

Cyber Attack Classification using Game Theoretic Weighted Metrics Approach

¹Bimal Kumar Mishra and ²Hemraj Saini

¹Department of Applied Mathematics, Birla Institute of Technology, Mesra, Ranchi, 835 215, India

²Department of Information Technology, Orissa Engineering College, Bhubaneswar, 752050, Orissa, India

Abstract: We propose a classification approach of cyber attack which uses characteristics metrics and game theoretic approach to classify the attacks to their closest category. The standard weights of the metrics are used as the base line to classify the cyber attacks in the proper category. The approach is simple and extendible; as new characters of the newly identified attacks can be added to the attack characteristic metrics and the corresponding unique weight to the character are assigned by the proposed formula. Besides this, the proposed approach clearly represents the cause effect relationship for all possible attacks which helps us to find the appropriate solution to restrict them in the Internet. A case study is also provided in the paper to classify Nimda worm, Slammer worm, W32/HLLP.Philips.jn Trojan, Stealth virus and Boot sector virus.

Key words: Cyber attacks . cyber attack classification . characteristic metrics . malicious objects . game theoretic approach

INTRODUCTION

Electronic mail and use of secondary devices are the major sources for the transmission of malicious objects in computer network these days. Malicious object is a code that infects computer systems. There are different kinds of malicious objects such as: Worm, Virus, Trojan horse etc., which differ according to the way they attack computer systems and the malicious actions they perform. There are many flat terms [1] for the cyber attack classification such as virus, worm, Trojan horse, Spam, DoS, Phishing etc. which is defined in the Appendix. Some terms are having common characteristics, i.e., viruses and Trojans need external event to be activated. So if an attack is having common properties of more than one term then it can be classified in all of these terms. For example, Code Red can be classified as worm and Trojan according to Trend-Micro, a well known security organization. Another problem is to make new attack more sophisticated after merging features of two or more categories. For example, Nimda is having features of both virus and worms according to F-secure [2]. Except this, if an attack steals the information, it can be classified as phishing attacks and Trojans, as both of them steals the information but by different ways. The first theory of classification is based on these flat lists. If we adapt these flat lists of attack types, then there is no relationship between the cause of an attack and its effects, that is, it doesn't represent the cause effect

relationship of cyber attacks [3]. In the second theory of classification based on list of categories it somewhat represents the cause effect relationship of a cyber attack but it is too poor [4]. The third theory is based on vulnerability [5]. Vulnerability based classification is also not able to represent the cause effects relationship of cyber attacks. Another possibility to classify the cyber attacks is metrics based classification. As the cyber attacks are made by programs, metrics should be the best way to represent the characteristics of a cyber attack. Other benefit to use the metrics is that it can represent the cause effects relationship of a cyber attack in a better way. In today's world of information, cyber attacks are more sophisticated and pose more threat to the valuable information stored on a machine or in transit. False positive rates and wrong detection of the attacks are due to lack of undefined boundaries. Hence it is important to have a sound understanding of cyber attacks to provide better defense against them. In this paper, we propose a mathematical approach to classify the cyber attacks by using the concept of game theory and characteristic metrics of cyber attacks.

RELATED WORK AND OUR APPROACH

Two categories of the classification approach are available in the literature. First, vulnerability based cyber attack classification approach [6-8].

Vulnerabilities are basically the loop holes in the software which are identified by the attackers and exploited. There are various types of vulnerabilities and on the basis of exploitation of a specific type of vulnerability, cyber attacks are classified. Michael Gegick, M., & Williams, L. [9] worked in this area. They gave a methodology for early identification of system vulnerabilities: Security Analysis for Existing Threats (SAFE-T). With SAFE-T, software engineers match attack patterns for system designs to identify potential vulnerabilities. SAFE-T is a tool for security to start in the design phase of the software process and provides taxonomy of attack patterns that can describe common security problems. Security awareness plagues software engineers today [10-12] and so they attempt to illuminate vulnerabilities with a software engineering approach. In this approach different types of attacks may exploit a same vulnerability and can categorize in more than one category, which is inconsistent. The approach is not so much exhaustive because it does not represent the causes and effects relationship so effectively. Second, system behavior based classification approaches are proposed in [13, 14]. The approach based on system behavior [15-17] gives the stress to use the impacts of attack to classify it. This approach is better suited to represent the cause effect relationship but it does not show the well defined distinction between the causes and effects. Cohen gave a list of flat cyber attack list [18-21] to represent the relationship among threat, attack and defense. Different terms used are virus, worms, Trojans, etc. which are not sufficient to represent the cause effect relationship as they are static in behavior. Therefore, metrics are used to categorize the cyber attacks to the best suited category in place of term. Our approach is based on weighted elements of the metrics. In this, we dynamically assign the proper weights to each element of the metrics based on its behavior and compute the final average total weight of a cyber attack. This final average total is compared with the standard weight of the type of cyber attack category. On the basis of the acceptable standard deviation we classify the cyber attack to the best suited one.

CHARACTERISTIC METRICS OF CYBER ATTACKS

To understand the cause effect relationship, let us to take an example of any protocol which follows request-response pattern e.g. ICMP or ARP. In such type of protocols, one host (say host-1) sends a request to the other host (say host-2) and gets the response for next activity. As the host-1 sends the request, due to spoofing host-2 gets the wrong information and sends

the response to the wrong destination (say host-3). But the host-3 will not send any response to it as it did not send any request for the received response; therefore, the host-2 sends the response again and again, causing congestion of traffic. See the following flow of information:

- Step-1: Different target machines are running a Response Generation Service (RGS).
- Step-2: Attackers send a packet to the RGS port of victim x.
- Step-3: Due to spoofing it returns the address of RGS port of victim y.
- Step-4: Victim x parse the packet and send it to its RGS server.
- Step-5: Victim x responds to the forged address (to RGS of port y as the destination).
- Step-6: Victim y receives the packet and responds to it.
- Step-7: This process will continue till the forging is not detected.

Take the step-3; Spoofing is the cause to return the wrong machines RGS port address. In Step-5 and Step-6, sending the response to the forged address is the cause of the unnecessary traffic in the network and increase of the CPU and memory usage for both the machines. Finally, we can say that the cause is responding to forged address and effects are increment in the network traffic congestion, CPU usage and memory usage for both the machines. Therefore, to create the characteristic metrics for cyber attacks we need to include the possible causes and effects. An attack must be having characteristics such as attack objective, way of attack propagation and way of initiation of attack such as exploitation of vulnerability or using its own method, misusing existing asset, effects of state of used asset, effects on performance of asset etc. Table 1 shows the attack characteristics with their possibilities.

We are showing a partial weighted cyber attack metrics in Fig. 1. It is having eight stages. We assign the unique weight to every component of each stage. The formula shown in equation-(1) is used to assign the weights to each component of each stage of the characteristics metrics of attack.

$$W_{S,i} = \frac{\left(i + \sum_{j=1}^{j=S-1} E_j \right)}{\sum_{j=1}^{j=n} E_j} \quad (1)$$

where, $W_{S,i}$ is weight for i^{th} component at the S^{th} stage, E_j is the number of elements in the j^{th} stage and n is the total number of stage in the characteristic metrics.

Table 1: Possible characteristics of an attack with possible alternates

Attack characteristic	Possibilities of characteristics
Attack objectives	Information cracking Terrorism Regular crime Any other
Way of attack propagation	By human being Independent or autonomous
For initiation vulnerability exploited	Design vulnerabilities Configuration vulnerabilities Implementation vulnerabilities
For initiation method used	Unauthorized access Unauthenticated access
Asset misused	BW misused Memory space misused CPU time misused Network missed Data misused System misused User misused
Effects of state of used asset	Availability of asset Integrity of asset Confidentiality of asset
Effects on performance of asset	Regarding timeliness Regarding precision Regarding accuracy

Stage-1:	Attack	= $W_{1,1} = (1+0)/40 = 0.025$
Stage-2:	Human	= $W_{2,1} = (1+1)/40 = 0.050$
	Autonomous	= $W_{2,2} = (2+1)/40 = 0.075$
Stage-3:	Local	= $W_{3,1} = (1+3)/40 = 0.100$
	Remote	= $W_{3,2} = (2+3)/40 = 0.125$
Stage-4:	Probe	= $W_{4,1} = (1+5)/40 = 0.150$
	Scanning	= $W_{4,2} = (2+5)/40 = 0.175$
	Flooding (Single)	= $W_{4,3} = (3+5)/40 = 0.200$
	Flooding (Multiple)	= $W_{4,4} = (4+5)/40 = 0.225$
	Authenticate	= $W_{4,5} = (5+5)/40 = 0.250$
	Bypass	= $W_{4,6} = (6+5)/40 = 0.275$
	Spoof	= $W_{4,7} = (7+5)/40 = 0.300$
	Read	= $W_{4,8} = (8+5)/40 = 0.325$
	Copy	= $W_{4,9} = (9+5)/40 = 0.350$
	Termination	= $W_{4,10} = (10+5)/40 = 0.375$
	Create Process	= $W_{4,11} = (11+5)/40 = 0.400$
	Execute	= $W_{4,12} = (12+5)/40 = 0.425$
	Steal	= $W_{4,13} = (13+5)/40 = 0.450$
	Modify	= $W_{4,14} = (14+5)/40 = 0.475$
	Delete	= $W_{4,15} = (15+5)/40 = 0.500$
	Misdirect	= $W_{4,16} = (16+5)/40 = 0.525$
	Eavesdrop	= $W_{4,17} = (17+5)/40 = 0.550$
	None	= $W_{4,18} = (18+5)/40 = 0.575$
Stage-5:	Configuration	= $W_{5,1} = (1+23)/40 = 0.600$
	Design	= $W_{5,2} = (2+23)/40 = 0.625$
	Implementation	= $W_{5,3} = (3+23)/40 = 0.650$
	Others	= $W_{5,4} = (4+23)/40 = 0.675$
Stage-6:	Network	= $W_{6,1} = (1+27)/40 = 0.700$
	Data	= $W_{6,2} = (2+27)/40 = 0.725$
	User	= $W_{6,3} = (3+27)/40 = 0.750$
	System	= $W_{6,4} = (4+27)/40 = 0.775$
	None	= $W_{6,5} = (5+27)/40 = 0.800$
Stage-7:	Confidentiality	= $W_{7,1} = (1+32)/40 = 0.825$
	Integrity	= $W_{7,2} = (2+32)/40 = 0.850$
	Availability	= $W_{7,3} = (3+32)/40 = 0.875$
	None	= $W_{7,4} = (4+32)/40 = 0.900$
Stage-8:	Timeliness	= $W_{8,1} = (1+36)/40 = 0.925$
	Accuracy	= $W_{8,2} = (2+36)/40 = 0.950$
	Precision	= $W_{8,3} = (3+36)/40 = 0.975$
	None	= $W_{8,4} = (4+36)/40 = 1.000$

INTRODUCTION TO GAME THEORETIC WEIGHTED APPROACH OF CYBER ATTACK

Classification: Game theory is one of the oldest areas of endeavor in artificial intelligence. In 1950, almost as soon as computers became programmable, the first chess program was developed by Claude Shannon and by Alan Turing. Game playing is having enough and good ideas to find out the best decision from different possibilities, as required in our case to classify the various attacks. We choose game theoretic approach due its advantages such as

- It is a quantitative approach
- It is inherently decentralized
- It is having possibilities to implement online learning mechanisms
- Parameter can be updated iteratively

A game theoretic approach uses following components to solve the problems:

- Initial state: It is the starting state of the board positions and identifies whose turn it is.
- Set of operations: This defines legal moves that a player can make.

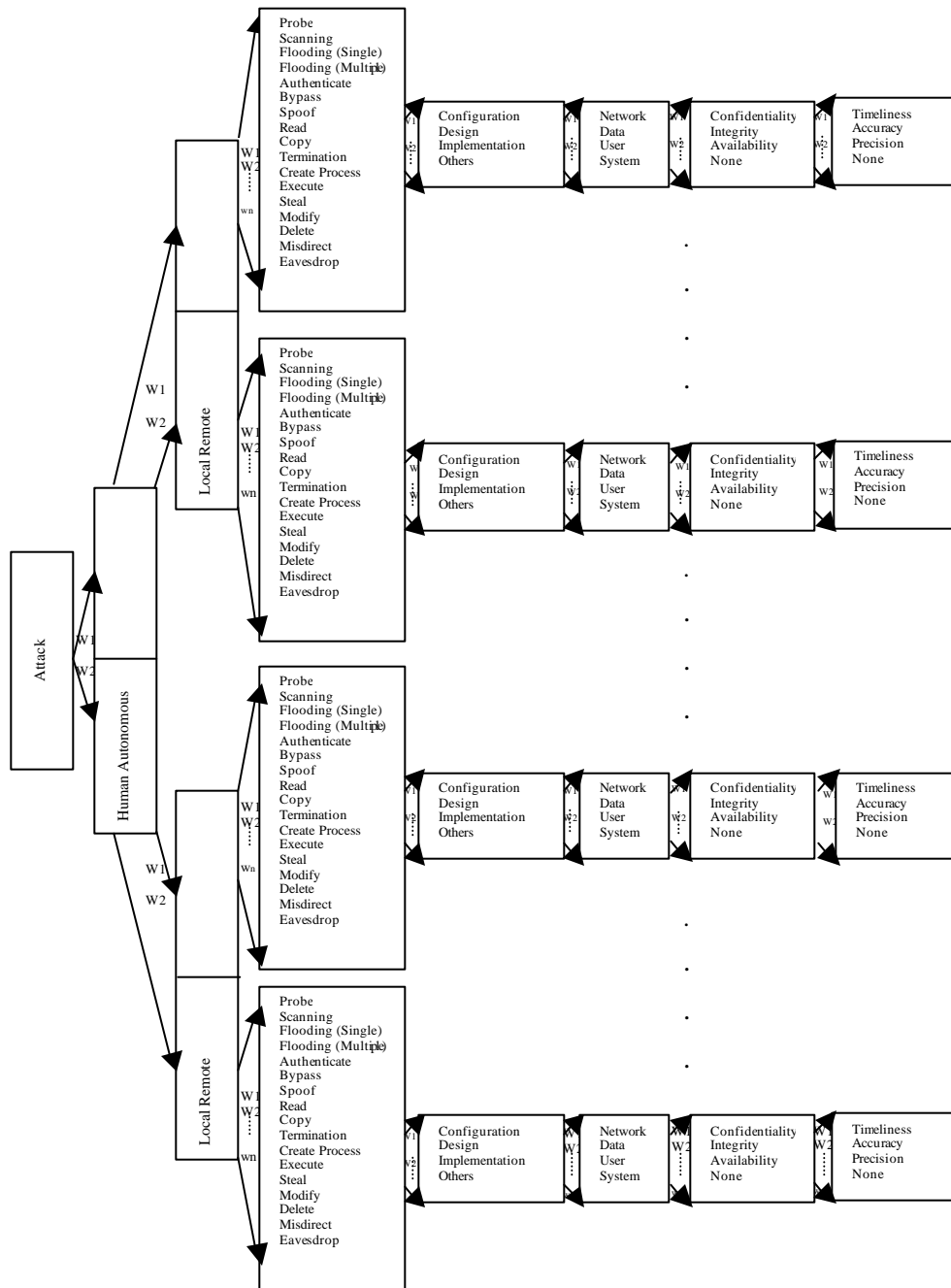
Fig. 1: Partial weighted metrics of cyber attack with 8-stages having weights of all its members

- Terminal test: This determines when the game is over.
- Utility function: It gives a numeric value for the outcome of a game.

In our case these requirements are defined as under:

- Initial state: The behavioral understand of attack is to be considered as the initial condition to classify it.
- Set of operations: Find out the most appropriate component from each stage of characteristics metrics by using the maximum search process of game playing. On the basis of the analysis of the attack pattern we can identify the causes and their concern effects and match them from the available characteristics metrics of attack.
- Terminating test: Visiting all stages of characteristics metrics of attack.

Utility function: In our case it is simple addition of respective weights of every selected component from each stage by using maximum



Note: For every stage having k number of elements $w_1 + w_2 + \dots + w_k = 1$

Here w_n represents the possibility factor of n^{th} element

Fig. 2: Partial metrics representing cause effects for cyber attacks

search process of game theory i.e. $\sum_{s=1}^k W_{s,j}$, where, s is stage having number from 1 to k and j is the position of selected component from the stage. After considering all possible causes and effects of cyber attack and game theoretic concept, we are giving the partial metrics of cyber attack classification as shown in Fig. 2.

To compute the weight for standard malicious objects, we go through their behavior and find out the concern component from each stage of the characteristics metrics. Respective weights are then added to find the actual weight. Let us take the example of Boot Sector Virus.

Boot sector viruses infect the boot sector (or master boot record) on a computer system. They first move or

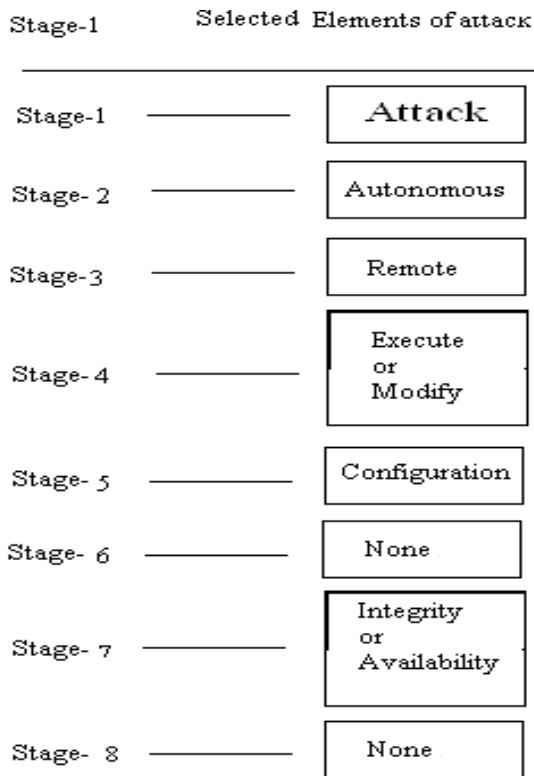


Fig. 3: Elements from the metrics for boot sector virus attack classification

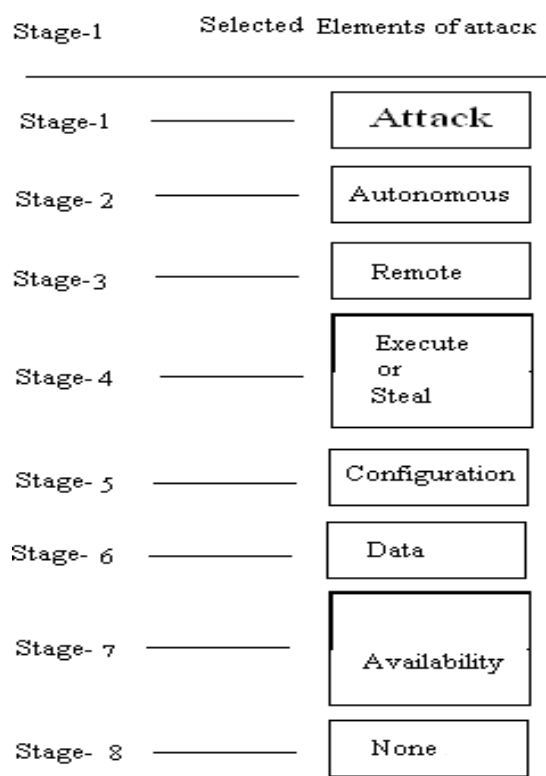


Fig. 4: Elements from the metrics for stealth virus attack classification

overwrite the original boot code, replicating it with infected boot code. They move the original boot sector information to another sector on the disk, marking that sector as a bad spot on the disk so that it can not be used in the future. Boot sector viruses can be very difficult to detect since the boot sector is the first thing loaded when a computer starts. In effect, the virus takes full control of the infected computer.

As an analysis result of the above behavior, the most appropriate elements of each stage are shown in Fig. 3.

The standard weight for the Boot Sector Virus from our attack characteristic metrics can be computed as follows:

$$\begin{aligned}
 &W_{1,1} + W_{2,2} + W_{3,2} + (W_{4,12} \text{ or } W_{4,14}) \\
 &+ W_{5,1} + W_{6,4} + (W_{7,2} \text{ or } W_{7,3}) + W_{8,4} \\
 &= 0.025 + 0.075 + 0.125 + (0.425 \text{ or } 0.475) \\
 &+ 0.600 + 0.775 + (0.850 \text{ or } 0.875) + 1 \\
 &= \{3.875, 3.900, 3.925, 3.950\} \quad (2)
 \end{aligned}$$

We take another case of *Stealth viruses*.

Stealth virus attempts to hide its replication from operating system and anti-virus software. It stays in memory to use the operating system calls. Once it executes, it performs different changes such as-change

file size, revert directory structures and/or other operating system aspects. Since the virus is memory resident, there will be less memory available to users.

On the basis of the above definition we can find out the most appropriate elements of each stage as show in Fig. 4.

The standard weight for the Stealth Virus from our attack characteristic metrics can be computed as follows:

$$\begin{aligned}
 &W_{1,1} + W_{2,2} + W_{3,2} + (W_{4,12} + W_{4,13}) \\
 &+ W_{5,1} + W_{6,2} + W_{7,3} + W_{8,4} \\
 &= 0.025 + 0.075 + 0.125 + (0.425 + 0.450) \\
 &+ 0.600 + 0.725 + 0.875 + 1 = \{3.850, 3.875\} \quad (3)
 \end{aligned}$$

In the above two computed sets of values for boot sector virus and stealth virus, some values are common because of some common behavior of two different types of viruses i.e. execution of file and replication.

CASE STUDY: W32/HLLP.PHILIS.JA, SLAMMER AND NIMDA WITH EMPIRICAL RESULTS OF THE APPROACH

W32/HLLP.Philis.jn and it's weight: Macafee, a well known security organization describes the facts about

W32/HLLP.Philis.jn in its database. Upon execution, it copies itself in %WinDir%\Uninstall\ folder as rundll32.exe and adds a load registry entry to activate itself on reboot. It also creates the following registry entries:

- HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DownloadManager
- HKEY_LOCAL_MACHINE\SOFTWARE\Soft\DownloadWWW\auto: "1"

It drops a file named RichDll.dll (detected as W32/HLLP.Philis.dll) in %WinDir%. It then injects this dll in processes Explorer.exe and IExplore.exe. This.dll is responsible for opening a backdoor and also downloading other password stealing Trojans such as PWS-Lineage from the following locations:

- [REMOVED]vcyin.com

W32/HLLP.Philis.jn searches for executable files and appends its viral code to target files. It adds its 57KB code in front of the original file, so whenever that file is executed the virus is also executed.

The virus creates files with the name "_desktop.ini" in every folder that it visits while looking for executable files to infect. This is created as a hidden system file and contains the date on which virus was executed to visit the folder in which the file resides. The date is shown in yyyy/mm/dd format.

The virus tries to spread via existing network shares. It searches for all active machines within the subnet. When it finds an active machine it sends an ICMP ping request and waits for a response. After getting the ping response it tries to access the ADMIN\$, IPC\$ and any other shares that might exist on the machine.

If the virus is able to access a shared resource, it first copies "_desktop.ini" to the root of the share to mark the share as visited and then infects executables present in the share. While infecting executables via a network share the virus does not limit itself to infecting specific file names as mentioned above. In the case of a shared printer, the viruses' infection routine effectively creates printer job to print the date as contained in "_desktop.ini" file that the virus tries to copy.

Indications of Infection:

- Presence of %WinDir%\RichDll.dll
- Presence of registry entries as described
- Presence of hidden system files named _desktop.ini in many folders.
- Increase in size of EXE files

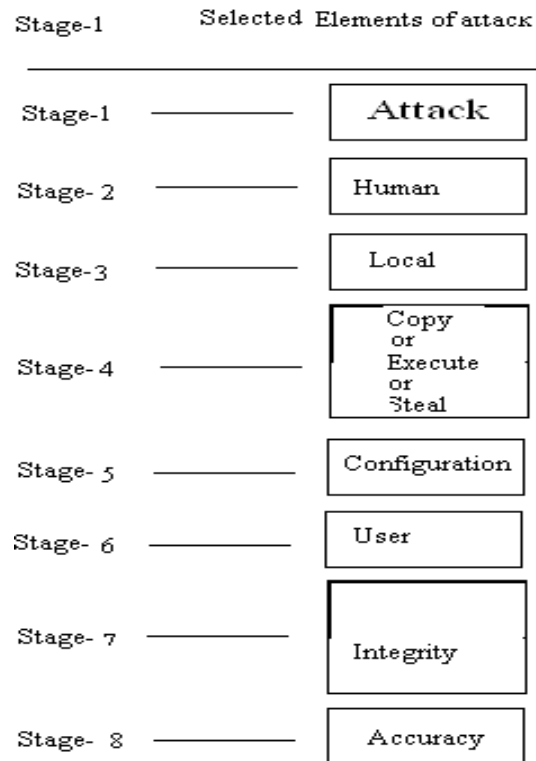


Fig. 5: Elements from the metrics for W32/HLLP.Philis.jn attack classification

Method of Infection: W32/HLLP.Philis.jn is a file infecting virus. Infection starts with manual execution of the binary. For spreading, the virus also relies on improperly configured/protected (open) shared drives.

On the basis of the above definition we can find out the most appropriate elements of each stage as show in Fig. 5.

The relevant weights of appropriate components from the partial metrics of cyber attack classification are used to find the standard weight of attack as follows.

$$\begin{aligned}
 &W_{1,1} + W_{2,1} + W_{3,1} + (W_{4,9} \text{ or } W_{4,12} \text{ or } W_{4,14}) \\
 &+ W_{5,1} + W_{6,3} + W_{7,2} + W_{8,2} = 0.025 + 0.050 \\
 &+ 0.100 + (0.350 \text{ or } 0.425 \text{ or } 0.475) + 0.600 \\
 &+ 0.750 + 0.850 + 0.950 = \{3.675, 3.750, 3.800\} \quad (4)
 \end{aligned}$$

Slammer worm and it's weight: The facts about the Slammer worm are as under:

Get inside: Slammer masquerades as a single UDP packet, which would normally be a harmless request to find a specific database service. The first byte in the string-04-tells SQL Server that the data following it is the name of the online database being sought. Microsoft's tech specs say that this name be at most

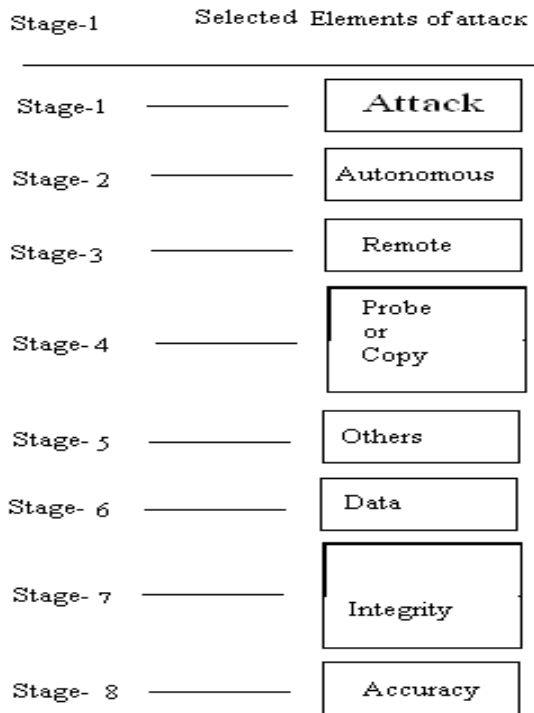


Fig. 6: Elements from the partial metrics for Slammer worm attack classification

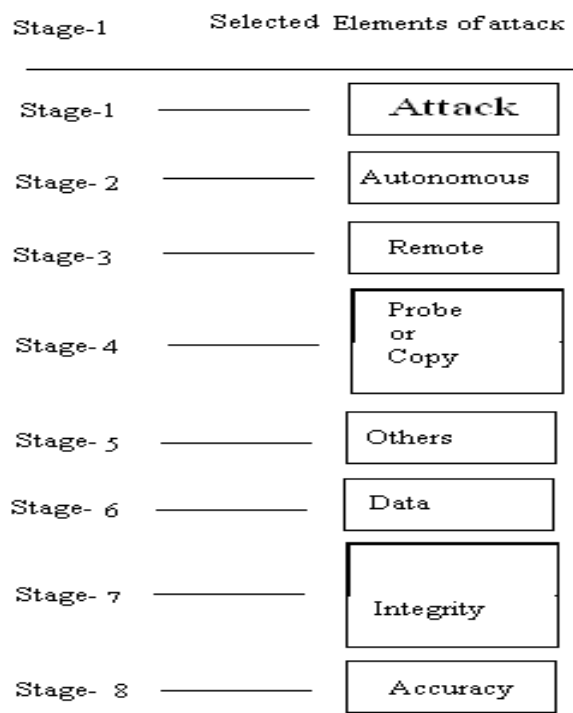


Fig. 7: Elements from the partial metrics for Nimda worm attack classification

16 bytes long and end in a telltale 00. But in the Slammer packet, the bytes run on, so there is no 00 among them. Therefore, SQL software pastes the whole thing into memory.

Reprogram the machine: The first thing the computer does after opening Slammer’s too long UDP “request” is that it overwrite its own stack with new instructions that Slammer has disguised as a routine query. The computer reprograms itself without realizing it.

Choose victims at random: Slammer generates a random IP address, targeting another computer that could be anywhere on the Internet. It looks up the number of milliseconds that have elapsed on the CPU’s system clock since it was booted and interprets the number as an IP address.

Replicate: Once the packet is created and addressed, it gets stuffed. Slammer points to its own code as the data to send. The infected computer writes out a new copy of the worm and licks the UDP stamp.

Repeat: After sending off the first packet, Slammer loops around immediately to send another to a different computer. It doesn’t waste a single millisecond. Instead of making another call to the system clock to get the time, it just shuffles the bits of the IP address already in

memory to create a new one. On the basis of the above behavior we find out the most appropriate elements of each stage as shown in Fig. 6.

On the basis of above facts of Slammer worm we find out the Slammer attack standard weight as follows:

$$\begin{aligned}
 &W_{1,1} + W_{2,2} + W_{3,2} + (W_{4,1} \text{ or } W_{4,9}) + W_{5,4} \\
 &+ W_{6,2} + W_{7,2} + W_{8,2} = 0.025 + 0.075 + 0.125 \\
 &+ (0.150 \text{ or } 0.350) + 0.675 + 0.725 + 0.850 \\
 &+ 0.950 = \{3.575, 3.775\}
 \end{aligned}
 \tag{5}$$

Nimda worm and it’s weight: Nimda worm is having the following characteristics:

The W32.Nimda virus is an Internet, fileshare and mass mailing worm. It spreads by exploiting unpatched Microsoft IIS Web servers and through mass-mailed e-mail attachments named "readme.exe". If successful, it appears to deface the Web pages on the IIS Web site and install a backdoor. Initial reports indicate that the worm might also spread via network shares. This page will be updated as more information becomes available.

Infected client machines attempt to send copies of the Nimda worm via email to all addresses found in the Windows address book. Likewise, the client machines begin scanning for vulnerable IIS servers. Nimda looks for backdoors left by previous IIS worms: Code Red II and sadmind/IIS worm [CA-2001-11]. It also attempts to exploit the IIS Directory Traversal vulnerability. The

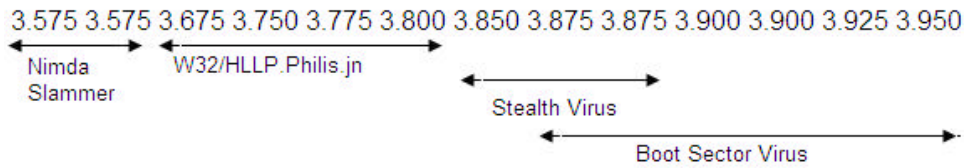


Fig. 8: Classification of different attacks on the basis of computed set of values by our approach

infected client machine transfers a copy of the Nimda code to any server that it scans and finds to be vulnerable. Once running on the server machine, the worm traverses each directory in the system (including all those accessible through a file shares) and write a copy of itself to disk using the name "README.EML". When a directory containing web content (e.g., HTML or ASP files) is found, the following snippet of Javascript code is appended to every one of these web-related files:

```
<script language="JavaScript">
window.open ("readme.eml", null,
"resizable=no, top = 6000, left = 6000 ")</script>
```

This modification of web content allows further propagation of the worm to new clients through a browser or browsing of a network file system.

On the basis of the described behavior of the attack we can find out the most appropriate elements of each stage as shown in Fig. 7.

On the basis of above facts we find out the Nimda attack standard weight as follows:

$$W_{1,1} + W_{2,2} + W_{3,2} + (W_{4,1} \text{ or } W_{4,9}) + W_{5,4} + W_{6,2} + W_{7,2} + W_{8,2} = 0.025 + 0.075 + 0.125 + (0.150 \text{ or } 0.350) + 0.675 + 0.725 + 0.850 + 0.950 = \{3.575, 3.775\} \quad (6)$$

COMPARISON OF ABOVE COMPUTED SET OF WEIGHTS OF CYBER ATTACKS AND THEIR CLASSIFICATION

All the computed weights of different attacks from equation-(2) to equation-(6) are shown in Table 2. On the basis of these data we can find the graphical analysis shown in Fig. 8.

From the Fig. 8 we can easily conclude that Nimda and Slammer are in one category i.e. worm, W32/HLLP.Philis.jn is a Trojan which is another category and Stealth virus and boot sectors are in third category i.e. virus. In case of Stealth virus and boot sector some of the features are common but some are not, therefore, only one value 3.875 is common, which is due to replication and execution of viral file.

Table 2: Corresponding computed weights of different cyber attacks

Sr. No.	Malicious object name	Set of metrics weights
1.	Boot sector virus	{3.875, 3.900, 3.925, 3.950}
2.	Stealth virus	{3.850, 3.875}
3.	W32/HLLP.Philis.jn	{3.675, 3.750, 3.800}
4.	Slammer	{3.575, 3.775}
5.	Nimda	{3.575, 3.775}

ALGORITHM FOR CYBER ATTACK CLASSIFICATION BY GAME THEORETIC WEIGHTED METRICS APPROACH

To classify an attack, we divide the algorithm into two parts; first, to compute the actual weight of attack according to the available metrics and second, to compare the actual weight with the standard weight of already available attacks to find the closer type of attack. In the first part, we use the goal theoretic approach to find out the most appropriate category of cyber attack. As the cyber attacks are going to more and more sophisticated with time, therefore, isolated or totally independent behavior of attack may not be their but it may be a mixture of two or more than two different types. Hence, we need to find out the dominant part of the behavior by using the game theoretic approach to categorize the cyber attack in the most suitable category. To fulfill the proper causes-effects chain, cyber classification metrics is having many stages to classify the cyber attacks. Hence, for

Algorithm for cyber attack classification by game theoretic weighted metrics approach

```
WA: Actual weight for attack in the metrics=0
w: Weight for attack in the most concern behavior in the stage=0
For all stages in the metrics
{
  For all elements in the stage
  {
    Select the component through maximum search process of game playing
    w=Compute the weight of selected component
  }
  WA = WA + w
}
For all available standard weights of different attacks
{
  Compute the closest type of attack category
}
```

each stage, we need to find out the most appropriate weight by using game theoretic approach and finally calculate the actual weight for the attack after adding all weights.

CONCLUSION

The paper presents a simple approach to classify the cyber attacks. The concepts of game theory and metrics are used to give the better cyber attack classification taxonomy. The metrics presentation is extendable, as new attacks can be added to it. We are using game theory concept, hence weighted stages are there to classify the cyber attack. The standard weights of the metrics are used as the base line to classify the cyber attacks in the proper category. As new attacks are identified with time, these need to be classified in their corresponding class. The most accurate categorization of cyber attacks can be done by the help of our approach. The approach is simple and extendable; as new characters of the newly identified attacks can be added to the attack characteristic metrics and the corresponding unique weight to the character are assigned by the proposed formula easily. Besides this, our approach clearly represents the cause effect relationship for all possible cyber attacks which helps to find the appropriate solution to restrict them in the Internet.

ACKNOWLEDGEMENT

The authors would like to thank the editor and the anonymous referees for their useful feedback and helpful comments which substantially improved the paper.

Appendix

Virus: A computer virus is represented by a self-replicating program which attaches itself to a program or file. It can spread from one computer to another, leaving infections as it travels. Almost all viruses are attached to an executable file, that is, the virus may exist on computer but it cannot infect computer unless someone run or open the malicious program. Virus cannot be spread without a human action such as running an infected program.

Worm: A worm is considered to be a sub-class of a virus. It spreads from computer to computer and has the capability to travel without any help from a person. It travels with the file or information on the system. It self-replicates on the system and sends thousands of its copies to create a huge devastating effect.

Trojans: The Trojan appears to be a useful program but damage the system when it installed on computer. The mastermind behind the Trojan can control the infected system without the knowledge of the owner. Trojans can change desktop, add silly active desktop icons, deleting files and destroying information on system. Trojans can also create a backdoor on computer. Unlike viruses and worms, Trojans do not reproduce by infecting other files nor do they self-replicate.

Spam: Spam is Unsolicited Bulk Email ("UBE"). Unsolicited email or Bulk email is not spam but the combination of the two is a spam. In other words, a large collection of unsolicited messages each of them having same contents are sent or posted is a spam.

DoS: Denial of service attack (DoS) is a type of attack on a network that floods it with useless traffic. Many DoS attacks, such as the Ping of Death and Teardrop attacks, exploit limitations in the TCP/IP protocols. Already known DoS attacks can be restricted by the help of available software but there is always a danger of new DoS attacks.

Phishing: The act of sending an e-mail to a user falsely claiming to be an established legitimate enterprise in an attempt to scam the user into surrendering private information that will be used for identity theft. The e-mail directs the user to visit a Web site where they are asked to update personal information, such as passwords and credit card, social security and bank account numbers, that the legitimate organization already has. The Web site, however, is bogus and set up only to steal the user's information.

REFERENCES

1. Hansman, S. and R. Hunt, 2005. A taxonomy of network and computer attacks. *Computer and Security*, 24 (1): 31-43.
2. Tocheva, K., G. Erdelyi, A. Podrezov, S. Rautiainen and M. Hypponen, 2001. F-Secure Computer Virus Information Pages: Nimda. Retrieved January, 2007 from <http://www.f-secure.com/v-descs/nimda.shtml>
3. Harish, B., 2004. Cyber Attack Profiling Using Cause Effect Networks. Thesis, Arizona State University.
4. Ye, N., B. Harish and T. Farley, 2005. Attack profiles to derive data observations, features and characteristics of cyber attacks. *Information Knowledge Systems Management*, IOS Press, Amsterdam, 5 (1): 23-47.

5. Kjaerland, M., 2005. A classification of computer security incidents based on reported attack data. *Journal of Investigative Psychology and Offender Profiling*, 2 (2): 105-120.
6. Ammann, P., D. Wijesekera and S. Kaushik, 2002. Scalable, graph-based network vulnerability analysis. *Proceedings of the 9th ACM conference on Computer and communications Security*. ACM Press, New York, NY, USA, pp: 217-224.
7. Wang, Y.M., Z.L. Liu, X.Y. Cheng and K.J. Zhang, 2005. An Analysis Approach for Multi-Stage Network Attacks. *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, Guangzhou, pp: 3949-3954.
8. Lindqvist, U. and E. Jonsson, 1997. How to Systematically Classify Computer Security Intrusions. *Proceeding of IEEE Symposium on Security and Privacy*, London, pp: 154 – 163.
9. Gegick, M. and L. Williams, 2005. Matching Attack Patterns to Security Vulnerabilities in Software-Intensive System Designs. *ICSE-SESS'05*, St. Louis, Missouri, USA, pp: 1-7.
10. Gegick, M., 2004. Analyzing Security Attacks to Generate Patterns from Vulnerable Architectural Patterns. MS, NCSU, Raleigh.
11. Howard, M. and D. LeBlanc, 2003. *Writing Secure Code* (2nd ed.), Redmond: Microsoft Corporation. <http://www.f-secure.com/v-descs/nimda.shtml>
12. Schumacher, M. and U. Roedig, 2001. Security Engineering with Patterns. *8th Conference on Pattern Languages of Programs*.
13. Viega, J. and G. McGraw, 2002. *Building Secure Software How to Avoid Security Problems the Right Way*. Boston: Addison-Wesley.
14. Burke, R., B. Mobasher, C. Williams and R. Bhaumik, 2006. Classification features for attack detection in collaborative recommender systems. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM Press, New York, USA, pp: 542 – 547.
15. Dahmouni, H., S. Vaton and D. Rossé, 2007. A Markovian Signature-Based Approach to IP Traffic Classification. *MineNet'07*, San Diego, USA.
16. Kumar, S., 1995. *Classification and Detection of Computer Intrusions*. Doctoral Thesis. UMI Order Number: UMI Order No. GAX96-01522, Purdue University.
17. Tidwell, T., R. Larson, K. Fitch and J. Hale, 2001. *Modeling Internet Attacks*. *Proceedings of the IEEE Workshop on Information Assurance and Security*. Unites States Military Academy, West Point, NY.
18. Howard, J.D., 1998. *An Analysis of Security Incidents on the Internet*. Doctoral Thesis. UMI Order Number: UMI Order No. GAX98-02539, Carnegie Mellon University.
19. Cohen, F., 1987. *Computer Virus-Theory and Experiments*. *Computer & Security*. Elsevier Science Publishers, B.V. (North-Holland), 6: 22-35.
20. Cohen, F., 1989. *Computational Aspects of Computer Viruses*. Elsevier Science Publishers, B.V. (North-Holland), 8: 325-344.
21. Cohen, F., 1992. *A Formal Definition of Computer Worms and Some Related Results*. Elsevier Science Publishers, B.V. (North-Holland), 11: 641-652.