Measuring Zero Day Susceptibilities

Sachin.C.Raykar
M.Tech, 4th Semester
Dept. of Computer Science & Engineering
AMC Engineering College, Bangalore

Jayashubha J
Associate Professor
Dept. of Computer Science & Engineering
AMC Engineering College, Bangalore

***Abstract*—Computer networks have long become the courage system of enterprise information systems and significant infrastructures on which our societies are more and more dependent. By empowering a direct comparison of dissimilar security solutions with respect to their relative effectiveness and network security metric may give quantifiable evidence to aid security practitioners in securing computer networks. The security risk of un-known susceptibilities has been un-measurable due to the less certain nature of software flaws. This causes a significant issue to security metrics, because a more secure arrangement would be of little value if it were equally vulnerable to zero-day attacks. We presented a robust security metric, k-zero day safety. Our security metric counts how many susceptibilities would be necessary to compromising network benefits; a larger count imply more security because the likelihood of having more unknown susceptibilities exploitable, applicable, and available all at the same time will be considerably lower. We applied the presented metric to the sensible issue of network hardening and extended the metric to distinguish various hardening options.**

***Keywords*— Software flaws; security metric; k-zero day safety; susceptibilities; hardening;**

I. Introduction

Containing the spread of worms or malicious code is a pressing problem. End system patching, due to the large time scale involved in achieving significant patch deployment, is by itself insufficient to protect network systems. For both technical (e.g., disruption, unreliability, irreversibility) and nontechnical (e.g., awareness, laxity) reasons [2], [3], software and signature patches incur a significant time lag before they are adopted, and even then, the deployment level is partial.

Computer networks have long become the courage system of enterprise information systems and significant infrastructures on which our societies are increasingly dependent. However, the severity and scale of security threats to computer networks have sustained to grow at an ever increasing speed. Possible consequences of a security attack have also become more and more serious as many high-profile attacks are allegedly targeting. Not only the computer applications but also military satellites, industrial control systems at nuclear power plants and entrenched heart defibrillators.

The main problems in securing computer networks is be short of means for directly measuring the relative success of dissimilar security solutions in a given network, because “you can’t get better what you can’t measure”. Indirect measurements, such as the negative rates and false positive of an intrusion detection firewall/system, may sometimes be obtain through laboratory testing, but they typically are very little about the real efficiency of the solution when it is deploy in a real-world network, it may be very dissimilar from the testing environment. In practice, choosing or selecting and deploying a security solutions is still heavily rely on human expert experiences follows the trial-and-error approach, which renders those errands an art, instead of a science. In such a setting, a system security metric is satisfying in light of the fact that it would empower a direct estimation and difference of the measures of security gave by diverse security arrangements.

Existing methods on network security metrics typically assign numeric values to susceptibility based on known facts about the susceptibilities. However, when we considered zero-day attack those methodologies is no longer applicable. In fact, well-liked criticism of past efforts on security metrics is that they cannot deal with unidentified susceptibilities, which are generally believed to be un-measurable. Unfortunately, without considering unknown susceptibilities, a security metric will only have questionable value at best, because it may decide a network configuration to be more secure while that arrangement is in fact equally susceptible to zero-day attacks. We, in this way, fall into the non-conviction that security is not quantifiable until we can alter all conceivable security imperfections yet we positively needn't bother with security metric by any means.

There exist numerous standardization efforts on security metrics, such as the Common Susceptibility Scoring System (CVSS) [24] and, more recently, the Common Weakness Scoring System (CWSS) [7]. Latter on software weaknesses, the former focus on ranking known susceptibilities,. Both CVSS and CWSS measure the relative severity of individual susceptibilities in segregation and do not address their overall impact. On the other hand, these efforts form a practical foundation for research on security metrics, as they provide security analysts and vendors standard ways for conveying numerical scores to known susceptibilities that are already available in public susceptibility databases, such as National Susceptibility Database (NVD) [25].

The authors in [13] describes that different security metrics will provide only a biased view of security, and the authors then presented a framework for grouping such metrics based on their relative significance. A recent work presents a risk management framework using Bayesian networks to enumerate the chances of attacks and to develop the security mitigation and the management plan [21]. Another study of several CVSS-based susceptibility metrics shows correlation between metrics and the time to cooperation of the system [11]. In our modern work, we have presented Parallel to our work on the probabilistic security metrics, in [12], the general framework for designing the network security metrics [8], a probabilistic approach [16]. Bayesian network- based metrics [9], address several significant issues in calculating such metrics including the dependency between the different attack sequences in an attack graph and the cyclic structure in such graphs.

Most previous work focuses on the developing security metrics for known susceptibility in a network. A few exceptions include in a empirical study on the total number of the zero-day susceptibility available on the single day based on existing facts about the susceptibilities [22], a report on the fame of zero-day susceptibilities among attackers [10], an empirical study on software susceptibilities' life cycles [19], and more recently an effort on estimate the effort necessary for developing new exploits [6].

Fig. 1 shows a toy example in which host1 and host2 comprise the internal network. The firewall allows all outbound association requests but blocks in-bound requests to the host2. Let the main security concern is whether any attacker on host0 can get the root permission on host2. Clearly, if we consider all the known susceptibilities services to be free, then attack graph or a susceptibility scanner will both conclude the same result that this network is secure attackers on host0 cannot obtain the root permission on host2. Now, consider the following two ip-tables policies:

- Policy-1: The ip-table rules are left in a default configuration that accepts all requests.
- Policy-2: The ip-table rules are configured to only allow only particular IPs and exclusive of host0, to access the ssh_service.

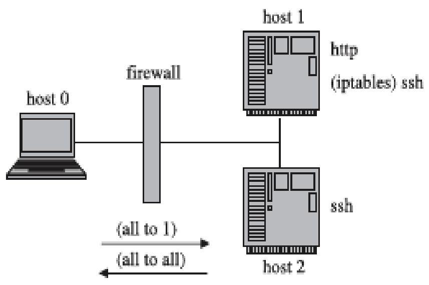

Fig 1: An example of network.

Clearly, the network is already secure; policy-1 will be preferable because of its simplicity (no special ip-tables rules need to be configured/set by the administrator) and functionality (any external host may connect to the ssh_service on host1).

By empowering a direct examination of the diverse security arrangements regarding their relative adequacy, network security metric may give a quantifiable evidence to assist security professional in securing computer networks. However, research on security metrics has been caught up by difficulties in handling zero-day attacks exploits unknown susceptibilities. The security risk of unknown susceptibilities has been considered as something un-measurable due to less certain nature of software flaws. This causes a most part of the difficulty to security metrics, because a major secure configuration would be a little value if it were equally vulnerable to zero-day attacks.

We propose a novel security metric that is k-zero day safety to address described issue. Instead of attempting to rank unknown susceptibilities, in our method, metric counts how many such susceptibilities would be required for compromising network possessions; a big count implies more security because the probability of having more unknown susceptibilities available, exploitable and applicable, all at the same time will be considerably lower. We applied the presented metric to the realistic issue of network hardening and extended the metric to typify various hardening options; we also described in details how the abstract model may be instantiated for given networks. The resolve to unknown susceptibilities is carried out by disabling the network services provided to each host which are violating the network rules.

II. K-ZERO DAY SAFETY METRIC MODEL

This metric model will permit a more centered talk on fundamental parts of the k-zero day safety. Extra highlights will be presented in later areas when they are required.

Considering following information about the network:

- A collection of hosts {*0, 1, 2, F*} (F for the firewall).
- The Connectivity relation {(0,F), (0,1), (0,2), (1,F), (1,0), (1,2),(2,F),(2,0),(2,1) }.
- Services {*http*, *ssh*, *iptables*} on host 1, {*ssh*} on host 2, and {firewall} on host F.
- Permissions {*user*, *root*}.

The main design rationale here is to hide inward details of hosts while focusing on the interfaces (services and connectivity) and crucial security properties (permissions). A few subtleties are as follows: First, hosts are intended to include not only computers but all networking devices potentially powerless against zero-day attacks (e.g., firewalls). Second, a currently disabled connectivity still needs to be considered since it may potentially be re-empowered through zero-day attacks (e.g., on firewalls). Third, only remote services (those remotely accessible over the network), and security services are considered. Modelling nearby services or applications is not always feasible. For this purpose, permissions under which services are running and those that can be potentially obtained through a privilege escalation will both are considered.

Definition 1(Network)

Next, we model zero-day exploits. The very idea of unknown vulnerability implies that we cannot expect any vulnerability-specific property, such as exploitability or impact. Rather, our model is taking into non particular properties of existing vulnerabilities. In particular we characterize two sorts of zero-day vulnerabilities. First, zero-day vulnerability in services are those whose details are unknown except that their exploitation requires a network connection between the source and destination hosts, a remotely available service on the destination host, and existing permission on the source host. In addition, exploiting such vulnerability can potentially yield any permission on the destination host.

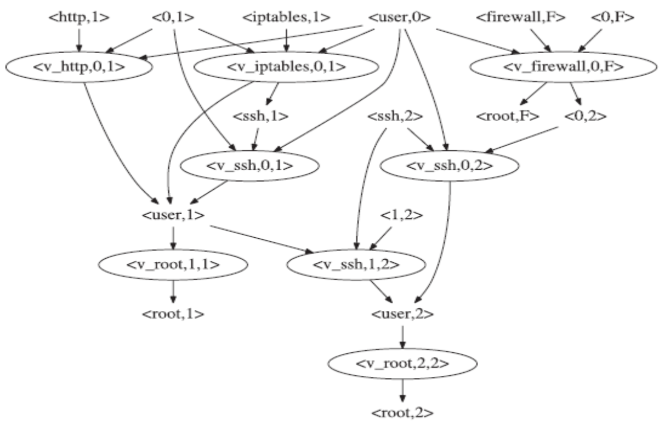

Fig 2: An Example of zero –day attack graph

Definition 2 (Zero-day exploit)

Since we have characterised zero-day exploits, it is direct to extend a traditional attack graph with zero-day exploits. In particular, a zero-day attack graph is simply a directed graph composed of both zero-day and known exploits, with edges indicating from preconditions to corresponding exploits and from exploits to their post conditions. Fig 2 shows the zero-day attack graph.

We are now ready to characterize the k-zero day safety metric. We do as such in three steps. First and foremost, we model two different cases in which two zero-day exploits should be counted only once, that is, either when they involve the same zero-day vulnerability or when they correspond to a trivial permission escalation due to the lack of isolation techniques. In spite of the fact that the equivalence relation in those two cases has very different semantics, the impact on our metric will be the same. The metric function k0d(.) counts how many exploits in their symmetric difference are distinct. The k-zero day safety metric is characterised by applying the metric function k0d(.) to the negligible attack sequences leading to an asset.

III. K-ZERO DAY SAFETY COMPUTATION

This section presents the algorithms for computing the k-zero day safety metric models. The first two algorithms are exhibited in [17] and the third algorithm is new presented algorithm in this paper.

A. *Compute the Value of k*

To compute the k-zero day safety of network, function/algorithm k0d_Bwd is shown in Fig. 3. First derive a logic proposition of each advantage in terms of exploits. In Disjunctive Normal Form (DNF) of the derived proposition, every conjunctive section will corresponds to a nominal set of exploit that can jointly compromise the benefit. Therefore, the metric value of that benefit can be calculated by applying the metric function k0d (.) to the conjunctive clauses and taking the minimum value among the consequences. The negated condition in a benefit will be replaced with the inversion of exploits, whereas the latter those will not be further processed (as indicated in line 6).

Next, we will reduce the known NP-hard problem by finding the minimum attack (i.e., an attack sequence with the minimum numbers of exploit) in attack graph [1], [20] to the present problem. First of all, the reduction cannot be trivially achieved by simply replacing every known exploit with the zero-day exploit in a given attack graph of the known exploits, because, unlike the former and a fixed number of hard-coded pre-condition and post-condition that may avoid them from fitting in the place of the known exploit.

We construct a zero-day attack graph G' by injecting the zero-day exploit before every known exploit. Firstly, let the G″ = G'. Then, for every known exploit 'e' of the service 's' from the host-h_1 to host-h_2, we inject the zero-day exploit e' with the post-conditions {<s, h_2>, $p_{useless}$}, where $p_{useless}$ is a permissions designed that not to be precondition of any of the exploit. Then we have the following two facts. Firstly, executing the 'e' requires e' to be executed; on the other hand, if e' needs to be executed, then, the only reason must be it should be satisfy the condition <s, h_2> and consequently to execute 'e'. Secondly, among the three conditions in pre(e')={<s', h_2>, <h_1, h_2>, <p_{least}, h_1>}, the first is an initial condition and the last two are the members of pre(e'). Therefore, G and G' are isomorphic if we are regard 'e' and e' as a single exploit and ignore the starting condition <s', h_2>. Then, for every known exploit 'e' involving to only one host-h, we replace 'e' with the zero-day exploit e' and a known exploit e″ fulfilling the post(e″)=post(e), pre(e″) = pre(e) \ {<p,h> U {<p', h>}, where <p,h> ϵ pre(e), and {<p', h>} are 2 permissions. We also let post(e') = {<p', h>}, and design relation $\equiv_v$ in such a way that e' is not allied to any other zero-day exploits in h. We have 2 similar facts as above.

```
Procedure k0d_Bwd
Input: Zero day attack graph $G$, a set of assets $A$ with the valuation function $v(.)$
Output: An integer $k$
Method:
1.  For each asset $a \in A$
2.      Let $L$ be the logic proposition representing $a$
3.      While at least one of the following is possible, do
4.          Replace each initial condition $c$ with $TRUE$
5.          Replace each condition $c$ with $\bigvee_{e \in \{e' : c \in post(e')\}} e$
6.          Replace each non-negated exploit $e$ with $e \wedge (\bigwedge_{c \in pre(e)} c)$
7.      Let $L_1 \vee L_2 \vee \ldots L_n$ be the DNF of $L$
8.      Let $k_a = min(\{k0d(F_i \cap E_0, \phi) : F_i$ is set of non-negated exploits
          in $L_i, 1 \leq i \leq n\})$
9.  Return $\lceil \sum_{a \in A}(k_a \cdot v(a)) / \sum_{a \in A} v(a) \rceil - 1$
```

Fig 3: Computing the value of k.

if we design $\equiv_v$ in such a way that no 2 zero-day exploits are related, then we have |q| = k0d(q' ∩ E_0, | ϕ). Therefore, for any of the non-negative integer k, finding q' in G' to minimize k0d(q' ∩ E_0, | ϕ) will immediately yield the q in G that also minimizes the |q|, and the latter is fundamentally the minimum attack problem. This proves and concludes the NP-hard problem.

B. *Deciding the k-Zero Day Safety for a Given Threshold*

Many real time or practical purposes, it might be sufficient to know that every benefit in a network is a k-zero day safe for a given threshold value k, even though the network may be in the realism k'-zero day safe for a number of unknown k' > k (identifying k0 is intractable). Fig. 4 shows a recursive function k0d_Fwd whose complexity is the polynomial in the size of a zero-day attack graph, if the value k is a constant compared to the size. Approximately speaking, the procedure attempts to negotiation each benefit with less than value k distinct zero-day exploits through a forward search of a limited depth. The benefit is not k-zero days safe if any branch of the search succeed, and vice versa. The complexity of this procedure is polynomial in the size of a zero-day attack graph if k value is a constant.

C. *Compute the k-Zero Day Safety as Shortest Paths in a DAG*

Although finding the value of k is NP-hard in general, efficient solutions may be there for special cases of practical relevance. We study that such case where k can be calculated in polynomial time when the following 2 assumptions hold on the given zero-day attack of the graph they are:

- Firstly, the conjunctive relationships between conditions of the hosts are mostly limited to be within every host or every small group of hosts. This is true if the remote hosts are maybe used as a stepping stones so every remote exploit will require only a preference on the remote host. We can then consider each condition as the vertex of an Acyclic Directed Graph (DAG) [18]. Finding the value of k amounts to finding the shortest path along which the collection of the zero-day exploits yields minimum metric value.
- Secondly, the similarity between zero-day exploits modelled by the relation $\equiv_v$ will also mostly be restricted to be with a host or small group of hosts. For these zero-day exploits that may be later related to other exploits by $\equiv_v$, we keep them in a set as the first part of the distance. For all other exploits, we only keep the result of applying the k0d (.) metric as the second part of the distance. We can then communicate such distances along each edge.

In Fig. 4, the Sub_Procedure k0d_Graph builds the DAG, based on the given zero-day attack graph and the benefit. The main function then follow a standard algorithm to determine the shortest path in a DAG [5], with a few changes.

```
Procedure $k0d\_Fwd$
Input: A zero day attack graph $G$, an asset $a$, $k > 0$, $T_e = \phi$, $T_c = C_I$
       //$T_e$ and $T_c$ denotes exploits and conditions visited so far, respectively
Output: $TRUE$, if $k0d(a) > k$; $FALSE$, otherwise
Method:
1. If $k0d\_reachable(T_e, T_c) \wedge k0d(T_e) < k$
2.     Return $FALSE$
3. ElseIf $k0d(T_e) \geq k$
4.     Return $TRUE$
5. Else
6.     For each $e \in E_0 \setminus T_e$ satisfying $pre(e) \subseteq T_c$
7.         If $\neg\ k0d\_Fwd(G, a, k, T_e \cup \{e\}, T_c \cup post(e))$
8.             Return $FALSE$
9.     Return $TRUE$

Sub-Procedure $k0d\_Reachable$
Input: $T_e, T_c$
Output: $TRUE$ or $FALSE$
Method:
10. While $(\exists e \in E_1 \setminus T_e)(pre(e) \subseteq T_c)$
11.         Let $T_e = T_e \cup \{e\}$
12.         Let $T_c = T_c \cup post(e)$
13. Return $(\bigwedge_{c \in T_c} c \rightarrow a)$
```

Fig 4: Determining the k-zero day safety for a given k.

Firstly, instead of the single number, each distance is now a set of pairs of <x, y>, where x denotes the result of applying k0d (.) to exploits that, later will not be related to the others by $\equiv_v$, whereas y denotes the converse. Secondly, the reachable edges are collected to find whether an exploit may later be associated to others by $\equiv_v$ (line 8).

```
Procedure $k0d\_Shortest$
Input: A zero day attack graph $G$, an asset $L$
Output: A non-negative real number $k$
Method:
1. Let $G_s$ be a DAG with vertex $L$, and $A$ be an empty array
2. Let $\langle G_s, A \rangle = k0d\_Graph(G, L, G_s, A)$
3. Let $vlist$ be any topological sort of $G_s$
4. Let $dist_L = \{\langle 0, \phi \rangle\}$ and $dist_x = \{\langle \infty, \phi \rangle\}$ for any other vertex $x$
5. While $vlist$ is not empty, do
6.     Delete the first vertex $u$ from $vlist$
7.     For each outgoing edge $\langle u, v \rangle$ of $u$
8.         Let $elist$ be the set of all edges reachable from $v$
9.         For each $\langle x, y \rangle \in dist_u$
10.            Let $y' = \{e : e \in y \cup A[\langle u, v \rangle], \exists e' \in elist\ \exists e'' \in A[e']$
                $e \equiv_v e''\}$
11.            Let $x' = x + k0d((y \cup A[\langle u, v \rangle] \setminus y') \cap E_0, \phi)$
12.            Let $dist_v = dist_v \cup \langle x', y' \rangle$
13.        While $(\exists \langle x, y \rangle, \langle x', y' \rangle \in dist_v)(x \geq (x' + k0d(y' \cap E_0, \phi)))$
14.                Delete $\langle x, y \rangle$ from $dist_v$
15. Return $min(\{x : \langle x, \phi \rangle \in dist_d, d$ is a dummy vertex $\})$

Sub_Procedure $k0d\_Graph$
Input: A zero day attack graph $G$, an asset $L$, a DAG $G_s$, an array $A$
Output: Updated $G_s$ and $elable$
Method:
16. Do
17.   (Lines 4-6 of Procedure $k0d\_Backward$)
18.   Let $L$ be its DNF
19.   While there exists a conjunctive clause $l$ in $L$ including multiple conditions
20. for each conjunctive clause $l$ in $L$
21.     If $l$ includes a condition $c$
22.         Add vertex $c$ and edge $\langle L, c \rangle$ to $G_s$
23.         Let $A[\langle L, c \rangle]$ be the set of exploits in $l$
24.         Let $\langle G_s, A \rangle = k0d\_Graph(G, c, G_s, A)$
25.     Else
26.         Add a dummy vertex $d$ and edge $\langle L, d \rangle$ to $G_s$
27.         Let $A[\langle L, d \rangle]$ be the set of exploits in $l$
28. Return $G_s$
```

Fig 5: Computing k-zero day safety as shortest paths in DAG.

The efficiency of Sub-Procedure/function k0d_Graph is exponential in the number of exploits and the conditions involved in the loop at lines 16-19. Therefore, if the first hypothesis perfectly holds, then this loop will always terminates after processing a single host. If we regard the number of exploits and the conditions on each of host as a constant, then the multifaceted nature of the sub-procedure will be linear in the number of hosts. Second, the intricacy of the main function depends on the size of the distance of each vertex. If the second hypothesis holds perfectly such that each of the distance has a negligible size, then the complexity of the main function will be dominated by the processing of reachable edges is elist and their labels A (line 10). Since each edge in Gs is visited exactly once by the main loop and the size of elist is linear to the number of the such edges, the processing of elist takes quadratic time in the number of the edges in G_s, which is roughly $O(|H|^4)$. Finally, multiplying this by the size of A, we will be having $|H|^4 \cdot |E_0|$.

D. *Zero-Day Attack Model*

To instantiate the zero-day attack graph model, we need to consider both

- The zero-day exploits, and
- Exploits of the known susceptibilities.

The former can directly be composed based on the network model, with no extra information needed, because, unlike known susceptibilities, all zero-day exploits have hard-coded conditions. On the other hand, exploits of known susceptibilities must be identified, together with their pre- and post-conditions (which are specific to each exploit). Known susceptibilities may be discovered through various susceptibility scanners, and their pre- and post-conditions may be obtained from the public susceptibility databases. These may also be directly available from the existing attack graphs of the known susceptibilities. One delicacy here is that the exploits not reachable from the benefit can no longer be omitted because they may now be reachable from the benefit with the help of zero-day exploits.

Conventional attack graphs are practical for realistic applications, with efficient implementations [27] and commercial tools [15] available. A zero-day attack graph would have been comparable complexity as traditional attack graphs, because of the number of added zero-day exploits (which depends on the number of remote services and privileges) on each host should be comparable to the number of known susceptibilities.

IV. EXPERIMENTAL STUDY

In this experimental section, we demonstrate through a series of experimental studies that our metric can make known interesting and sometimes astonishing results, for small network which all are not always obvious; for well-built and for more practical networks, the methodical approach to security evaluation using the algorithms and metric, thus, become still more important.

A. *Firewall*

In Fig. 6, a personal firewall on host3 allows only outbound requests to host4 and in-bound connection requests from host1. The firewall on host4 allows outbound requests to host5 and inbound requests from host3. The firewall on host6 allows in-bound requests from host5 or host7. Moreover, let the personal firewall service on hostC has a known susceptibility that may permit attackers to establish the connections to ftp service running on the host3. Root privilege on host6 is the most concerned one.

We can illustrate that the shortest attack sequences are shown below

[<v_{ftp}, 0, 2>, <$v_{firewall2}$, 2, firewall2>,

<v_{http}, 2, 7>, <v_{ftp}, 7, 6>] and

[<v_{ftp}, 0, 2>, <$v_{p\_firewall1}$, 2, 3>, <v_{ftp}, 2, 3>,

<v_{nfs}, 3, 4>, <v_{ssh}, 4, 5>, <v_{ftp}, 5, 6>]

Since $v_{p\_firewall1}$ is known, both sequences require 2 different zero-day susceptibilities.

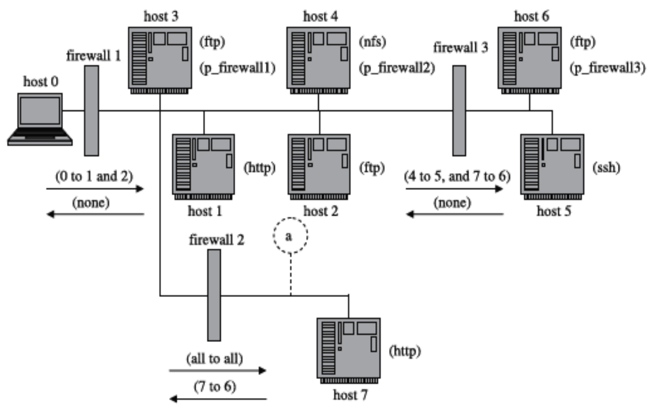

Fig 6: Experimental study: firewall.

Suppose, as a temporary work around, the administrator decides to move host 3 to a location behind firewall2, and remove its personal firewall the p_firewall1 but it keeps the same network access controls by adding the extra rules to the firewall2 to allow only connection requests from host1 to the host3 and from host3 to the host4.

On first look, the above solution may seem to be a reasonable approach. However, by applying the presented metric, we can able to show that the approach doing this will actually cause to be the network less secure. Specially, after moving host3 to new location, it can be show that the shortest attack sequence becomes the [<v_{http}, 0, 1>, <v_{ftp}, 1, 3>, <v_{http}, 3, 7>, <v_{ftp}, 7, 6>], which requires only 2 different zero-day susceptibilities, and k decreases by 1.

B. *Backup of asset*

In this experimental study, we demonstrated that by placing an asset backup at the different locations inside a network and the amount of security with respect to the benefit may actually increase, decrease, or remain the same.

In Fig. 7, let we are most anxious by the root privilege on the host4. We assume that the known susceptibility exists in the http service on both the hosts i.e., host1 and host5, exploiting which provides the root privilege on the host. Finally, suppose we have chosen 3 aspirant positions for placing the back-up server for the host4, as indicated by the 3 dashed line circles.

Without introducing any asset backup, we can find the 3 shortest attack sequences, they are:

[<v_{http}, 0, 1>, <v_{ssh}, 1, 3>, <v_{nfs}, 3, 4>]

[<v_{smtp}, 0, 2>, <v_{nfs}, 2, 4>] and

[<v_{http}, 0, 1>, <v_{smtp}, 1, 2>, <v_{nfs}, 2, 4>]

We can see that v_{http} is a known susceptibility, and therefore, two different zero-day susceptibilities are needed to compromise host4.

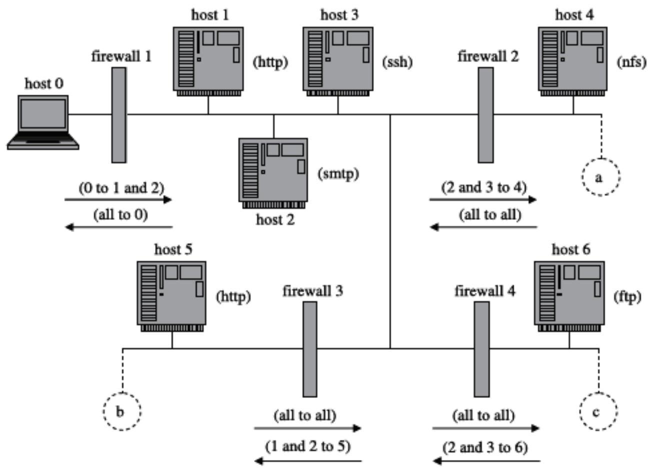

Fig 7: Experimental study: benefit backup.

V. CONCLUSION

Computer systems have long turned into the fearlessness arrangement of big business data frameworks and our social orders are progressively reliant on noteworthy bases. The seriousness and size of security dangers to Computer systems is kept on ascending at a regularly expanding pace. The security danger of obscure susceptibilities has been considered as a little unmeasurable because of the minimal certain view of programming defects. This makes a significant trouble for security measurements, because high secure configuration would be of less value if it were equally vulnerable to zero-day attacks.

We presented a robust security metric, k-zero day safety and address the k-zero safety issue. Instead of attempting to rank the unknown susceptibilities, our presented metric counts the how many such susceptibilities would be necessary for compromising network benefits; a greater count implies extra security because of the likelihood of having higher unknown susceptibilities applicable, exploitable and available all at the same time will be considerably lower. We applied the presented metric to the real time/practical issue of network extended and hardening the metric to distinguish various hardening options. We also described in details how the abstract model may instantiated for a given networks in real-time. Finally, we reveal how to apply the presented metric may lead to surprising and interesting results through a series of experimental studies.

REFERENCES

[1] P. Ammann, D. Wijesekera, and S. Kaushik, “Scalable, Graph- Based Network Susceptibility Analysis,” Proc. Ninth ACM Conf. Computer Comm. Security (CCS ’02), pp. 217-224, 2002.

[2] P. Szor. The Art of Computer Virus Research and Defense. AddisonWesley, 2005.

[3] H. Wang, C. Guo, D. Simon, and A. Zugenmaier. Shield: susceptibilitydriven network filters for preventing known susceptibility exploits. In Proceedings ofACM SIGCOMM, pages 194-204, 2004.

[4] H. Kim and B. Karp. Autograph: Toward automated, distributed worm signature detection. In Proceedings of USENIX Security Symposium, pages 271-286, 2004.

[5] E.W. Dijkstra, “A Note on Two Problems in Connection with Graphs,” Numerische Mathematik, vol. 1, pp. 269-271, 1959.K.

[6] T. Sommestad, H. Holm, and M. Ekstedt, “Effort Estimates for Susceptibility Discovery Projects,” Proc. 45th Hawaii Int’l Conf. System Sciences (HICSS ’12), pp. 5564-5573, 2012.

[7] MITRE Corp., “Common Weakness Scoring System (CWSS),” http://cwe.mitre.org/cwss/, 2010.

[8] L. Wang, A. Singhal, and S. Jajodia, “Measuring Network Security Using Attack Graphs,” Proc. ACM Third Workshop (QoP ’07), 2007.

[9] M. Frigault, L. Wang, A. Singhal, and S. Jajodia, “Measuring Network Security Using Dynamic Bayesian Network,” Proc. Fourth ACM Workshop Quality of Protection (QoP ’08), 2008.

[10] A. Greenberg, “Shopping for Zero-Days: A Price List for Hackers’ Secret Software Exploits,” Forbes, Mar. 2012.

[11] H. Holm, M. Ekstedt, and D. Andersson, “Empirical Analysis of System-Level Susceptibility Metrics through Actual Attacks,” IEEE Trans. Dependable Secure Computing, vol. 9, no. 6, pp. 825- 837, Nov. 2012.

[12] J. Homer, X. Ou, and D. Schmidt, “A Sound And Practical Approach to Quantifying Security Risk in Enterprise Networks,” technical report, Kansas State Univ., 2009.

[13] N. Idika and B. Bhargava, “Extending Attack Graph-Based Security Metrics and Aggregating Their Application,” IEEE Trans. Dependable and Secure Computing, vol. 9, no. 1, pp. 75-85, Jan./Feb. 2012.

[14] D. Kim, Kim,1. Oh, and 1. Jang. Tracing stored program counter to detect polymorphic shellcode. IEICE Transactions on Information and Systems, E91-D(8):2192-2195, 2008.

[15] S. Jajodia, S. Noel, and B. O’Berry, “Topological Analysis of Network Attack Susceptibility,” Managing Cyber Threats: Issues, Approaches and Challenges, V. Kumar, J. Srivastava, and A. Lazarevic, eds., Kluwer Academic, 2003.

[16] L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia, “An Attack Graph-Based Probabilistic Security Metric,” Proc. 22nd Ann. IFIP WG 11.3 Working Conf. Data and Applications Security, 2008.

[17] L. Wang, S. Jajodia, A. Singhal, and S. Noel, “k-Zero Day Safety: Measuring the Security Risk of Networks against Unknown Attacks,” Proc. 15th European Conf. Research Computer Security (ESORICS ’10), pp. 573-587, 2010.

[18] L. Wang, S. Noel, and S. Jajodia, “Minimum-Cost Network Hardening Using Attack Graphs,” Computer Comm., vol. 29, no. 18, pp. 3812-3824, 2006.

[19] M. Shahzad, M. Shafiq, and A. Liu, “A Large Scale Exploratory Analysis of Software Susceptibility Life Cycles,” Proc. 34th Int’l Conf. Software Eng. (ICSE ’12), 2012.

[20] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing, “Automated Generation and Analysis of Attack Graphs,” Proc. IEEE Symp. Security and Privacy (S&P ’02), 2002.

[21] N. Poolsappasit, R. Dewri, and I. Ray, “Dynamic Security Risk Management Using Bayesian Attack Graphs,” IEEE Trans. Dependable Secure Computing, vol. 9, no. 1, pp. 61-74, Jan. 2012.

[22] M. McQueen, T. McQueen, W. Boyer, and M. Chaffin, “Empirical Estimates and Observations of 0Day Susceptibilities,” Proc. Hawaii Int’l Conf. System Sciences, pp. 1-12, 2009.

[23] P. Porras, H. Saidi, and V. Vegneswaran, “An analysis of conficker’s logic and rondezvous points,” SRI International, Tech. Rep., February 2009.

[24] P. Mell, K. Scarfone, and S. Romanosky, “Common Susceptibility Scoring System,” IEEE Security and Privacy, vol. 4, no. 6, pp. 85-89,

[25] Nat’l Institute of Standards and Technology, “National Susceptibility Database Version 2.2,” http://www.nvd.org, May 2008.

[26] R. Lippmann et al., “Validating and restoring defense in depth using attack graphs,” in IEEE Military Communications Conference (MILCOM), 2006.

[27] X. Ou, W. Boyer, and M. McQueen, “A Scalable Approach to Attack Graph Generation,” Proc. 13th ACM Conf. Computer Comm. Security (CCS’ 06), pp. 336-345, 2006.