equation for simulation considerations. Likewise, Equation (3) reduces to the form $S_{001}(\alpha, \beta, \gamma) = \bar{\alpha} \cdot \bar{\beta} \cdot \gamma$ as we drop all the minterms with $\bar{\gamma}$. In the same manner, Equation (4) reduces to $S_{010}(\alpha, \beta, \gamma) = \gamma\ (\alpha \oplus \beta)$ and we use Equation (5) as-is considering that it has no complemented values of $\gamma$. This gives us a system of compact equations:

$$S_n(\alpha, \beta, \gamma) \quad = \quad \left\{ \begin{array}{l} \alpha \cdot \beta \cdot \gamma \\ \gamma\ (\alpha \oplus \beta)\ where\ \alpha, \beta, \gamma \in \mathbb{N}_2 \\ \bar{\alpha} \cdot \bar{\beta} \cdot \gamma \end{array} \right\}$$

We therefore base our simulations on this system of equations in the next section. We test the attack on different combinations of literal and complemented values of the binary variables and XORing where applicable according to the equation.

## 3 Experiment Simulations

The experiment setup consist of two networks; that of the attacker and one where the targeted hosts reside. We build our test-bed environment in Virtual Box where we simulate the internetwork connection and the network attack itself.

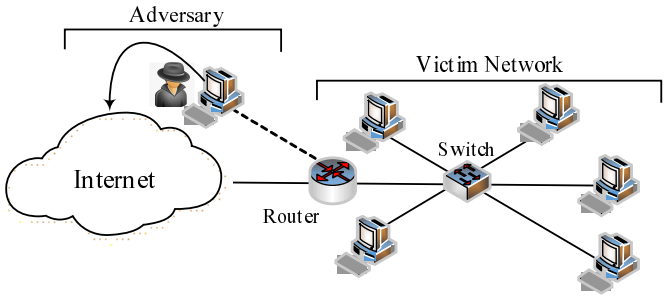


Figure 7: Experiment setup for Malware-free intrusion

The attacking agent defined in the attack model runs on a Linux machine in the first network while the victim hosts running different versions of the Windows operating system reside in the second network. Since Network Level Authentication (NLA) affects the establishment of

RDP session via remote access, we switch it on and off by manipulating the corresponding registry keys. Our experiment setup is shown in Figure 7 above.

We implant the backdoor by file switching method as per $\alpha$ definition and registry debugger configuration as per $\beta$ definition. We further activate Terminal Services, the service responsible for RDP-based remote access, by changing the hexadecimal value of *fDenyTSConnections* in the registry from 1 to 0 in the registry path $HKEY\_LOCAL\_MACHINE\backslash SYSTEM\backslash CurrentControlSet\backslash Control\backslash TerminalServer$. We likewise switch on and off NLA for different combinations of $\alpha, \beta, \gamma$ for registry path HKEY_LOCAL_MACHINE \SYSTEM \CurrentControlSet\Control \Terminal Server\WinStations \RDP-Tcp by changing the key hexadecimal value of UserAuthentication from 0 to 1. We test the attack with different combinations of these parameters and the results are shown in Table 2 below.

Table 2: Attack status with different parameters

| $S_n$ | FileSw | RegDebug | RDP Reg | Status |
|---|---|---|---|---|
| $S_{001}$ | X | X | OFF | *NIL* |
| $S_{001}$ | 0 | 0 | ON | *NIL* |
| $S_{010}$ | X | X | OFF | *NIL* |
| $S_{010}$ | 1 | 0 | ON | *SUCCESS* |
| $S_{010}$ | 0 | 1 | ON | *SUCCESS* |
| $S_{011}$ | 1 | 1 | ON | *SUCCESS* |

It is evident from Table 2 that the core of the success of this malware-free intrusion attack vector is RDP. When RDP is *OFF*, the combination of the rest of the attack parameters is irrelevant as the overall attack does not materialize. The attack in $S_{001}$ does not materialize despite having a state where RDP is ON because the other two parameters which actualize the backdoor are OFF. The "do not care "values are represented by the denotation $X$ to denote that, whether the value is *1* or *0* is actually insignificant as it bears no effect on the end result.

We further observed that with RDP on, only one of the attack vectors of backdoor implantation is required as evidenced by the XOR operation of state $S_{010}$ in Table 2. Furthermore, the presence of two implanted backdoors by the defined attack vectors in the presence of activated RDP adds no difference in that the attack will materialize as if only one backdoor was implanted. However, considering that this is a malware-free intrusion, an attacker might choose to activate both backdoors to increase the level of persistence in the event that one backdoor is detected. Now, having observed that only three states out of the eight actually represent the status of the system where the attack is successful, we finally deduce the overall attack chain for the malware-free intrusion using our defined attack vectors. The attack chain is depicted below in Figure 8.

The chain comprises four steps of which the first and

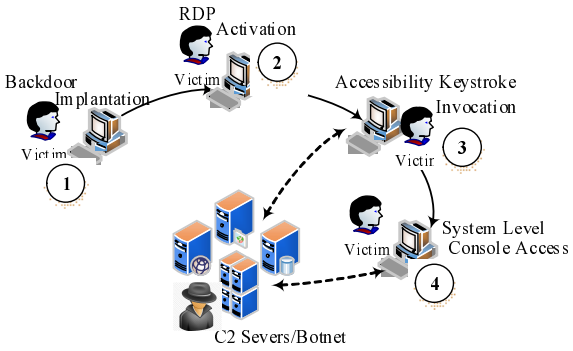


Figure 8: Malware-free intrusion attack chain

second step could come in any order. The attacker can be initiating the attack from Command and Control (C2) servers or botnets and since this backdoor can be persistent, the attacker can include the victim to their list of C2 servers or to their botnet for further compromise. The accessibility keystroke invocation over the network in the third step can only work when the first two steps are completed. Likewise, system level access over RDP-based remote access is only feasible after materialization of the first three steps. Skipping any of these steps thwarts the attack.

It is worth noting that NLA [10], a security feature introduced into later versions of the Windows operating system to thwart Denial of Service (DOS) attacks over an RDP session somewhat inadvertently helps mitigate backdoor attacks discussed herein. Activating NLA after switching on RDP requires that the connecting user avail their authentication credentials before a session is established. We did not include activation of NLA as an attack vector because RDP-based remote access functions without this feature. NLA, however, has its own implementation challenges as later elaborated in this paper.

# 4 Bayesian and Security Assessment Model

We engage the services of Bayesian network statistics to the attack vectors defined in our attack model and attack chain in the preceding sections. Attack vectors and paths have been employed in the study of vulnerabilities of various information systems [11, 19] and further to develop probabilistic metrics of enterprise networks [6, 21]. We likewise extend and employ this technique to our assessment model via the construction of a directed graph with specific nodes and corresponding edges. We apply the aforementioned attack vectors to the paths and the resultant is a directed graph of an infiltration Bayesian network shown in Figure 9 below.

Our infiltration Bayesian network is presented as a cascaded hierarchical graph of three levels where the topmost level denotes the entry point of the network. The set of

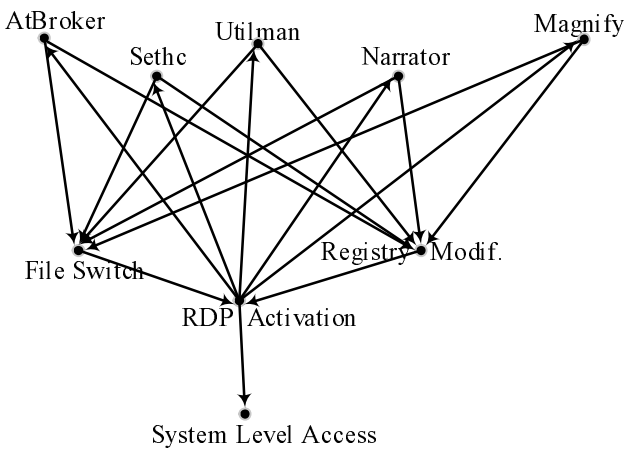


Figure 9: Infiltration Bayesian network

first entry points are attacks on the accessibility suite executable binaries which when switched (second level) with an executable binary (e.g. cmd.exe) capable of availing system level access or where such a file is set as the debugger (second level) for specified accessibility suite executable binary in the registry results into implantation of the malware-free backdoor. We differentiate five accessibility suite binaries namely Atbroker.exe, Sethc.exe, Utilman.exe, Narrator.exe and Magnify.exe which reside in the %systemroot%\system32 directory. The second entry point of the infiltration Bayesian network is RDP activation though the order of these entry points at this level is of no significance as observed in the previous section. The third and final level is system level access after successful traversal of the appropriate paths.

We thus denote the infiltration Bayesian network (BN) with the following parameters:

$$BN^{inf} = (G^{inf}, \Omega^{inf})$$

where $G^{inf}$ is a directed graph containing nodes representative of random variables and the links between the nodes denoting direct dependence relationship, and where $\Omega^{inf}$ denotes the set of quantitative parameters $\omega_i$ of the given network for $i = 1,...,n$. We represent the sequence of random variables of the given nodes as:

$$S_r^{inf} = \{B_i\}_{i=1}^{n}$$

where the random binary variable $B_i$ dictates if a pursued node $i \in \{1, ..., n\}$ has been accessed through the corresponding attack vector. We define the Bayesian probability $b_{ij}$ as the probability of accessing node $i$ given that node $j$ is accessed by the pursued attack vector:

$$b_{ij} = Pr(B_i|B_j)$$

Therefore, the link set $L^{inf}$, denoting a collection of nodes through which an attack vector can be pursued with positive probability is defined as:

$$L^{inf} = \{(i,j) : a_{ij} > 0,\ i,j \in \{1,...n\}, i \neq j\}$$

We further define the Bayesian probability of the network parameter $\omega_i$ as:

$$\omega_i = Pr(B_i|\Phi_i)$$

where the set of parent nodes $\Phi_i = \{B_j : b_{ij} > 0\}$ and the network parameter is a subset such that $\omega_i \in \Omega^{inf}$.

We now construct a table, Table 3, consisting different security control profiles which can help address the vulnerabilities revealed in our models. We consider four distinct systems with seven security controls partitioned progressively starting with a minimal set of security controls. Enhanced security controls are shaded green while average controls are shaded orange and the least controls not shaded at all.

Security controls in the first system configuration, System 1, are at their minimum. There are no additional controls integrated in the system and those controls that come by default with the setting are left unaltered. This is the least desirable state which would otherwise correspond to state $S_{011}$ of Figure 1 and subsequently Equation (5). Since the attack is a malware-free intrusion attack, there is no high expectation of detection by the IDS that be.

The security controls of the second system configuration, System 2, introduce two security features. These controls are classified as average because file integrity check in the %systemroot%\system32 directory is not complete. This implies the checking mechanism might not be able to detect a backdoor implanted by the unchecked accessibility executable binary. This might be the case of an updated system in which the update introduces a new the set of accessibility features not captured by the integrity check before the update. The registry modification detection will depend on the set of accessibility features present and will likewise in the same manner miss some registry modifications on the same pretext that partial file integrity check fails. This corresponds to state $S_{001}$ of Figure 1 and subsequently Equation (3) where it is possible to pursue either of the two attack vectors to completion but not both.

The set of security controls applied in System 3 include full file integrity check in the %systemroot%\system32 directory and full hash collision detection. This implies that any backdoor implanted by the attack vector of file switch will probably be detected. Furthermore, this security profile includes port obscurity which obscures the RDP port against brute-forcing attacks [24] on the default port 3389 or against the attacks initiated by automated RDP discovery scripts [3].

However, for a targeted attack, the attacker will have to go a step further to probe all ports on the system to overcome this security control. Nonetheless, this security profile lacks NLA which otherwise prevents establishment of an RDP session at pre-authentication. Likewise, it does not employ FDE with key preservation.

Table 3: System security control profiles

| System | File Integrity Check | Hash Collision Check | Reg. Modification Detection | NLA | Port Obscurity | FDE | Pre-Authentication Accessibility Features |
|---|---|---|---|---|---|---|---|
| *System 1* | Absent | Absent | Not Present | Left OFF | Default RDP | No | Default Settings |
| *System 2* | Partial | Absent | Not Present | Turned OFF | Default Port | No | Default Settings |
| *System 3* | Complete | Complete | Present | Unchecked | Obscured Port | No | Default Settings |
| *System 4* | Complete | Complete | Present | Turned ON | Obscured Port | Yes | All turned off |

To this effect it is possible to implant the backdoor but only via setting cmd.exe as the debugger for any of the chosen accessibility suite executable binaries. This, in the event that RDP is activated covertly, would correspond to state $S_{010}$ of Figure 1 denoted by Equation (4) implying the pursuance unto completion two attack vectors which actualize the attack.

The last security profile presents hardened security configurations which provide the highest level security counter measures against this malware-free intrusion backdoor. All binary executables in the %systemroot%\system32 directory are hashed unto completion to verify file integrity and a hash detection computed to determine any indicators of compromise in the event of a hash collision. This entails that the malware-free backdoor cannot be implanted via this attack vector. Furthermore, registry modification detection are carried out for both backdoor implantation of setting a debugger to an accessibility executable binary and also detection for covert activation of RDP through the registry as elaborated in Section 3. In addition to port obscurity, this security profile also enforces the usage of NLA implying that even in an extenuating scenario where a backdoor were to somewhat slip through slip the aforementioned security controls of this profile, interactive console system level access would not be attained as this would require authentication first before session establishment as per NLA requirement. Such a security profile is reflected by the most secure state $S_{000}$ of Figure 1 where Equation (2) represents such as state.

Since the outcome of the attack depends on the conditional probability that the present attack vector can only be pursued unto completion, if the other related attack vector has successfully been actualized, we can employ conditional probabilities of Bayesian inference to evaluate our assessment model. The conditional probabilities are denoted by directed edges in the Bayesian network in Figure 9. So we apply the security controls in Table 3 to these edges in the formulation of our assessment model.

Since Table 3 presents security controls capable of detecting the backdoor, we define a binary random variable $\Psi_i^{inf}$ for $i = 1, ..., n$ to denote detection of infiltration of the corresponding *i-th* node. We therefore compute the probability of undetected infiltration as:

$$Pr(B) = 1 - \prod_{i=1}^{n}[1 - Pr(B|B_i) \cdot Pr(B_i) \cdot [1 - Pr(\Psi_i^{inf})]]$$

and we evaluate the probability of access of the *i-th* node as:

$$Pr(B_i) = 1 - \prod_{j=1}^{n}[1 - Pr(B_i|B_j) \cdot Pr(B_j)]$$

for $i$ = *1,...,n*, we have:

$$Pr(B_i) = 1 - \prod_{j=1}^{n}[1 - b_{ij} \cdot Pr(B_j)]$$

considering that $b_{ij} = Pr(B_i|B_j)$. If the probability of not detecting an attack via a given node is defined as $Pr(B|absence\,of\,detection)$, that is to mean $Pr(\Psi_i^{inf}) = 0$, then we estimate the probability that the implemented security controls can detect the attack as:

$$Pr(\Psi^{inf}) = \frac{Pr(B|absence\,of\,detection) - Pr(B)}{Pr(B)}$$

We assumed the Markov property [7] in our formulations that access to the *i-th* node depends only on its parents and not on the history of the subsequent nodes thereof.

Application of the model requires computation of conditional probabilities for all edges in our Bayesian network. We compute the probability scores of our network using Conditional Probability Tables (CPT), representative of the strength on an influence, as shown in Table 4 below. The CPT likewise represents the probability distribution of possible states of a node of which the preconditions are based on its parents' states.

Table 4: CPT for $Node_i$

| $\Phi_1$ | $\Phi_2$ | $\Phi_3$ | $\Phi_4$ | *Node i* |
|---|---|---|---|---|
| *T/F* | *T/F* | *T/F* | *T/F* | $Pr(B_i\|B_j)$ |
| *Otherwise* | | | | $1 - Pr(B_i\|B_j)$ |

Since our malware-free intrusion attack structure is not directly based on software vulnerability exploit, we esti-

mate the probability of success directly in the corresponding knowledge base. Therefore, to derive CPT parameters, we employ the use of discrete levels homologous to those reflected in the Common Vulnerability Scoring System (CVSS) metrics [9, 23]. In view of the aforementioned, we thus assign the following values to the conditional probability: very high – assigned a value of 1, *high* – assigned a value of 0.9, *medium* – assigned a value of 0.5, *low* – assigned a value of 0.1 and *very low* – assigned a value of 0.01. The Table 5 below shows some selected probability scores.

Table 5: Selected probability scores

| Attack | System 1 | System 2 | System 3 | System 4 |
|---|---|---|---|---|
| Initial access | 1 | 1 | 1 | 1 |
| Reg. modify | 1 | 0.9 | 0.9 | 0.01 |
| Sethc switch | 0.9 | 0.5 | 0.1 | 0.1 |
| RDP ON | 0.9 | 0.5 | 0.5 | 0.01 |

We assume that access to the entry node is used as given implying that for initial system access: $Pr(B_1) = 1$. This is so because the spectrum of this access is so wide that it most likely encompasses attack vectors that might employ malware like exploit kits which is in conflict with the approach considered in this paper.

Having evaluated the probability scores, we now present security profile assessment results of the security profiles from Table 3 in the following Table 6 below. The probabilities are presented for System Level Access (SLA) with different detection parameters.

Table 6: Security profile assessment results

| Security Profile | SLA with detection | SLA minus detection | SLA detection Prob. |
|---|---|---|---|
| System 1 | 0.88 | 0.99 | 0.1 |
| System 2 | 0.85 | 0.97 | 0.12 |
| System 3 | 0.26 | 0.49 | 0.47 |
| System 4 | 0.002 | 0.005 | 0.5 |

The highest level of infiltration is echoed in system profile 1 and 2 with probability averaging 98%. This is explained by the lack of robust security controls as depicted in Table 3. On the other hand, if the attack is detected before actual SLA, the probability is reduced as evidenced in the third profile with relatively better security controls as opposed to the first and second profile. The fourth profile, with the most hardened security controls, has the highest detection probability and the lowest infiltration cases. This profile extensively employs rigorous integrity checks both in the SystemRoot and Registry whilst counter-checking any indicators of compromise via hash collisions. These security controls mitigate the known attack vectors through which the accessibility backdoor is implanted and subsequently accessed with SLA.

# 5 Conclusion

We have demonstrated how a security assessment model based on Boolean Logic for security state formulation and Bayesian inference for probability evaluation can be used for the assessment of the security environment of a specified scenario of malware-free intrusion. The number of attack vectors as variables of Boolean functions have a direct influence on the attack paths and the infiltration thereof generated through BN. The assessment model gives insight into the significance of the various security controls meant to counter attacks via malware-free intrusions. Since the characteristics of a malware-free intrusion differs from those IDSs are accustomed to, countering attacks via these attack vectors calls for tailor made solutions which might otherwise not come with the common security products.

Compared to other works on network security assessment based on Bayesian models [1, 20], our work introduces the use of Boolean state machines for precise representation of security states of affected systems. Furthermore, our work encompasses malware-free intrusions which have not been explicitly inferred in Bayesian networks and state machines before.

Inasmuch as the collaboration of Boolean Logic and Bayesian inference sheds more light on the constituents of a malware-free attack, the approach also faces challenges in that there are some components of the Bayesian network which cast a considerable level uncertainty difficult enough to be captured by Boolean Logic reasoning. So regardless of the robustness of the implemented security measures against attacks in this respect, it is not as straightforward to postulate and extrapolate for certain that the attack will not materialize as a binary response.

# References

[1] F. X. Aguessy, O. Bettan, G. Blanc, V. Conan, and H. Debar, "Hybrid risk assessment model based on bayesian networks," in *International Workshop on Security*, pp. 21–40, Springer, 2016.

[2] I. Arce and G. Richarte, "State of the art security from an attacker's viewpoint," in *PacSec Conference*, Tokyo, Japan, 2003.

[3] E. Beqiri and D. Campus, *Information and Communication Technology Security Issues*, University of East London, 2010.

[4] M. K. Daly, "Advanced persistent threat," *Usenix*, vol. 4, no. 4, pp. 2013–2016, 2009.

[5] G. Dong, H. Hao, R. Du, and L. Tian, "Attacking mode based on shell structure of complex networks,"

*International Journal of Nonlinear Science*, vol. 20, no. 3, pp. 148–153, 2015.

[6] M. Frigault, L. Wang, A. Singhal, and S. Jajodia, "Measuring network security using dynamic bayesian network," in *Proceedings of the 4th ACM Workshop on Quality of Protection*, pp. 23–30, 2008.

[7] M. Frydenberg, "The chain graph markov property," *Scandinavian Journal of Statistics*, vol. 17, no. 4, pp. 333–353, 1990.

[8] A. Futoransky, L. Notarfrancesco, G. Richarte, and C. Sarraute, "Building computer network attacks," *arXiv preprint arXiv:1006.1916*, 2010.

[9] L. Glanz, S. Schmidt, S. Wollny, and B. Hermann, "A vulnerability's lifetime: enhancing version information in cve databases," in *Proceedings of the 15th ACM International Conference on Knowledge Technologies and Data-driven Business*, pp. 28, 2015.

[10] K. H. Ho, *Remote Desktop Services in Windows 2008 R2*, 2008. (`http://sharepointgeorge.com/2009/remote-desktop-services-windows-2008-r2-part-1/`)

[11] M. Khosravi-Farmad, R. Rezaee, A. Harati, and A. G. Bafghi, "Network security risk mitigation using bayesian decision networks," in *4th IEEE International eConference on Computer and Knowledge Engineering (ICCKE'14)*, pp. 267–272, 2014.

[12] M. Z. Mas' ud, S. Sahib, M. F. Abdollah, S. R. Selamat, and C. Y. Huoy, "A comparative study on feature selection method for n-gram mobile malware detection," *International Journal of Network Security*, vol. 19, no. 5, pp. 727–733, 2017.

[13] S. Mauw and M. Oostdijk, "Foundations of attack trees," in *International Conference on Information Security and Cryptology (ICISC'05)*, pp. 186–198, 2005.

[14] J. Moon, D. Lee, J. Jung, and D. Won, "Improvement of efficient and secure smart card based password authentication scheme.," *International Journal of Network Security*, vol. 19, no. 6, pp. 1053–1061, 2017.

[15] L. Ray and H. Felch, "Detecting advanced persistent threats in oracle databases: Methods and techniques," in *Strategic Information Systems and Technologies in Modern Organizations*, pp. 71–89, 2017.

[16] K. Rieck, P. Trinius, C. Willems, and T. Holz, "Automatic analysis of malware behavior using machine learning," *Journal of Computer Security*, vol. 19, no. 4, pp. 639–668, 2011.

[17] C. Sarraute, G. Richarte, and J. Lucángeli Obes, "An algorithm to find optimal attack paths in nondeterministic scenarios," in *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence*, pp. 71–80, 2011.

[18] S. Saxena, "Demystifying malware traffic," *SANS Institute InfoSec*, 2016.

[19] V. Shandilya, C. B. Simmons, and S. Shiva, "Use of attack graphs in security systems," *Journal of Computer Networks and Communications*, vol. 2014, 2014.

[20] J. Shin, H. Son, G. Heo, *et al.*, "Development of a cyber security risk model using bayesian networks," *Reliability Engineering & System Safety*, vol. 134, pp. 208–217, 2015.

[21] A. Singhal and X. Ou, *Security Risk Analysis of Enterprise Networks Using Probabilistic Attack Graphs*, US Department of Commerce, National Institute of Standards and Technology, 2011.

[22] X. Wang, K. Zheng, X. Niu, B. Wu, and C. Wu, "Detection of command and control in advanced persistent threat based on independent access," in *IEEE International Conference on Communications (ICC'16)*, pp. 1–6, 2016.

[23] A. Younis, Y. K. Malaiya, and I. Ray, "Assessing vulnerability exploitability risk using software properties," *Software Quality Journal*, vol. 24, no. 1, p. 159, 2016.

[24] A. Zimba, "Malware-free intrusion: A novel approach to ransomware infection vectors," *International Journal of Computer Science and Information Security*, vol. 15, no. 2, pp. 317, 2017.

## Biography

**Aaron Zimba** received his Master and Bachelor of Science degree from the St Petersburg Electrotechnical University in St Petersburg in 2009 and 2007 respectively. He is currently a PhD student at the University of Science and Technology Beijing in the Department of Computer Science and Technology. He is also a member of the IEEE. His main research interests include Network Security Models, Network and Information Security, Cryptovirology and Cloud Computing Security .

**Hongsong Chen** received his Ph.D degree in Department of Computer Science from Harbin Institute of Technology, China, in 2006. He was a visiting scholar in Purdue University from 2013-2014. He is currently an associate professor in Department of Computer Science, University of Science and Technology Beijing, China. His current research interests include wireless network security, attack and detection models, and cloud computing security.

**Zhaoshun Wang** is a Professor and the Associate Head of the Department of Computer Science and Technology at the University of Science and Technology Beijing. He graduated from Department of Mathematics at Beijing Normal University in 1993. He received his PhD from Beijing University of Science and Technology in 2002. He completed postdoctoral research work at the Graduate School of the Chinese Academy of Sciences in 2006. He holds patents and has many awards to his name. His main research areas include Information Security, Computer Architecture and Software Engineering.