# Assessing Attack Impact on Business Processes by Interconnecting Attack Graphs and Entity Dependency Graphs

Chen Cao[1], Lun-Pin Yuan[1], Anoop Singhal[2], Peng Liu[1], Xiaoyan Sun[3], and Sencun Zhu[1]

[1] The Pennsylvania State University
[2] National Institute of Standards and Technology
[3] California State University, Sacramento
caochen11@mails.ucas.ac.cn,lunpin@psu.edu,anoop.singhal@nist.gov, pliu@ist.psu.edu,xiaoyan.sun@csus.edu,szhu@cse.psu.edu

**Abstract.** Cyber-defense and cyber-resilience techniques sometimes fail in defeating cyber-attacks. One of the primary causes is the ineffectiveness of business process impact assessment in the enterprise network. In this paper, we propose a new business process impact assessment method, which measures the impact of an attack towards a business-process-support enterprise network and produces a numerical score for this impact. The key idea is that all attacks are performed by exploiting vulnerabilities in the enterprise network. So the impact scores for business processes are the function result of the severity of the vulnerabilities and the relations between vulnerabilities and business processes. This paper conducts a case study systematically and the result shows the effectiveness of our method.

## 1 Introduction

Although enterprises and organizations have been paying ever more attention to cyber defense, today's cyber-attacks towards enterprise networks often undermine the security of business processes. The reason is directly related to several main limitations of existing cyber-defense practice, because the security of business processes heavily relies on the deployed cyber-defense measures and procedures.

Although a fundamental limitation of existing cyber-defenses is that zero-day attacks cannot be prevented, this limitation is clearly *not the only reason* why cyber-attacks can undermine security. In many, if not most, real-world cyber-security incidents, the security of business processes is actually undermined by known attacks.

Regarding why known attacks could sigificantly undermine the security of business processes, the following main reasons have been recognized in the research community. First, enterprises and organizations do not have the resources needed to patch all the known vulnerabilities. As a result, although the security

administrators are working hard to patch as many vulnerabilities as possible and as soon as possible, many vulnerabilities are actually in the "not yet patched" status when cyber-attacks happen. Another contributing factor to the result is that the time a vulnerability becomes known is *not* the time the corresponding patch becomes available.

Second, when cyber-attacks are happening, even if the intrusion detection system accurately detects the intrusions, the intrusion alerts and alert correlation results are still not able to directly tell "what should I do?" in terms of intrusion response. (In real-world enterprises new intrusion alerts keep on being raised, and the security administrators are already fully loaded.) It has been widely recognized in the research community [8, 18, 20] that there is a wide *semantic gap* between the information contained in intrusion alerts and how the cost-effectiveness of intrusion response is evaluated. On one hand, the cost-effectiveness of intrusion response is usually evaluated based on business process-level metrics (e.g., the number of customers affected by a cyber-attack, the number of tasks that need to be undone) and measurements. On the other hand, business process-level metrics are not really measured by intrusion detection systems.

Therefore, to achieve cost-effective intrusion response, this semantic gap must be bridged. To bridge the semantic gap, impact assessment is necessary. Although researchers have found the necessity of using entity dependency graphs [8] to assess the impact of attacks on business processes for quite a few years, the existing impact assessment techniques still face a key challenge. The challenge is two-fold: (1) impact assessment results cannot be automatically used to make recommendations on taking active cyber-defense actions; and (2) existing active cyber-defense techniques cannot be business-process-aware. That is, these techniques will not be able to directly state their effectiveness using business process-level measurements such as how much of what tasks will be accomplished by when.

In [19] it has been perceived that attack graphs and entity dependency graphs could be interconnected to address the above key challenge; however, no realistic case study has been conducted to validate the perceived method. As a result, the intrusion response research community still lacks essential understanding about (a) how to efficiently implement the perceived method; (b) whether it really works; and (c) how well it works.

The goal of this work is to efficiently design and implement the perceived method and conduct a realistic case study to assess the impact of attacks on business processes using not only system-level metrics (e.g., how many files are corrupted, which processes are compromised) but also business process-level metrics. We believe that this case study is a solid step forward towards bridging the aforementioned semantic gap.

The main contributions of this work are as follows.

- We propose the first efficient implementation of the method perceived in [19]. We extend the perceived method to make use of CVSS scores. We invent an algorithm to prune the raw interconnected graph. Through logic

programming, the implemented tool can automatically generate an interconnected graph, which interconnects an attack graph and an entity dependency graph, and calculate the impact scores of an attack on tasks in a business process.
- The first realistic case study is systematically conducted to show how the perceived method and our implementation can assess the impact of attacks on business processes using not only system-level metrics but also business process-level metrics.
- Through the case study, we also evaluate our implementation in several aspects such as scalability and running time.

## 2 Background

### 2.1 CVSS score

The Common Vulnerability Scoring System (CVSS) provides a way to measure the impacts of vulnerabilities and produce a numerical score for the attack impact [9]. The current version of this score system is version three, which is released in 2015. The system contains three metric groups: base score metrics, temporal score metrics, and environmental score metrics. A base score ranging from 0 to 10 is assigned to a vulnerability according to the base score metrics. The temporal score metrics and environmental score metrics can be used to refine the base score to better reflect the risks caused by a vulnerability to the user's environment. However, the temporal score metrics and environmental score metrics are optional. Therefore, in this paper we only use base score for impact analysis and still refer it as CVSS score. The National Vulnerability Database (NVD) provides a CVSS base score for almost all known vulnerabilities. A higher CVSS base score of a vulnerability implies that: 1) the vulnerability is easier to be exploited due to more vulnerable components and available technical means for exploitation; or 2) more impact on the availability, confidentiality, and integrity upon successful exploitation. Therefore, the base score can be leveraged to assess the impact of vulnerability exploitation on business processes in terms of both exploitability and impact.

### 2.2 Attack Graph

To analyze the impact of attacks on business processes, it's necessary to first understand how the vulnerabilities in an enterprise network can be used to compromise the host machines. Attack graph [1, 7, 10, 12, 17] is a very effective way to generate potential attack paths. Given the vulnerabilities, the attack graph is able to show the possible attack sequences to the final attack target.

MulVAL (Multihost, multistage Vulnerability Analysis) is an attack graph generation tool that models the interaction between software vulnerabilities and the system and network configurations [11]. It leverages Datalog [14] to model network system information (such as the vulnerabilities, configurations of each

machine, etc.) as facts and the interaction of various network components as rules. With these facts and rules, MulVAL can generate an attack graph showing the potential attack paths from the vulnerabilities to the attack goal. In the attack graph, facts and rules are represented by nodes with different shapes. There are two types of fact nodes: primitive fact nodes and derived fact nodes. The primitive facts nodes are denoted with boxes, which represents host and network configuration information. The derived fact nodes are denoted with diamonds, which are generated according to certain rules. The interaction rules are denoted with ellipses.

Fig. 1 shows a very simple attack graph containing only 5 nodes. In Fig. 1, if the conditions in node 1, 2 and 3 are satisfied, then the rule in node 4 can be applied. The eventual consequence is that the attacker is able to execute arbitrary code on the host machine (shown in node 5).

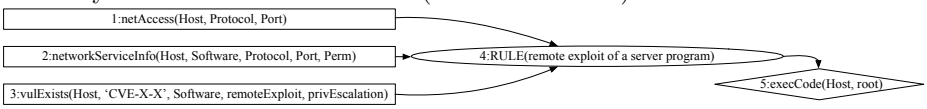



**Fig. 1.** An Example Attack Graph

The attack graph is essential for business process impact assessment, as it shows how the vulnerabilities can be leveraged to compromise the host machines. If the host machines are involved in the business processes, the impact of vulnerabilities on business processes can then be further analyzed.

### 2.3 Entity Dependency Graph

In an enterprise network, a business process is supported by a number of entities at several abstraction layers: asset layer, service layer and business process task layer. At the asset layer, an asset is (part of) a persistent disk and the file stored on the disk, a computer (hypervisors, desktops or servers), or a peripheral device. At the service layer, services represent the functionalities provided by hosts, such as web services, database services, etc. At the business process task layer, a business process is composed of one or more tasks.

An entity dependency graph [2] can be established due to the dependencies between the abstraction layers and the dependencies on each individual layer. Generally, the higher layer depends on the function of the lower layer. The business process task layer depends on the functionality provided by the services at service layer. One task may even depend on several services. The services further depends on the assets at the services layer. In addition, dependencies also exist at an individual layer. For example, at the business process task layer, a task may depend on another task.

## 3 Approach Overview

The primary goal of our paper is to assess the attack impact on business processes. Since attacks essentially exploit vulnerabilities in the enterprise network,

the attack impact heavily relies on the intrinsic characteristics of each individual vulnerability. Considering that the characteristics of vulnerabilities have been measured using the CVSS scores, the impact towards a business process can also be measured based on the scoring system. That is, an impact score can be generated for a business process to indicate the impact of attacks towards the business process. Therefore, the key problem need to be addressed is how to generate the impact score for a business process given the CVSS scores of involving vulnerabilities.

In this paper, we propose an three-step approach for business process impact assessment. The general idea is to generate an interconnected graph by analyzing the dependency relationships between vulnerabilities and attacks on hosts, between services and hosts, and between tasks and services. The approach takes three sets of knowledge units as the inputs and generates the business impact score as the output.

The three sets of knowledge units are respectively 1) Common Vulnerability and Exposure (CVE) system that provides information of publicly known vulnerabilities and their CVSS scores, 2) the vulnerability information generated by the vulnerability scanner, and 3) the business process dependency graph. The business impact assessment approach mainly involves the following steps:

Step 1: Instantiate the knowledge units with Datalog as facts and rules in MulVAL. Utilize MulVAL to generate an interconnected graph which consists of impact paths from the vulnerabilities.

Step 2: Prune the interconnected graph to get a more clear relationship between business processes and vulnerabilities.

Step 3: Calculate the impact score based on the CVSS scores of the vulnerabilities exploited in this attack.

### 3.1 Instantiate Knowledge Units

CVE system refers to the vulnerability database which contains all information about publicly known vulnerabilities. From this system, we can get the CVSS score of each vulnerability. The vulnerability information generated by vulnerability scanner contains the exact CVE IDs of each vulnerability and where these vulnerabilities are located in the enterprise network. By combining these two sources of knowledge, we can easily get the whole picture of these vulnerabilities, including CVSS score, CVE ID, and location in the enterprise network, etc. Such vulnerability information can be used to analyze the potential attacks that might happen, which may further impact the business processes. As the information represents facts about vulnerabilities in the network, we crafted fact nodes in MulVAL to instantiate the information.

Business process dependency graph describes how entities in the network depend on each other. Sun et al. [19] summarizes and bridges the semantic gap between the attack graph generated by MulVAL and the business process dependency graph. Hence, in this paper, we extend MulVAL to craft new fact nodes and new rule nodes to interconnect the attack graph and the business process dependency graph.

**Listing 1.1.** Example Interaction Rules Describing Three Dependency Relationships

```
interaction_rule(       /* And depends */
    (nodeImpact(Task):-
        node(Task, and, Task1, Task2), nodeImpact(Task1)
    ),
    rule_desc('An impacted child task affects an And task')
).
interaction_rule(       /* Or depends */
    (nodeImpact(Task):-
        node(Task, or, Task1, Task2),
        nodeImpact(Task1), nodeImpact(Task2)
    ),
    rule_desc('Both impacted child task affects an Or task')
).
interaction_rule(       /* Flow depends */
    (nodeImpact(Task):-
        node(Task, flow, Task1, Task2), nodeImpact(Task2)
    ),
    rule_desc('A flow node is impacted from its flow')
).
```

First of all, entities in a business process dependency graph become primitive fact nodes or derived fact nodes. Primitive fact nodes usually represent already known information, such as host configuration, network configuration, etc. Derived fact nodes are computed information by applying interaction rules towards primitive fact nodes.

Secondly, rule nodes are added to model the causality relationships among fact nodes. For example, if a service $S$ runs on a machine $H$ and an attacker has exploited a vulnerability to *execute arbitrary code on the machine*, then this service $S$ can be impacted by the attacker. This relation can be interpreted as a rule "A compromised machine impacts a service running on it". In other words, when two fact nodes "S runs on machine H" and "attacker executes arbitrary code on the machine" are both present, this rule node will take effect and the derived fact node "S is impacted" will become present. In this example, machine $H$ has a vulnerability. The attack graph generated by MulVAL can only tell "attacker executes arbitrary code on the machine," but it is not able to tell "S is impacted". Therefore, interconnecting the attack graph and the business process dependency graph can help infer the impact of attacks on business process.

Thirdly, the dependency relationships among entities in the business processes become rule nodes. There are three dependency relationships in the business process dependency graph: Or-depends, And-depends and Flow-depends. Listing 1.1 shows a set of example interaction rules crafted to depict the impact propagation among tasks when different types of dependency relationships exist among these tasks. That is, if a task *and-depends* on task 1 and task 2, then this task is impacted by the attacker when either of the two tasks are impacted. if a task *or-depends* on task 1 and task 2, then this task is impacted only when both tasks are impacted. if a task *flow-depends* on task 1 and task 2, then this task will be impacted when task 2 is impacted. In this case, task 2 can be completed only after task 1 is completed. So if task 1 is impacted,

then task 2 is impacted. We will explain more about the dependency relationships in section 5.1.

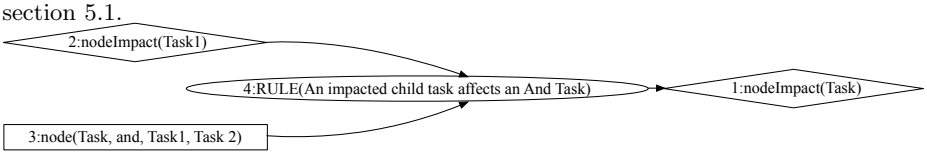



**Fig. 2.** And-dependency in the graph

With all the fact nodes and rule nodes set up, MulVAL can be used to generate the interconnected graph. For example, Fig. 2 shows the first and-depends example in Listing 1.1. In the interconnected graph, different nodes are represented by different shapes, i.e., box, ellipse and diamond. The ellipse shape represents rule node, which is applied only if all needed precondition fact nodes are present. Hence, the ellipse shape represents AND-relation for all precondition fact nodes. The diamond shape represents derived fact node, which is generated as long as one deriving rule node is present. Therefore, the diamond shape represents OR-relation between the deriving rule nodes. In other words, the interconnected graph reflects the relationship between vulnerabilities and the business processes. However, the interconnected graph is too complicated for generating the impact assessment score for a business process. To enable computation of the impact score, we prune the graph to reduce the complexity.

### 3.2 Prune Raw Interconnected Graph

Impact score is a function result of CVSS scores of the vulnerabilities involved in the interconnected graph. When we prune the graph, we must preserve the vulnerability node and the impacted business process node. We apply all the five rules below to prune the graph. The entire process of pruning may take several rounds by applying different rules in each round. In addition, based on different circumstances, we also deal with the edges connecting to the reduced nodes correspondingly.

*Prune all the non-vulnerability leaf nodes.* In this interconnected graph generated by MulVAL, derivation nodes (rule nodes) imply AND relations and derived fact nodes imply OR relations. The primitive fact node in this graph represents the facts in this network, such as the vulnerabilities and deployment configuration. They are represented as leaf nodes in the graph with a shape of box. These non-vulnerability leaf nodes do not participate in the function of CVSS scores. So if a node is not a vulnerability node and is not an AND or OR node, we can prune it. Then each edge derived from these nodes can also be pruned.

*Prune the nodes that have only one ancestor node.* If a node has only one ancestor node, no matter how many child nodes it has, it does nothing but directly deliver impact from its ancestor node to its child nodes. This node is an intermediate impact deliverer for its ancestor node and can be directly pruned without information loss. This kind of nodes is usually the derivation nodes which have only one ancestral vulnerability node, or derived fact nodes which have only one rule to be generated. By pruning one node, the edges from the ancestor node to this node and from this node to the child nodes are removed. A new edge is added directly between the ancestor node and the child node. This operation of pruning one-ancestor nodes may be done several times in the graph-pruning process, as more of them may be produced in other rounds of pruning.

*Prune the nodes, except the vulnerability nodes, which have no ancestors.* Because all left nodes are relation nodes, vulnerability nodes, and the impacted business process nodes. If a node has no ancestor node and is not a vulnerability node, it is a relation node and does not contain any valuable information. This kind of nodes are produced by pruning their ancestor nodes that are usually non-vulnerability nodes. As their ancestor nodes have been pruned, no impact information is delivered to them. Therefore, they can be pruned without impact information loss. The edges from these nodes can also be pruned.

*Find the shortest path from one vulnerability node to the target impacted business process node and merge these paths.* The impact assessment for an attack is to find the relationship between vulnerabilities and the target impacted business processes. If a vulnerability can be exploited in an easy way to affect a business process, there is no need to make it more complex. The assumption in our paper is that attackers always choose the easiest way to achieve the attack goal. Based on this assumption, if there are different paths between a vulnerability node and the impacted business process node in the interconnected graph, the shortest path that has least nodes should be chosen. As a result, each vulnerability node has a shortest path to the target business process node. All other nodes and edges that are not on these paths should be pruned. In some cases, one vulnerability node may have more than one shortest paths to the target business process node. In this case, these paths should also be preserved. To simplify these circumstances, if there are two or more equal shortest paths between one vulnerability node and the impacted business process node, we convert this interconnected graph to two or more interconnected graphs to ensure there is only one shortest path for a vulnerability in one interconnected graph. Finally we calculate each graph's impact score to get the average score.

*Leave only one edge for linked nodes and prune the other edges between them.* In some cases, there are more than one edges between two nodes. The extra edges could be produced by the previous rounds of pruning. They are not needed and thus should be removed too.

These five ways are applied sequentially to the raw interconnected graph generated by MulVAL until the graph does not change again. Two or more graphs could possible be generated as one vulnerability may have two or more equal shortest paths to a target business process node.

### 3.3 Calculate Impact Score

Step 2 can prune the raw interconnected graph to the simplified graph which contains only the vulnerability nodes, the target business process node and their relations. The impact score of the vulnerability node and the target business process node can be represented by $V$ and $M$ respectively. The impact score calculations based on AND-relations and OR-relations are called AND-calculation and OR-calculation. We take the following steps to generate the impact score.

First, we value $V$ by a number between 0 and 1, i.e.,

$$V_i = \frac{CVSS_i}{10}. \tag{1}$$

Second, we define AND-calculation as:

$$V_i \ \ AND \ \ V_j = V_i \times V_j. \tag{2}$$

and OR-calculation as:

$$V_i \ \ OR \ \ V_j = V_i + V_j - V_i \times V_j. \tag{3}$$

Finally, $M$ can be easily calculated by above mentioned calculation methods. For example,

$$M = FUNC(V_1,\ V_2,\ V_3) = (V_1 \ \ OR \ \ V_2) \ \ AND \ \ V_3 = (V_1 + V_2 - V_1 \times V_2) \ \times \ V_3 \tag{4}$$

In this paper, we use the above definitions of AND-calculation and OR-calculation to compute the impact score. However, the administrators of an enterprise network can change the definitions of AND-calculation and OR-calculation based upon different situations and scenarios.

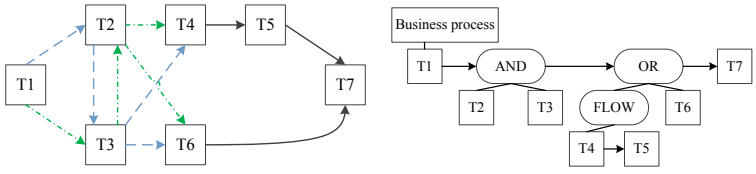
The results of AND-calculation and OR-calculation are directly influenced by the CVSS score of the vulnerabilities. Higher CVSS score usually leads to higher impact score towards the business process, which implies more impact the attack can bring to the business process.

## 4 Case Description

To demonstrate the method for attack impact assessment, we describe a concrete case in this section. We will illustrate the application of our method to this case in section 5.1.

*Business Process Scenario.* This case is a travel reservation system supporting a business process of "providing customers with a web interface for reserving tickets and hotel". This business process consists of seven tasks: $T_1$: Search travel information; $T_2$: Reserve tickets and hotel options; $T_3$: Prompt for signing in or signing up; $T_4$: If signed in, load preference and promotion code; $T_5$: If signed in, reserve a hotel and tickets as a member; $T_6$: If not signed in, reserve a hotel and tickets as a guest; $T_7$: Prompt for payment and confirm the reservation.

From $T_1$ (start of the business process) to $T_7$ (end of the business process), the business process may be executed through four different workflows (i.e. execution paths) as shown in Fig. 3a : $P_1$: $T_1T_2T_3T_4T_5T_7$; $P_2$: $T_1T_3T_2T_4T_5T_7$; $P_3$: $T_1T_2T_3T_6T_7$; and $P_4$: $T_1T_3T_2T_6T_7$. The difference between $P_1$ and $P_2$ and between $P_3$ and $P_4$ is the order of $T_2$ and $T_3$. The customer can either first make reservations ($T_2$) and then be prompted to sign in ($T_3$), or first sign in and then make reservations. If the customer chooses not to sign in during $T_3$, she is recognized as a guest. The difference between $P_1$ and $P_3$ and between $P_2$ and $P_4$ is whether the customer has signed in. If signed in, the system loads customer preference and promotion code ($T_4$) for reserving a hotel ($T_5$). Since $T_5$ depends on the information obtained from $T_4$, $T_5$ should come after $T_4$.




(a) Execution paths (b) Dependency tree

**Fig. 3.** Inter-task dependency

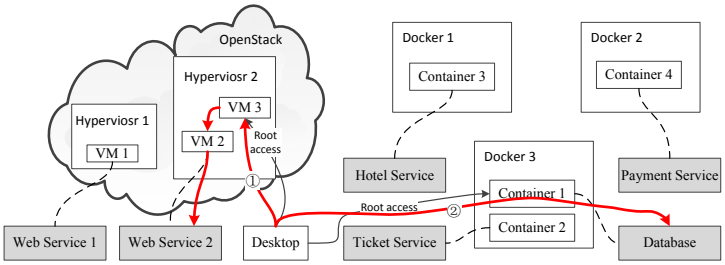



**Fig. 4.** Software Architecture

This travel reservation system can be viewed as a complicated business-process-support enterprise network shown in Fig. 4. The services provided by the network are hosted on different hosts. VM 1, VM 2 and VM 3 are three virtual machines. Web service 1 is hosted in VM 1 which runs in Hypervisor 1. Web service 2 is hosted in VM 2 which runs in Hypervisor 2. VM 3 also runs in Hypervisor 2.

Database service runs in Container 1 which is hosted by Docker 3. Ticket service, which processes ticket-related business, runs in Container 2 which is also hosted by Docker 3. Hotel service, which processes hotel-related business, runs in Container 3 which is hosted by Docker 1. Payment service, which is responsible for monetary transaction, runs in Container 4 hosted by Docker 2. These dockers run in different workstations. A developer's desktop can access the VM 3 and has a root account credential. It can also access Container 1 as a root user. This desktop has a dashboard which displays through HTTP protocol, i.e. it runs a web service. It can also be accessed through SSH protocol from Internet.

**Table 1.** Vulnerability Information

| Vulnerability | CVSS Score | Exploited Result |
|---|---|---|
| CVE-2016-0777 | 6.5 | Privilege Escalation |
| CVE-2016-7479 | 9.8 | Privilege Escalation |
| CVE-2016-6325 | 7.8 | Privilege Escalation |
| CVE-2014-3499 | 7.2 | Container Escape |
| CVE-2016-6258 | 8.8 | Virtual Machine Escape |

*Attack Scenario.* We assume this network has five vulnerabilities and their related information is displayed in table 1. CVE-2016-0777, CVE-2016-7479 and CVE-2016-6325 locate in the developer's desktop and allow attackers to escalate privilege. CVE-2014-3499 locates in the docker software and can enable an attacker to escape from the container. CVE-2016-6258 locates in the Kernel-based Virtual Machine(KVM) software and can also be used to break the virtual machine.

There are two attack paths in Fig. 4. One attack path is denoted as red line 1 in Fig. 4. The attacker firstly exploits the vulnerability in the web application or the SSH application to compromise the developer desktop, which has the log-in credential for VM 3. By leveraging the vulnerability in the KVM software, the attacker can directly access the host, i.e. Hypervisor 2, by breaking the isolation between the virtual machine and the host. The attacker can then access VM2 which hosts Web service 2 and execute arbitrary code on this virtual machine. Once Web service 2 is compromised, all tasks depend on this service are impacted. The other attack path is denoted as red line 2

in the Fig. 4. As the developer's desktop has the log-in credential for Container 1, the attacker can also access this container. With the database running in this container, the attacker can execute arbitrary code in the database process and then affect all tasks depending on the database.

# 5 Case Study and Evaluation

## 5.1 Case Study Results

In this section, we applied the impact assessment method to the case described above and demonstrate the experiment results.

First, we obtained the CVSS scores for the five vulnerabilities in this case according to their CVE IDs.

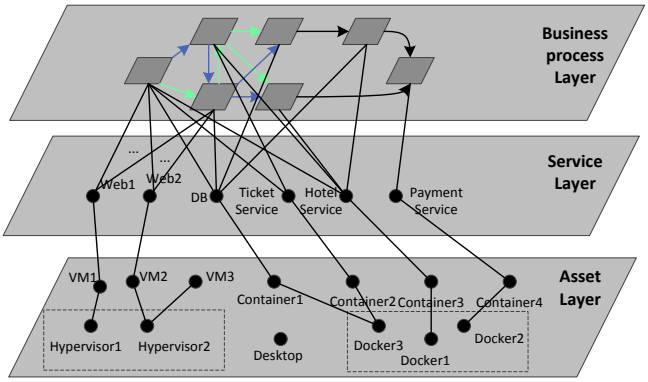



**Fig. 5.** The Entity Dependency Graph

Second, we constructed a entity dependency graph for this network, as shown in Fig. 5 (as web services are depended on by each task, some edges from the tasks to Web1 and Web2 are ignored in this figure). The entity dependency graph contains three layers: asset layer, service layer and business process task layer. Among these tasks, $T_1$ and-depends on the web services, database service, ticket service and hotel service. $T_2$ and-depends on web services, ticket service and hotel service. $T_3$ and-depends on web services, and database service. $T_4$ and-depends on web services, and database service. $T_5$ and-depends on web services, database service, and hotel service. $T_6$ and-depends on web services, and hotel service. $T_7$ and-depends on web services, and payment service.

At the business process layer, we specified the dependency relationships among tasks. To better understand the relationships, we firstly define three special tasks: $T_{or}$, $T_{and}$ and $T_{flow}$. As the name implies, these tasks represent three relationships: Or-dependency, And-dependency, and Flow-dependency. That is, if a task $T_{or}$ or-depends on sub-tasks $T_i$ and $T_j$, then $T_{or}$ is impacted only when $T_i$ and $T_j$ both are impacted. If a task $T_{and}$ and-depends on sub-tasks $T_i$ and $T_j$, then $T_{and}$ is impacted when $T_i$ or $T_j$ is impacted. If a task $T_{flow}$ flow-depends on sub-tasks $T_i$ and then $T_j$, then $T_{flow}$ is impacted when $T_j$ is impacted. In addition, the impact on $T_i$ will cause an impact on $T_j$, which leads to an impact on $T_{flow}$. The relationships of the seven tasks of this business process can be depicted in Fig. 3b. In other words, this business process viewed as one $T_{flow}$ flow-depends on $T_1$, $T_{and}$, $T_{or}$ and then $T_7$. $T_{and}$ and-depends on $T_2$ and $T_3$. $T_{or}$ or-depends on $T_6$ and $T_{flow}$, which flow-depends on $T_4$ and then $T_5$.

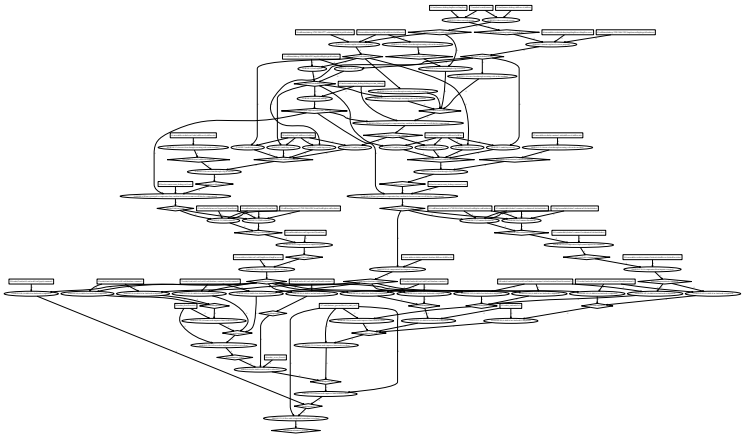


**Fig. 6.** Interconnected Graph

After instantiating the knowledge units, we can get the interconnected graph as shown in Fig. 6. In this graph, the ellipse represents AND-calculation and the diamond represents OR-calculation. By applying the five pruning rules described in section 3.2 against the raw graph, we generated the pruned graph, as shown in Fig. 7, to show the relationship between vulnerabilities and the target business process. The expression "nodeImpact(X)"means "X" is impacted, e.g. "nodeImpact(business process)" means the target business process is impacted. The CVSS scores of these vulnerabilities are shown in table 1. Therefore, the final impact score of this attack can be calculated as:

$$\begin{aligned} M &= (((V_{CVE-2016-0777} \;\; OR \;\; V_{CVE-2016-7479}) \;\; AND \;\; V_{CVE-2016-6325}) \\ &\quad AND \;\; V_{CVE-2016-6258}) \;\; OR \;\; V_{CVE-2016-3499} \\ &= 0.91. \end{aligned}$$

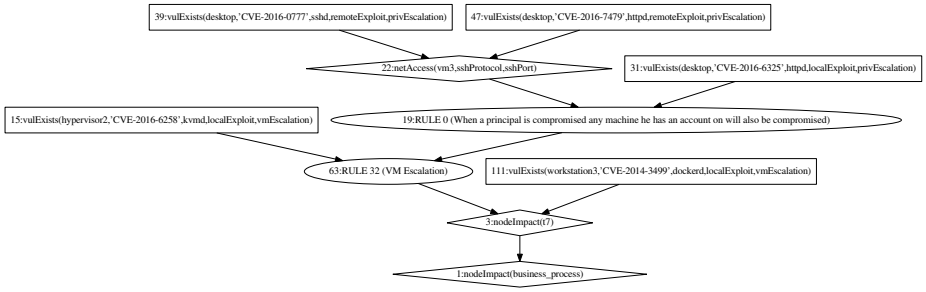



**Fig. 7.** Pruned Interconnected Graph

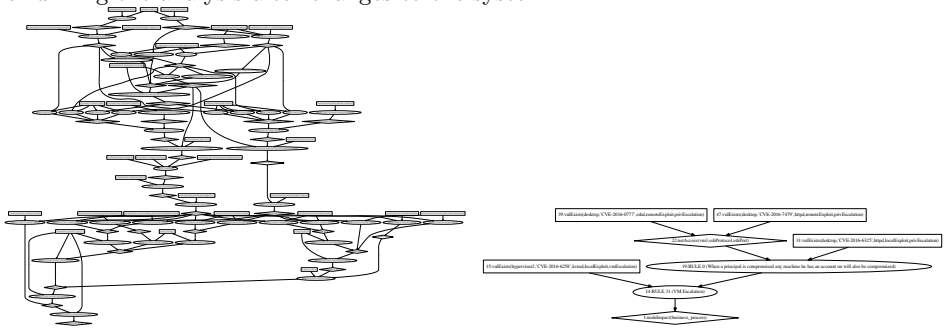
Apart from the impact score calculated from the pruned interconnected graph, there is more information about whether the services and tasks are impacted or not from the raw interconnected graph. By searching through the raw interconnected graph showed in Fig. 6, we can get that all tasks are impacted by this attack. Three services including Web service 2, Ticket service and Database service are also impacted. All tasks are impacted as they all and-depend on Web service 2. These three services are

impacted as they can be accessed by the developer's desktop which can be controlled by the attacker. That is, the impact on these services match the attack path described in Section 4. Moreover, we can also get the impact score for each task through the same process: pruning the graph and calculating the score based on the AND-calculation and OR-calculation. The impact score for each task is 0.992 for task 1, 0.91 for task 2, 0.973 for task 3, 0.973 for task 4, 0.973 for task 5, 0.682 for task 6, and 0.91 for task 7. We can see some scores are higher than the impact score for the whole business process. This is because some task are easily attacked by the attacker from the Internet. For example, task 3 and-depends on web service 1, web service 2 and database service. The attacker can impact task 3 without exploiting the vulnerability "CVE-2014-3499", which lowers the requirement for the attacker.

There are three services that are not impacted by the attack, including Web service 1, Hotel service and Payment service. They cannot be found as the impacted nodes in the raw interconnected graph. This is because they are not involved in the attack path. Therefore, the raw interconnected graph can precisely present the attack path in the real world.

### 5.2 Analysis of Different Cases

Section 5.1 has shown a successful application of our impact assessment method to the case described in section 4. However, in the real world, the enterprise network is not static. For example, a vulnerability can be patched or a host can be removed. In this section, we will show that our method can still handle the dynamic changes in the enterprise network and generate new impact scores for the business processes by re-running the analysis after changes to the system.



(a) The raw interconnected graph (b) The pruned interconnected graph

**Fig. 8.** Interconnected graphs with a vulnerability is patched

*A vulnerability is patched.* When a vulnerability is patched, it means a fact node should be deleted. As a consequence, the interconnected graph will be different and so is the pruned graph. For instance, we assume the vulnerability "CVE-2014-3499" is patched as this vulnerability is the oldest one in these five vulnerabilities. Fig. 8 shows the new raw interconnected graph and pruned graph without "CVE-2014-3499." By analyzing this pruned graph, the new impact score towards the business process is 0.682, which is much smaller than 0.91.

Whether a task or a service is impacted can also be acquired through the raw interconnected graph. By searching this graph, we can see all tasks are still impacted.

Two services including Web service 2 and Database service are impacted. The other four services, including Web service 1, Hotel service, Payment service and Ticket service, are not impacted. Compared with section 5.1, ticket service is not impacted in this case. This is because patching the vulnerability "CVE-2014-3499" prevents the escape from Container 1. The attacker cannot access Container 2 any more so that the ticket service running in Container 2 is free from the impact.

*The developer desktop is removed.* When the developer desktop is removed, several fact nodes should be deleted. For example, three vulnerabilities in this desktop no longer impact the network, so these vulnerability nodes are deleted. When generating the interconnected graph with MulVAL, we found no graph was generated. This means although there are vulnerabilities in this network, the attacker located in the Internet cannot impact this business process. The reason is that all attack paths start from this desktop as the entry point. Removing this desktop prevents the attacker from exploiting the vulnerabilities inside the network. Therefore, the interconnected graph can precisely reflect the real-world impact circumstances.

### 5.3 Evaluation of Scalability

**Table 2.** Time consumed to generate interconnected graphs according to different Number of Units (NoU) and different Connectivity Level (CL)

| CL \ NoU | 100 | 200 | 400 | 600 |
|---|---|---|---|---|
| 5 | 1m2.45s | 7m44.71s | 67m55.64s | 228m42.53s |
| 10 | 1m0.33s | 7m49.49s | 65m4.48s | 253m9s |
| 100 | 0m59.67s | 7m48.85s | 65m18.60s | 224m33.49s |

Section 5.1 illustrates how to leverage our impact assessment method to calculate the impact score for an attack targeting a particular business process. The key idea is to extend MulVAL to generate an interconnected graph and calculate the impact score based on the pruned graph. In this process, generating the interconnected graph is the most time-consuming part. It directly affects the scalability of our impact assessment method. Therefore, in this section, we evaluate the scalability of our method in terms of how fast interconnected graphs can be generated for different scopes of network.

In order to get different scopes of network, we view the small network of the aforementioned case in section 4 as one unit and duplicate it. These units are then combined on the basis of different connectivity levels. Because different connectivity levels differ the network complexity, which may affect the time used to generate the interconnected graph. We define connectivity level as how widely one web server is shared, i.e., how many units share one web server. These units sharing one web server constitute one group and each group is connected by the database server of one unit in the group. Therefore, the scope of a network generated through this method can be measured by number of units and connectivity level.

Table 2 describes the time consumed to generate interconnected graphs for different scopes of network according to different number of units and different connectivity level. The first column indicates connectivity level and the first row presents the total number of duplicated units. The other grids in the table indicate how much time is used to generate one graph. For example, with 100 duplicated units in the network and every 5 units sharing one web server, generating the interconnected graph for this scope of network consumes 1 minute and 2.45 seconds.

From table 2, we can see the time used to generate an interconnected graph is mainly determined by the number of connected units, not the connectivity level. This is because when generating the interconnected graph, the time is mainly consumed by finding new path from one node to another node. As sharing web server does not increase paths in the graph, the consumed time does not affected by the connectivity level. Furthermore, the time increases non-linearly, i.e., the time increases faster than the number of connected units increases. In summary, our method cannot scale well in a very large network. However, it does not mean our solution is not practical in the real world. Taking a university as an example, the scope of one unit is similar to a network of a department. Therefore, for a big university with 100 departments, the time consumed to generate an interconnected graph is less than 2 minutes, which means our solution is feasible in practice.

## 6 Related Work

Little research has been done on business process impact assessment in recent years. Jakobson [8] presents a business process impact assessment that quantifies impact by using Operational Capacity (OC), and considers intra and inter dependencies between assets, services, and business processes. Dai et al. [3] propose a cross-layer Situation Knowledge Reference Model (SKRM) which considers intra and inter-dependencies between instruction layer, OS layer, app/service layer, and workflow (task) layer. Sun et al. [18] introduce a novel probabilistic impact assessment method which leverages Bayesian networks. Sun et al. [20] also propose a multi-layer impact evaluation model which includes four layers, namely vulnerability layer, asset layer, service layer, and mission layer. They measure impacts by OC and impact factor. Poolsappasit et al. [13] leverages attack graph (called Bayesian Attack Graph) and attack tree to revise the likelihoods in the event of attack incidents and identify the vulnerable points in the network system. Frigault et al. [5] use attack graph as a special Bayesian network to model probabilistic risks in a network. They also introduce Dynamic Bayesian Networks [6] with attack graphs to model the security of dynamically changing networks. Dewri et al. [4] leverage an attack tree model with multi-objective optimization to solve the problem, i.e. balance between security hardening and limited budget for an enterprise network. Ray et al. [15] also utilize an attack tree model with an algorithm simplifying the tree to locate the malicious insiders in a network. Saripalli et al. [16] present QUIRC which utilizes Microsoft's STRIDE to assess the security risk in a cloud computing environment and define risk as a combination of the Probability of a security thread event and its severity.

Our method uses the interconnected graph, which interconnects attack graph and entity dependency graph, to demonstrate the relationships between vulnerabilities and the impacted business process. By pruning the interconnected graph, we can get simplest relationships and calculate the impact score based on vulnerabilities' CVSS score. For different cases in one network, our method can handle these changes and generate related impact scores. With these impact scores, the network operator may do further security hardening for the network.

## 7 Conclusion

In this paper, we propose a new business process impact assessment method, which measures the impact of an attack towards a business process in an enterprise network.

Our method produces a numerical score for the attack impact. We extend MulVAL, a logic-based network security analyzer, to support more fact nodes and rule nodes for business process impact assessment. With the facts and rules, our approach generates an interconnected graph for an attack and prunes the interconnected graph to show the simplified relation between vulnerabilities and business processes. In the end, the impact score can be calculated by analyzing the pruned graph and following the relation calculation rules. According to our case study, this business process impact assessment method is effective and can facilitate the cyber-defense and cyber-resilience in an enterprise network that supports business processes.

## Acknowledgment

We thank the anonymous reviewers for their valuable comments. This work was supported by NIST 60NANB17D279, NSF CNS-1505664, ARO W911NF-13-1-0421 (MURI), and NSF CNS-1618684.

## Disclaimer



## References

1. Ammann, P., Wijesekera, D., Kaushik, S.: Scalable, graph-based network vulnerability analysis. In: Proceedings of the 9th ACM Conference on Computer and Communications Security. pp. 217–224. ACM (2002)
2. Chen, X., Zhang, M., Mao, Z.M., Bahl, P.: Automating network application dependency discovery: Experiences, limitations, and new solutions. In: OSDI. vol. 8, pp. 117–130 (2008)
3. Dai, J., Sun, X., Liu, P., Giacobe, N.: Gaining big picture awareness through an interconnected cross-layer situation knowledge reference model. In: Cyber Security (CyberSecurity), 2012 International Conference on. pp. 83–92. IEEE (2012)
4. Dewri, R., Poolsappasit, N., Ray, I., Whitley, D.: Optimal security hardening using multi-objective optimization on attack tree models of networks. In: Proceedings of the 14th ACM conference on Computer and communications security. pp. 204–213. ACM (2007)
5. Frigault, M., Wang, L.: Measuring network security using bayesian network-based attack graphs. In: Proceedings of the 2008 32nd Annual IEEE International Computer Software and Applications Conference. pp. 698–703. IEEE Computer Society (2008)
6. Frigault, M., Wang, L., Singhal, A., Jajodia, S.: Measuring network security using dynamic bayesian network. In: Proceedings of the 4th ACM workshop on Quality of protection. pp. 23–30. ACM (2008)

7. Jajodia, S., Noel, S., OBerry, B.: Topological analysis of network attack vulnerability. In: Managing Cyber Threats, pp. 247–266. Springer (2005)
8. Jakobson, G.: Mission cyber security situation assessment using impact dependency graphs. In: Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on. pp. 1–8. IEEE (2011)
9. NIST: Cvss score. https://nvd.nist.gov/vuln-metrics/cvss (2017)
10. Noel, S., Jajodia, S., O'Berry, B., Jacobs, M.: Efficient minimum-cost network hardening via exploit dependency graphs. In: Computer security applications conference, 2003. proceedings. 19th annual. pp. 86–95. IEEE (2003)
11. Ou, X., Boyer, W.F., McQueen, M.A.: A scalable approach to attack graph generation. In: Proceedings of the 13th ACM conference on Computer and communications security. pp. 336–345. ACM (2006)
12. Phillips, C., Swiler, L.P.: A graph-based system for network-vulnerability analysis. In: Proceedings of the 1998 workshop on New security paradigms. pp. 71–79. ACM (1998)
13. Poolsappasit, N., Dewri, R., Ray, I.: Dynamic security risk management using bayesian attack graphs. IEEE Transactions on Dependable and Secure Computing **9**(1), 61–74 (2012)
14. Racket: Datalog. https://docs.racket-lang.org/datalog/ (2017)
15. Ray, I., Poolsapassit, N.: Using attack trees to identify malicious attacks from authorized insiders. In: European Symposium on Research in Computer Security. pp. 231–246. Springer (2005)
16. Saripalli, P., Walters, B.: Quirc: A quantitative impact and risk assessment framework for cloud security. In: Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on. pp. 280–288. Ieee (2010)
17. Sheyner, O., Haines, J., Jha, S., Lippmann, R., Wing, J.M.: Automated generation and analysis of attack graphs. In: Security and privacy, 2002. Proceedings. 2002 IEEE Symposium on. pp. 273–284. IEEE (2002)
18. Sun, X., Singhal, A., Liu, P.: Who touched my mission: Towards probabilistic mission impact assessment. In: Proceedings of the 2015 Workshop on Automated Decision Making for Active Cyber Defense. pp. 21–26. ACM (2015)
19. Sun, X., Singhal, A., Liu, P.: Towards actionable mission impact assessment in the context of cloud computing. In: IFIP Annual Conference on Data and Applications Security and Privacy. pp. 259–274. Springer (2017)
20. Sun, Y., Wu, T.Y., Liu, X., Obaidat, M.S.: Multilayered impact evaluation model for attacking missions. IEEE Systems Journal **10**(4), 1304–1315 (2016)