A Graphical Modelling of Threat Assessment of Enterprise Business Application

Submitted in fulfillment of the requirements
of the degree of
Master of Technology

by
Major Manjunath Bilur
173054001

Supervisor :

Prof. R. K. Shyamasundar

COMPUTER SCIENCE ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY BOMBAY

June 2019

Acknowledgements

I would like to express my sincere thanks of gratitude to my guide **Prof. R.K. Shyamasundar, Department of Computer Science and Engineering, IIT Bombay** for his invaluable guidance and continuous encouragement and motivation throughout the MTP Phase I. A special thanks to my seniors for continuous support and help. Without guidance and suggestion of these people, this project would not have been possible. I am truly grateful to my parents and god for the blessings. At last but not the least, I am thankful to ISRDC lab-mates, roommate, and friends for being with me.

Major Manjunath Bilur

Abstract

Cyber resiliency has been a very challenging engineering research area as it's aim is to assure that enterprise network has the capability to continuously provide essential functions for the underlying support of mission and business processes in the midst of an attack. There have been several innovative case studies done earlier to assess the cyber resiliency of an enterprise business system through the development of attack graphs with the underlying mission dependency and service-dependency graph etc. For either automating or say semi-automating (more realistic!) such an approach, the challenge is to extract distinct layers like Asset layer, Service layer and Business process task layer etc. Such an information extraction will lead to realize possible attack trees for threat assessment taking into account the underlying vulnerabilities of the system in the architecture of the business system along with threat perception. In this paper, we extract such information from the underlying Business Process Modeling Notation (BPMN) description of the business application. First, from the BPMN graphical representation, we obtain the task dependency graph. Next, with assistance either from the administrator or from system architecture documents, we obtain the hierarchical dependency graph consisting of the asset layer, service layer and business process task layer. From the above graphical dependency graph, we generate an input file for the MulVAL [1], after creating rules of interactions of the entities in the architecture, so that plausible attack paths can be generated. The attack graph generated from MulVAL is imported into a graphical DB to accomplish real time analysis and possible update of the graph that may be required as per the operational scenarios of the business application. From the graphical output obtained, we generate the rules for risk assessment. The analysis of the model is done through the graph DB, Neo4j, that enables us to arrive at a real time assessment that is both scalable and re-configurable in real time. We illustrate the application of our approach to enterprise systems and the power of graphical modeling that allows us to view, analyze and arrive at threat assessments of the business applications.

[1]MulVAL : `http://people.cs.ksu.edu/xou/argus/software/mulval`

Contents

List of Figures

1. Introduction

Threat is an unavoidable evil which persists and grows exponentially with increase in easy reach of computing base for the common man. It is augmented further with new digital assets capturing centre stage in all walks of life covering the complete canvass of all material and non material domain. The security of the new age network in which information flows at wire speed is an important task for service providers and administrators for constant conflict against adversaries. The traditional paradigm of linear and reactive approach is not competent enough for today's and future networks. The proactive security as an important parameter from the inception is the need of the day. Constant vigil of network against all kinds of attacks-in-progress and those which can be anticipated including zero day attacks is in demand.

The technology has penetrated in every activity and is only dominating its footprint with time. All simple to complex day to day problems or problems using some high end technology have transcended to e-domain. Being connected to information is not a luxury but quintessential requirement in modern world. Due deliberations to address the security issues with respect to particular software, vulnerability, and application in isolation is in vogue since many years. Today an attacker can infiltrate a well guarded network through multistep, multistage attack. The attacker can use numerous vulnerabilities present at various stages as a stepping stone to next level. A hierarchical and proactive threat mitigation and vigilance is indispensable.

Several approaches have been suggested in this regard. Logic trees, Cyber threat trees, fault trees to attack trees have been envisaged in great detail as suitable models for analyzing threats. Many papers suggest using metrics to quantify the attack path or the resource required for attacker. Suggested ones are probability based, CVSS metric based or dependency based. The diverge techniques to understand and classify high impact paths have been suggested. Among all, attack tree seems to stand out as best bet for modern multistep, multistage vulnerability systems. Attack trees to model and analyze the threat against digital platforms like computer systems were defined by Bruce Schneier [1][2]. The name attack trees was first mentioned by Salter et al in 1998[3]. Weiss in 1991[4] and Amoroso in 1994[5] proposed tree based approach to analyze security of any system. Attack trees cover a complete canvas of all known tree types like Cyber threat trees[6], fault trees[7], logic trees for threat and attack specification language[8] which are AND-OR tree structures in graphical modelling.

The root of the tree is the main security threat which the attacker would initially try to attain. From root, the tree is branched out to child nodes which depict sub attacks. The branching is carried out till it cannot further break down the attack into sub attacks. The branching may be an AND or an OR gate which has its own significance. These last level child nodes which cannot be broken down are called leaf(Fact) nodes. Keinzle and Wulf were the one who covered the complete framework of attack tree[9]. Further many authors highlighted various different templates and patterns of generating and applying the attack tree concept.

Many Attack Tree software application are available both free and commercial version. Example of one such software is MulVAL (Multihost Multistage Vulnerability Analysis). This tool generates an attack tree based on input which has to be manually fed. The input for the tool is based on Network design, System configurations, Vulnerabilities, Tasks in business process and any other Firewall or Access rules with respect to network. But these have only basic assumptions in place for a network. Every network design varies based on application, hardware and other parameters. The recent work [10] provides an excellent case study for risk assessment through interconnected attack graphs and entity dependency graph. This work is specific to a case study and a model. Our aim is to provide a computational platform, where such risk assessment can be visualized and articulated with the needed user/administrator interactions.

The rest of the paper is organized as follows. Section 2 discusses the background followed by a running example in Section 3, that is used for our gentle narration in Section 4. Section 4 informally illustrates the whole approach. Section 5 provides the broad formal steps of the threat platform being built for enterprise applications and highlights the implementation. This is followed by conclusions in Section 6.

2. Background

In this section, we provide a brief overview of business enterprise systems and the BPMN notation used for formalizing business enterprise systems along with a brief on graph DB.

2.1 Business Model and Its Architecture

Any business process is dependent on various sub modules which can be analyzed as sub layers also known as abstraction layers. In the paper, the layers encompass Asset layer, Service layer and Business Process task layer. Asset layer mainly deals with disks and hardware which support this business process. Service layer consists of services like database services, web servers, application servers which run many feature providing services like shopping service, booking service, payment service, RBAC services. At the top, the business process layer would list all tasks that make the complete business process.

The dependency graph can be derived based on overall logic of interdependence among all layers. It is evident that a service at service layer may depend on more functions in addition to those provided by Asset layer. At the business process layer, one task may depend on other task/tasks for completion of the process.

2.2 BPMN Notation

It is a standard graphical representation or notation for any business application by which one can visualize the complete internal working model for any given business. It will clearly portray the understanding of all players, flow of procedures, options available, interactions among different entities and make it easy to comprehend the overall business understanding.

The standard is continuously monitored and revised for efficiency by OMG (Object Management group). All events which are part of a business process are given a standard notation making it universally acceptable. Many online platforms like lucid chart, Camunda, Sparks, MagicDraw, UModel, Visual Paradigm etc to enable us to make the BPMN model for any application. This acts as an common interface among all stakeholders from all different dimensions involved in modelling and executing the overall Business. For further details readers are referred to [11].

2.3 Graph DB - Neo4j

The ease of storage and modelling the business model to assess threat impact in real time with scalability as an important factor is met by the graph DB Neo4j. It depicts connected as well as semi structured data using graphs. It allows us to retrieve any information from model by using Cypher Queries. Cypher Query Language(CQL) is available to do any transactions on the database. No complex joins are required for information retrieval. Different clauses in the Cypher queries are:-

Read Clauses: [MATCH, WHERE, START, LOAD CSV]
Write Clauses: [CREATE, MERGE, SET, DELETE, REMOVE, FOREACH, CREATE, UNIQUE]
General Clauses: [RETURN, ORDER BY, LIMIT, SKIP, WITH, UNWIND, UNION, CALL]

For further details, the reader are referred[12][13]. Other tools used in the platform is given in the Appendix.

3. Introduction of running example structure

For purposes of ease of presentation of issues, we use a running business application that has the following tasks (we have given a brief description of the tasks for the same reasons):

Task *T1*: Search for options (start of the application)

Task *T2*: Selecting an option

Task *T3*: Prompt for signing in or signing up

Task *T4*: If signed in, place order as a member

Task *T5*: If not signed in, place order as a guest

Task *T6*: Payment done via UPI, Net Banking, debit/credit card

Task *T7*: Prompt payment confirmation message

The broad BPMN model is shown in Fig. 4.1

Vulnerability	**CVSS Score**	**Exploited Result**
CVE-2016-9962	6.4	Container Escape
CVE-2016-3697	7.8	Privilege Escalation
CVE-2018-15514	8.8	Privilege Escalation
CVE-2018-2844	8.8	Virtual machine escape
CVE-2016-0777	6.5	Privilege Escalation

TABLE 3.1: Vulnerability Information

3.1 Attack paths for the business model under consideration

For our system payment gateway, let us assume that it has five vulnerabilities. The detailed explanation with respect to its impact score (Base Score) and the outcome is shown in Table 3.1. The vulnerabilities that aid the attacker to gain access are CVE-2016-0777, CVE-2016-3697 and CVE-2018-15514 present in the desktop that has root access to Hypervisor-1 and Workstation-2 and Workstation-3. CVE-2016-9962 is a vulnerability that is part of docker and aids the attacker in ability to move out of container. The vulnerability that allows attacker to open virtual machine is CVE-2018-2844 which is part of VM kernel.

The attacker can target the system by the paths shown in the red. The attacker can exploit both web and ssh vulnerability on the desktop which gives root access to VM1. On accessing VM1, by exploiting the vulnerability in the kernel, he can bypass isolation between VM and Hypervisor-1. The attacker now can access VM2 and can execute any arbitrary code as it hosts web service-2. Once web service is compromised, then any service dependent on them is also affected. As per architecture, the desktop has root access to Container-2 and Container-3. This gives attacker privilege to execute any code in database and reservation service thereby impacting both the services.

4. Gentle informal illustration of our approach of threat analysis of Business Application

In this section, we illustrate our end-to-end approach through various steps that highlight the underlying algorithmic aspects as well as user interactions.
Consider the BPMN representation of the running example discussed above shown in Fig. 4.1.

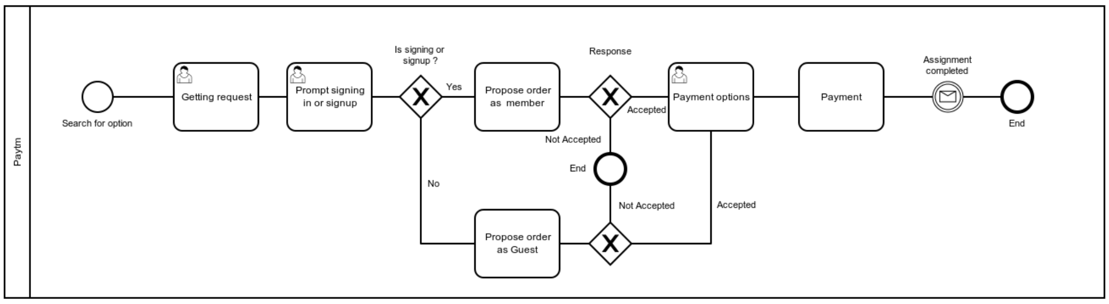

FIGURE 4.1: BPMN Model for Business Process under consideration

Step 1: Derivation of Logical Dependency Graph:

Creating the main task graph from BPMN. From Fig. 4.1, the task graph extracted is illustrated in Fig. 4.3. We transform the task graph into a logical dependency graph as shown in Fig. 4.4. The process of converting from BPMN to dependence tree is depicted in Fig. 4.2. In order to convert the BPMN to Dependence tree a python script is being used. Following are the steps.

- Script.py takes .bpmn file as an argument.
- After providing the appropriate file to the script, the first operation of the script is to extract the tasks from the file which is done by providing appropriate regular expression.
- Once the tasks are being extracted, the next operation of script is to create a map of these tasks with their IDs which will be further used during graph creation.
- Apart from extracting the tasks from the file, the script also needs to extract the sequence flow and message flow between the tasks in order to generate the graph. So the next operation script does is to extract the sequence and message flow from the file. The sequence and message flow contain two crucial information that is the source ID and target ID.
- This source IDs and target IDs are being checked against the map in order to find the corresponding tasks.
- Once the source and target tasks are being found, accordingly a 2D matrix is created which represents a directed graph of the tasks and their dependence.
- This matrix is then further used for creating the Dependence tree.

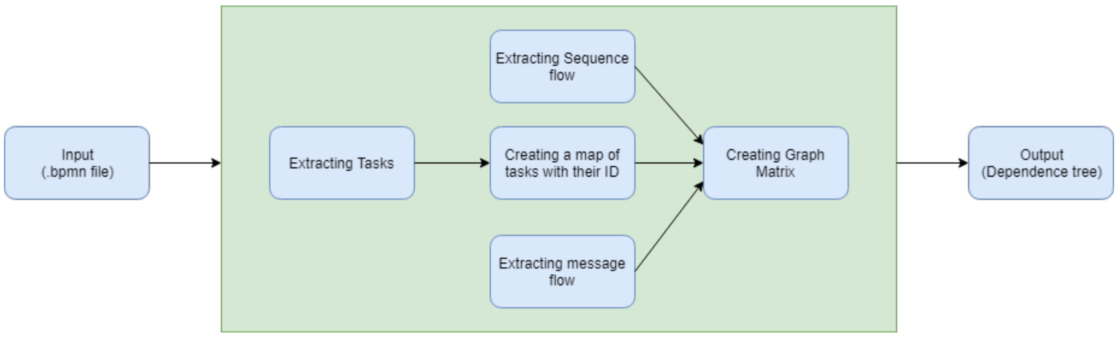

FIGURE 4.2: Process of converting BPMN to Dependence tree

These steps are almost algorithmic/automated. From the task dependency graph, we can infer possible ways in which an insider/outsider can become an attacker. These aspects depend on the threat perceived from the overall security of the physical system as well.

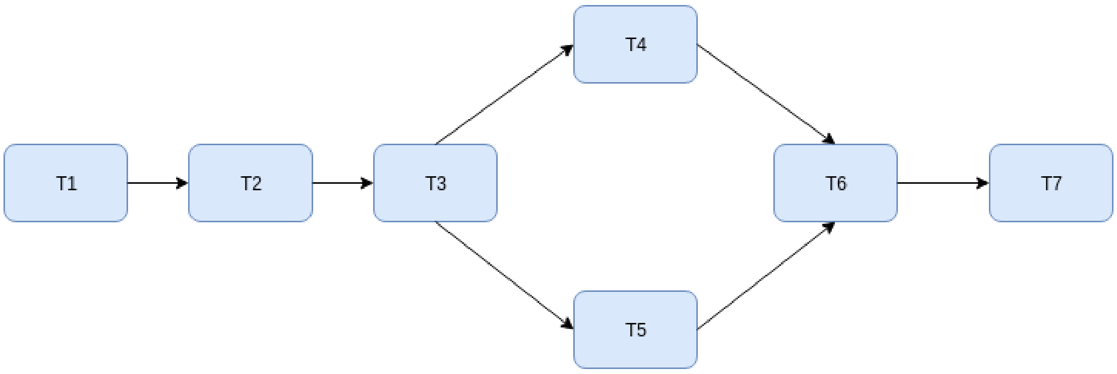

FIGURE 4.3: Task Flows

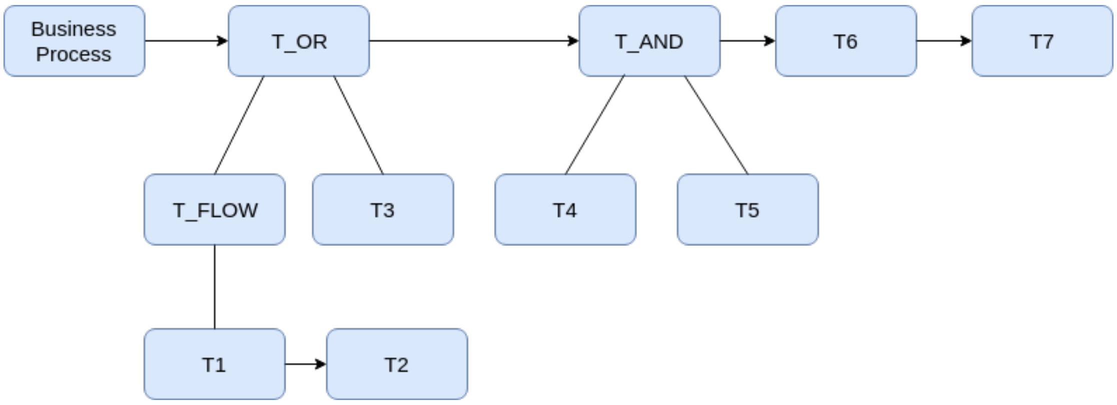

FIGURE 4.4: Logical Task Dependency Graph

For instance, from the flow of tasks, we can see that after the transaction is initiated by T1, it is followed by T2 and T3 in sequence. After this, there are two options with the attacker either to login-if-member else continue as-guest. If logged-in, then the sequence of the path is T4 followed by T6 and then go to T7 to confirm after doing the necessary payment, or if he chooses to do as a guest, then he goes to T5 and T6 in sequence and then to T7 to confirm.

Step 2: Mapping the Task Dependency Graph on the Actual Physical Network:

From the above graphical representation, we arrive at the physical network architecture for each of the tasks with details of hardware/software and their respective vulnerabilities. The mapping is to be done by the actual system designed for the task. Hardware/-software vulnerabilities can be obtained by using global databases in coordination with probing/mapping software, and streamlined into plausible or impossible, based on the access control enforced on the system. The actual physical architecture of the business application under consideration is shown in Fig. 4.5 .

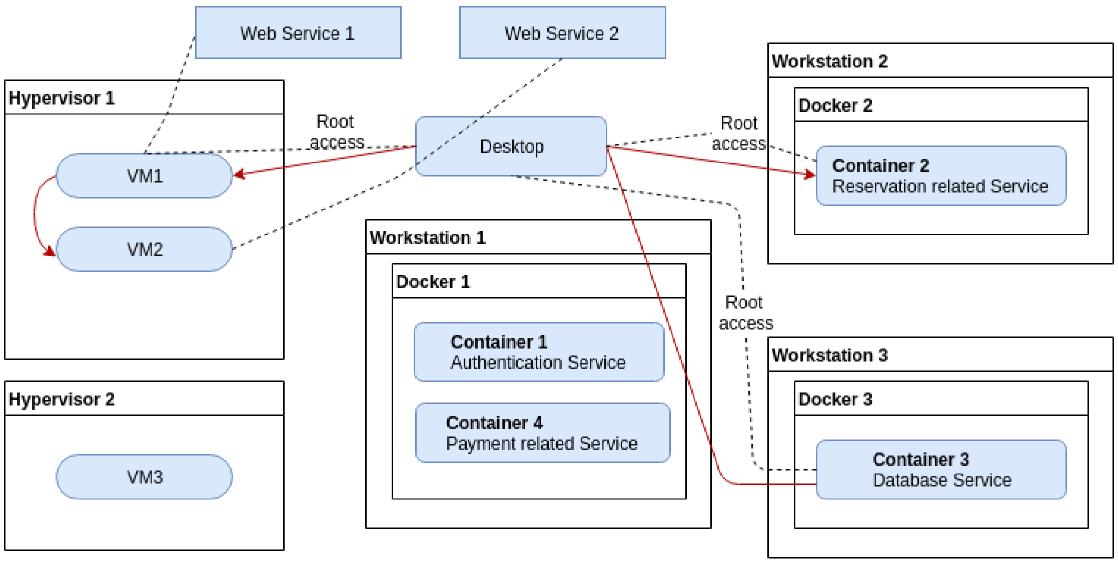

FIGURE 4.5: Software architecture for the model under consideration

Step 3: Layered architecture of BP:

From the logical/physical representation of the business application shown in Fig. 4.4 and Fig. 4.5 respectively, we obtain an integrated representation of the asset layer, service layer and the business process layer for the business application. In our running example, such a hierarchical layer is depicted in Fig. 4.6.

Step 4: Generation of interaction rules for vulnerabilities/attack flow in the system dependency graph:

Having generated the dependencies of hardware/software on the service components in the service layer which in turn impacts the business layer, we need to come up with rules that defines the propagation of vulnerabilities resulting in the attacks at the service layer or the business layer. For instance, let us consider the interaction among T1, web 1 and VM1 shown in Fig. 4.6. Suppose the attacker has reached VM1 (Correspond desktop). Let us assume VM1 has a vulnerability that allows the attacker to execute any code in VM1 with super user privilege. Now, web 1 that is dependent on VM1 suffers and hence T1 gets compromised. This can be captured as a Prolog rule. This illustrated

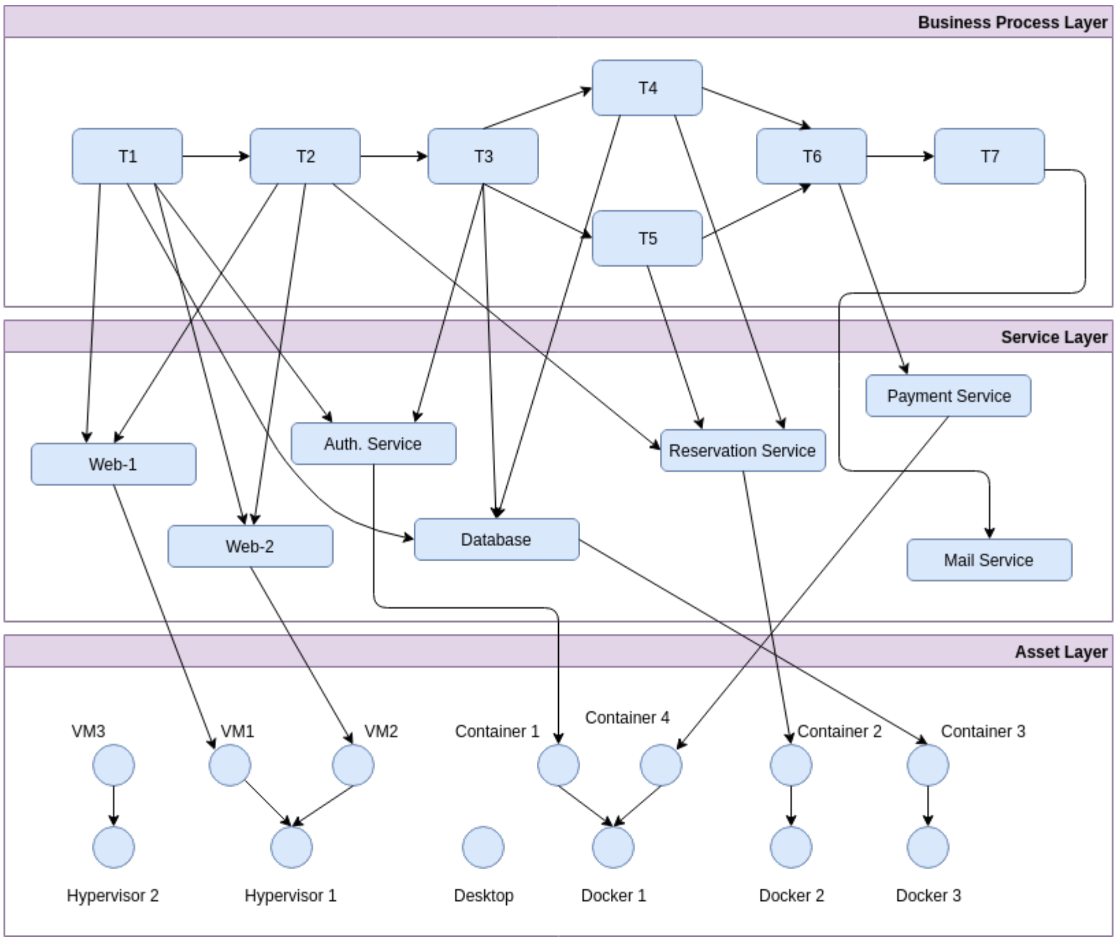

FIGURE 4.6: Dependency Graph among all three layers

in the example given in Step (7) of section 5.

Step 5: Creation of Input of Attack tree Generation:

Create an input to MulVAL consisting of the vulnerabilities in the Fig. 4.6, along with interaction rules from step 4. From such an input, MulVAL generates the attack graph and also generates complete list of nodes and relations of the generated attack graph.

Step 6 : Exporting attack graph to Graph DB:

Capture attack graph in a graph DB so that online/real time evaluation can be done. This is done by transforming the attack graph information into a graph DB that can be simulated/visualized on browser. The attack graph generated is shown in Fig. 4.7.

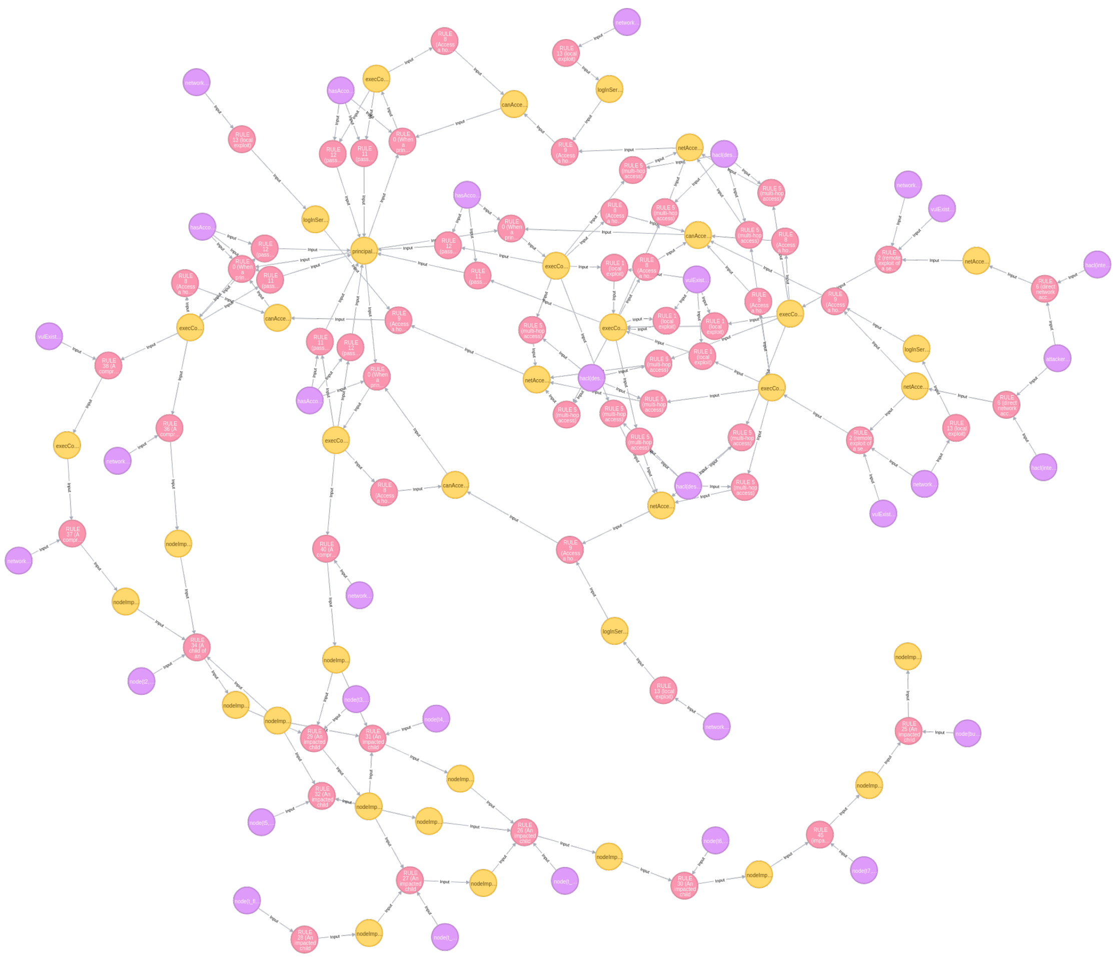

FIGURE 4.7: Attack graph generated by graph DB

Step 7: Threat/Risk Assessment Using Cypher Queries:

Having transformed the attack graph, we need to assess the possible attacks through appropriate queries (or Cypher queries) on graph DB. The shortest path in the attack graph generated is shown in Fig. 4.8. The concept of looking into threat assessment is also well captured in various works such as [14],[15], [16],[17].

Step 8: System Hardening for Threats and Risks:

From the above assessment one can articulate various solutions like Network Hardening or Crypto Measures etc.

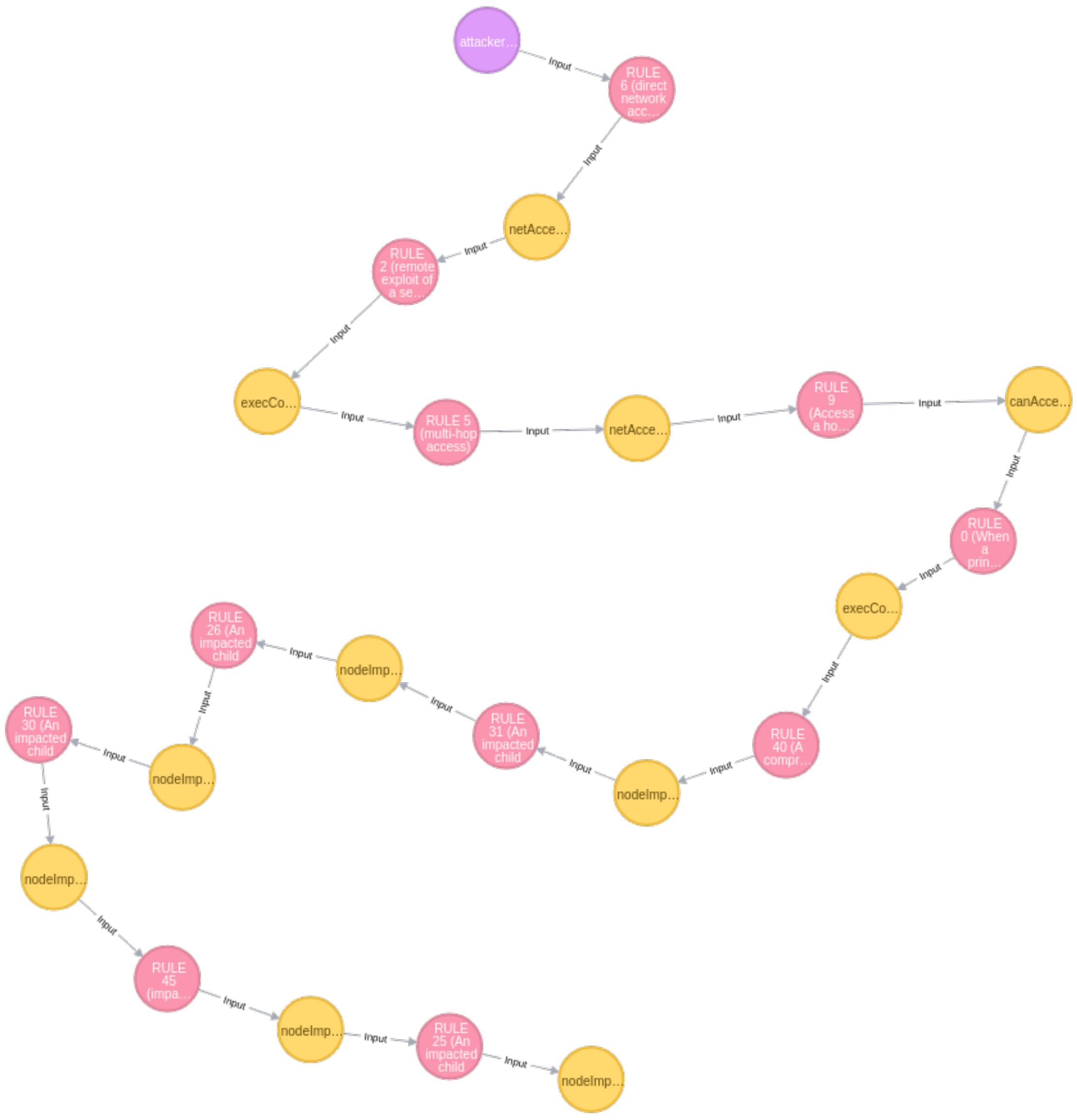

FIGURE 4.8: Shortest path from attacker to Goal in graph DB

5. Broad steps of threat analysis platform: An overview

For lack of space, we are providing only an overview. Our approach entails analyzing the given business enterprise system represented as a BPMN with the following steps:

(1) From a given BPMN model, derive task in the BP and get validated from admin for correctness. Breakdown the business process into all possible subtasks which decide the complete process. Derive logical task dependency graph from this.

(2) Get the Network design and Implementation, Access privileges and Vulnerabilities from administrator.

(3) Get the software architecture which contains details of Asset layer, Service layer and Business process task layer which would help framing possible attack paths.

(4) Create additional fact nodes as required to represent the complete business flow model.

(5) **Computational Menu on the platform : Fact Nodes**

1. System gives drop down menu for most likely Deployment, Configuration, Network info, Access Rules, Vulnerabilities.
2. Admin can include additional options above according to his Business Model.
3. Admin must add fact node based on Business Model as this step cannot be fully automated.

(6) **Computational Menu on the platform : Derived Nodes**

1. List all basic and known derived nodes for Admin to select for the business model under investigation.
2. Admin can add new derived nodes which are outcome of fact nodes, vulnerabilities and access rules which allow an attacker to impact the task or process as he progresses.
3. Only that fact node is impacted whose corresponding vulnerabilities and access rules allow. Hence the resulting derived node is compatible with the fact and rule node.
4. These derived nodes should be clubbed with one or more fact or derived nodes which act as preconditions for new rule to be applied.
5. This process is continued till attacker goal is reached.

(7) Create the input.P file which act as input to MulVAL for analysis.

1. The Input.P (datalog) file contains mainly **attackerLocated()** which is the starting point of the attack, **attackerGoal()** is the final goal of the attack. It also contains all derived and fact nodes which have been created. Proper definition including their parameters and number of parameters have to be checked. The syntax compliance is must for successful execution of MulVAL.

(8) Create all supporting Rule nodes over and above default rule nodes defined in MulVAL. These require detailed and thorough understanding of all nodes and process and tasks with software architecture. This will become clear from the following example:

1. *execCode(Container3, root): It indicates that the attacker has got root privilege on Container3 and can execute any arbitrary code.*
2. *vulExists(workstation2, 'CVE-2016-9962', dockerd, localExploit, vmEscalation): It indicates that above mentioned vulnerability(defined by CVE id*

whose details can be obtained from NIST website) is present in workstation2 through dockerd daemon which can be locally exploited giving raise to vmEscalation.

3. *containerInfo(docker3, container3, workstation3, dockerd, docker): It indicates that container3 is in docker3 which is in workstation3.*

4. *deploymentInfo(docker3, workstation3, dockerd, docker): It indicates that dockerd daemon running docker3 which is running in workstation3.*

5. From the above, the following rule is introduced.
   interaction_rule((execCode(workstation3,docker):-
   vulExists(workstation3, 'CVE-2016-9962', dockerd, localExploit, vmEscalation),
   containerInfo(docker3, container3, workstation3, dockerd, docker),
   deploymentInfo(docker3, workstation3, dockerd, docker),
   execCode(container3,root)),
   rule_desc('Container Escalation',0.4)).

(9) **Computational Menu on the platform : Interaction Rule**

1. Basic rules are already defined as part of MulVAL (kb folder)
2. New rule nodes need to be defined based on the business model and software architecture with all inputs.
3. Rule node should be appropriate and corresponding to input fact or derived node.

(10) Once the new rules are fed into Kb folder of MulVAL, then input.P can be fed as input and obtain attack tree from MulVAL as output.MulVAL generates attackgraph.eps (Attack tree graph), vertices.csv (All nodes in the attack tree), attackgraph.txt (All relations among nodes in attack tree) for the corresponding Input.P file. One can directly see attack graph or convert using eps to pdf online converter.

(11) Export attackgraph.txt and vertices.csv after doing appropriate modifications to graph DB (Neo4J) and recreate the complete attack tree.

(12) Create appropriate Cypher queries to obtain necessary intelligence required for assessing the impact.

1. **All paths from one node to any other node**
   - (All possible paths available to attacker from a particular node at any other node or Goal). This also suggests the paths available for any insider attack.
2. **Shortest, longest or K length paths from one node to another**
   - Aids in identifying shortest and all intermediate paths an attacker has on reaching a particular node to the goal or any other node. This gives the min and max resources required by an attacker to damage the system.
3. **Bottleneck or intersection nodes**
   - This aids the admin to identify where network hardening is required thereby isolating the attacker with acceptable vulnerabilities within the system. Based on threat perception, invest only in addressing that vulnerability which allows an outsider to gain entry to critical levels of system.
4. **Impact calculation on any path using any metrics for security**
   - CVSS score for a vulnerability is internationally accepted metric for given security threat in a system. It is constantly verified and updated for correctness. The score indicates the threat due to vulnerability. The same metric can be used to identify the max and min impact path in the attack graph. The metric values can be assigned to the vulnerable fact nodes and values can be propagated. Impact calculation logic is as follows:

     Node Impact Score (N_i) = $\frac{CVSS Base score for Vulnerability}{10}$

     For two node N_1 and N_2 as input:

     AND Node Impact Score = $N_1 \times N_2$

OR Node : Impact Score = $N_1 + N_2 - N_1 \times N_2$

(N_1 **OR** N_2) **AND** N_3: Impact Score = $(N_1 + N_2 - N_1 \times N_2) \times N_3$

(13) Based on the impact and vulnerabilities, decide on which vulnerability patching would meet sufficient security requirement.

(14) **Other Metric Based Evaluation**: The model has captured even the probability based assessment. The following method has been incorporated into it.

- All leaf nodes are given random probability between 0 and 1(both excluded). However the network admin is free to allot any Probability of the node being compromised based on the average observation of the network.
- The further progression of this probability from leaf to root is applied as follows:

AND Node	New Probability=Min (All incoming Probability)
OR Node	New Probability=Max (All incoming Probability)

- We find the leaf with max probability as it indicates the most likely point of attack by any attacker. From here we can find shortest path upto main goal. Hence come with all such assessments from probabilistic model in place.
- The same above model can be used to do assessment wrt :

  Detect ability: The ability of the attacker being detected due to monitors by assigning appropriate values.

  Cost: The overall cost that would be required for an attacker to commit his actions to achieve the target.

  Technical Skill: The technical skill required to make way and progress into network till goal is reached by the attacker. The skill may approximated to some metric which can be assigned to nodes.

- The model has been incorporated in such a way that we can model the network hardening based on **Green, Yellow and Red Zone**. This is modelled by collecting all shortest paths from vulnerable nodes and appropriate intersection points have been identified as demarcation boundaries for type of zone.

(15)The model can be extended to multi-business process where the same underlying asset/service layer supports more than one business process. The attacker can exploit vulnerabilities existing in this to stage attack from one business process to another.

(16) Iterate the above steps till we realize the intended level of security.

6. Conclusion and Future Work

Several works have come up with innovative solutions for threat assessment using various metrics and methods. The attack graph solutions have also been addressed by several authors. A recent approach by Singhal[10][18] has inspired our analysis. Our paper takes the concept further towards realistic assessment to any business process with a computational platform. It entails semi-automation of the whole threat assessment in which validation from admin or user is sought at regular intervals which is quite appropriate. This ensures in addition to speed in assessment, avoiding human error factors and also the automation process is cross-checked at all levels. Being pro-active, our approach aids in prevention of attacks. The model is highly scalable and up-gradable to meet most of enterprise business applications in vogue. The attacker can be tracked at each level as he progresses from leaf node to mainn goal. For better network management, we can also classify users as they reach various threat levels into color code schemes like

- Red : Severe threat to system, on attacker reaching depth of say 70% and above.
- Yellow : Medium level threat to system, on attacker reaching depth of say 50%.
- Green : Low level threat to system, on attacker reaching depth of say 10%.

Based on the above color code scheme the administrator can track the attacker and take call on network hardening and necessary actions.

Our paper highlights the usage of Graph DB for real time visualization of the complete attack graph. The Cypher queries supported by the graph DB enhances the ability to probe the attack graph from all possible approaches. It allows structuring of queries to gather information to arrive at hardening solutions. Impact recovery recommendations or self healing solutions to admin based on assessment is the highlight of our model. Our approach as described in the paper would act as a platform for further research in this field for which various machine could learning techniques could be added.

Appendices

A. Common Vulnerability Scoring System (CVSS)

This is an open framework which has unique standardized scoring system. This acts as a benchmark and is useful for practically any organization. It has three groups: Base, Temporal and Environmental. The numerical score ranges between 0-10. These scores are provided by National Vulnerability Database(NVD). The score indicates the usefulness of that vulnerability to the attacker in terms of its ease of exploit-ability and the impact it is going to have on integrity, availability and confidentiality. More the value, the better it is for the attacker. So a good attacker would look for high value vulnerabilities in the system.

Base: The constant basic characteristics of a vulnerability

Temporal: The characteristics that change on time but not on environment

Environment: Character that are specific to particular environment

B. Multihost, Multistage Vulnerability Analysis - MulVAL

MulVAL- is a software research tool which requires an input file which should constitute Network configuration, Machine configuration, all Access and Binding rules. If any dependencies other than default are fed, then accordingly, interaction rules are to be incorporated for the same. MulVAL uses Datalog as modelling language for all inputs to analyze.

Input requires: Host configuration, Network Configuration, Principals, Vulnerability, Task definition, Access Controls.

B.1 Interaction Rules

These are the rules defining how and what paths are available to the attacker in multistage and multi path attack. We start with what would be final occurrence which are followed by all those which aid in achieving these. It forms the complete logic by which MulVAL generates attack tree with all its dependent entities. There are default interaction rules defined inside MulVAL/kb folder. Based on Business Model or Process under analysis, new rules need to be added. Now, MulVAL does not know where to look for new routines that input file requires so it will throw error. Therefore one has to define these rules in interaction rules.P. The defined new rules should capture all new definitions/routines.

Bibliography

[1] B. Schneier. Dobbs J. Attack trees: modeling security threats. URL http://www.ddj.com/security/184414879.

[2] Ind 2004. B. Schneier . Wiley, Indianapolis. Secrets and lies: Digital security in a networked world.

[3] Chris Salter, O. Sami Saydjari, Bruce Schneier, and Jim Wallner. Toward a secure system engineering methodolgy. In *Proceedings of the 1998 Workshop on New Security Paradigms*, NSPW '98, pages 2–10, New York, NY, USA, 1998. ACM. ISBN 1-58113-168-2. doi: 10.1145/310889.310900. URL http://doi.acm.org/10.1145/310889.310900.

[4] J .D. Weiss 14th Annual NCSC/NIST National Computer Security Conference. pp. 572581. A system security engineering process.

[5] Edward G. Amoroso. *Fundamentals of Computer Security Technology*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1994. ISBN 0-13-108929-3.

[6] P. Ongsakorn, K. Turney, M. Thornton, S. Nair, S. Szygenda, and T. Manikas. Cyber threat trees for large system threat cataloging and analysis. In *2010 IEEE International Systems Conference*, pages 610–615, April 2010. doi: 10.1109/SYSTEMS.2010.5482351.

[7] Jan Steffan and Markus Schumacher. Collaborative attack modeling. In *Proceedings of the 2002 ACM Symposium on Applied Computing*, SAC '02, pages 253–259, New York, NY, USA, 2002. ACM. ISBN 1-58113-445-2. doi: 10.1145/508791.508843. URL http://doi.acm.org/10.1145/508791.508843.

[8] Vineet Saini, Qiang Duan, and Vamsi Paruchuri. Threat modeling using attack trees. *J. Comput. Sci. Coll.*, 23(4):124–131, April 2008. ISSN 1937-4771. URL http://dl.acm.org/citation.cfm?id=1352079.1352100.

[9] Secureitree. . URL http://www.amenaza.com/documents.php.

[10] Chen Cao, Lun-Pin Yuan, Anoop Singhal, Peng Liu, Xiaoyan Sun, and Sencun Zhu. Assessing attack impact on business processes by interconnecting attack graphs and entity dependency graphs. 2018.

[11] Bpmn. . URL https://www.omg.org/spec/BPMN/2.0/About-BPMN/.

[12] Neo4. . URL https://neo4j.com/.

[13] Neo4j. . URL https://www.tutorialspoint.com/neo4j/neo4j_tutorial.pdf.

[14] Lingyu Wang, Steven Noel, and Sushil Jajodia. Minimum-cost network hardening using attack graphs. *Computer Communications*, 29(18):3812–3824, 2006.

[15] S. Jajodia, S. Noel, P. Kalapa, M. Albanese, and J. Williams. Cauldron mission-centric cyber situational awareness with defense in depth. In *2011 - MILCOM 2011 Military Communications Conference*, pages 1339–1344, Nov 2011. doi: 10.1109/MILCOM.2011.6127490.

[16] Rajesh Kumar. *Truth or dare: quantitative security risk analysis via attack trees*. PhD thesis, University of Twente, Netherlands, 10 2018.

[17] Xiangdong An, Dawn Jutla, and Nick Cercone. Privacy intrusion detection using dynamic bayesian networks. In *Proceedings of the 8th International Conference on Electronic Commerce: The New e-Commerce: Innovations for Conquering Current Barriers, Obstacles and Limitations to Conducting Successful Business on the Internet*, ICEC '06, pages 208–215, New York, NY, USA, 2006. ACM. ISBN 1-59593-392-1. doi: 10.1145/1151454.1151493. URL http://doi.acm.org/10.1145/1151454.1151493.

[18] Xiaoyan Sun, Anoop Singhal, and Peng Liu. Towards actionable mission impact assessment in the context of cloud computing. In Giovanni Livraga and Sencun Zhu, editors, *Data and Applications Security and Privacy XXXI*, pages 259–274, Cham, 2017. Springer International Publishing.