Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 007:004.056
DOI: https://doi.org/10.24867/06BE13Kovacevic

СИСТЕМ ЗА УПРАВЉАЊЕ РАЊИВОСТИМА У СОФТВЕРУ

SOFTWARE VULNERABILITY MANAGEMENT SYSTEM

Марија Ковачевић, *Факултет техничких наука, Нови Сад*

Област – СОФТВЕРСКО ИНЖЕЊЕРСТВО И ИНФОРМАЦИОНЕ ТЕХНОЛОГИЈЕ

Кратак садржај – *У раду је описана имплементација система за управљање рањивостима у софтверу и објашњени су механизми за откривање јавно идентификованих рањивости.*

Кључне речи: *Моделовање претњи, дијаграм тока података, база знања, NVD, CPE, CVE, XML*

Abstract – *This paper presents implementation of the Software Vulnerability Management System. Description of the mechanisms for detecting publicly known vulnerabilities is given.*

Keywords: *Threat modelling, Data flow diagram, Knowledge base, NVD, CPE, CVE, XML*

1. УВОД

У данашње време, напади на софтверске системе постају све учесталији. Нападач може да угрози читав систем нападом на само један ток података или један елемент система. Произвођачи софтвера се константно боре са овим проблемом и раде на његовом сузбијању. Један од начина за решавање овог проблема јесте разматрање безбедносних аспеката током читавог животног века софтвера укључујући и фазу дизајна производа [1]. Дугогодишњим радом, велики број техника је развијен у сврху решавања различитих безбедносних проблема. Једна од тих техника је моделовање претњи система.

Моделовање претњи система захтева разумевање комплексности целог система и идентификовање свих могућих претњи, без обзира да ли напад може одмах да се реализује [1]. Овај посао треба радити систематично и на време, јер се само на тај начин може гарантовати да ће потенцијалне претње и рањивости бити откривене од стране програмера, а не нападача. Ако се моделовање претњи ради након што је производ већ у употреби, сузбијање напада и отклањање рањивости ће бити много захтевније и скупље, него да се вршило у фази дизајна система.

Тема овог рада је креирање Система за управљање рањивостима у софтверу. Систем треба да на основу дијаграма тока података креира извештај са потенцијалним нападима на софтвер и листом јавно идентификованих рањивости у компонентама софтвера.

НАПОМЕНА:
Овај рад проистекао је из мастер рада чији ментор је био проф. др Горан Сладић.

2. МЕХАНИЗМИ ЗА ОТКРИВАЊЕ РАЊИВОСТИ

Сви софтвери, који поседују механизме за откривање рањивости, у великој мери умањују степен ризика од потенцијалних напада и обезбеђују ефикаснију израду буџета за дати софтвер.

2.1. Дијаграм тока података

Дијаграм тока података (ДТП) је графички дијаграм за конструкцију и визуелизацију модела система [2]. Он служи за дефинисање захтева у графичком приказу. Користи се приликом моделовања претњи, јер је лак за разумевање и усмерен је на податке и њихов ток [3]. ДТП има 4 врсте компоненти [4]:

- Процес - активност или функција, која извршава одређену бизнис логику.
- Складиште података - репозиторијум где се подаци чувају или одакле се добављају.
- Ток - репрезентује податке, који се преносе између елемената.
- Спољни елементи - елементи ван система, који су у интеракцији са њим.

2.2. Шема за именовање производа

Common Platform Enumeration (CPE) је стандардизован начин за описивање и идентификацију класа апликација, оперативних система и хардверски уређаја [5]. CPE идентификује производе тако што за сваки наводи низ парова атрибут-вредност.

2.3. Листа рањивости и изложености

Листа рањивости и изложености (енг. *Common Vulnerabilities and Exposures*) (CVE) је листа јавно познатих рањивости и изложености из поља рачунарске безбедности [6]. Ова листа је бесплатна и доступна свима. Састоји се од ставки, где свака ставка представља рањивост и омогућава лако приступање информацијама о њој уз помоћ више референци које је чине [6]. Ставку чине:

- идентификатор,
- опис,
- минимум једна референца ка јавно познатој рањивости.

2.4. Национална база рањивости

Национална база рањивости (енг. *National Vulnerability Database*) (NVD) је база владе Сједињених Америчких Држава објављена 2005. године од стране Националног института за стандарде и технологију [7]. Задатак базе је да преузима ставке из CVE листе и да врши њихову анализу. Експерти за безбедност анализирају CVE ставке и проширују их са додатним метаподацима попут: CPE, CVSS, итд. [8].

2.5. Систем за оцењивање рањивости

Систем за оцењивање рањивости (енг. The Common Vulnerability Scoring System – CVSS) обрађује главне техничке карактеристике рањивости софтвера и хардвера. Резултат анализе су нумеричке вредности које указују на озбиљност анализиране рањивости у односу на друге рањивости [9].

3. СИСТЕМИ ЗА МОДЕЛОВАЊЕ ПРЕТЊИ

Моделовање претњи је техника анализе дизајна [3]. Може да буде базирана на ресурсима, (енг. *Asset-centric*), нападачу (енг. *Attacker-centric*) или софтверу (енг. *Software-centric*). Резултат моделовања су рангиране претње моделованог система.

Моделовање могу да раде експерти, инжењери или и експерти и инжењери заједно. Тежи се оспособљавању инжењера да самостално моделују, јер ангажовање експерата може да буде веома скупо [3].

Главни циљ моделовања је унапређење безбедности софтвера [3]. Моделује се у раним фазама развоја, јер нападачи могу да направе велику штету произвођачима. Штета која је материјалне природе се надокнадити, али нарушен кредибилитет се тешко поново стиче, зато произвођачи све озбиљније схватају зашто је моделовање битно.

У овом поглављу су описана два алата за моделовање претњи: *Threat Modelling Tool* и *Систем за проналажење напада*. Систем за проналажење напада је настао по узору на Threat Modelling Tool и исправио је суштинске недостатке тог алата.

3.1. Threat Modelling Tool

Threat Modeling Tool (TMT) је алат који омогућава софтверским архитектима да рано открију и уклоне потенцијалне проблеме из домена безбедности софтвера.

Рано уочени проблеми у великом мери смањују укупне трошкове и време утрошено за развој софтвера [10].

Цео алат је изузетно једноставан за употребу, јер је дизајниран тако да особе које нису експерти из области безбедности, могу да направе и анализирају дијаграме тока података [10].

3.2. Систем за проналажење напада

Систем за проналажење напада (енг. *Exploits Detection System*) (СПН) је софтверски алат који се користи за моделовање претњи [11].

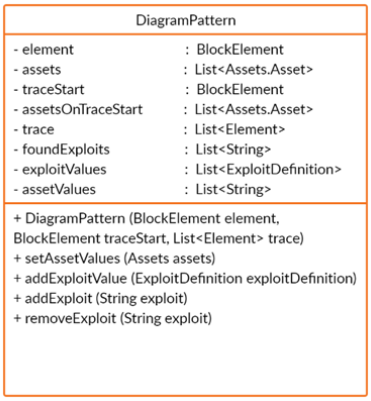
СПН као улазне параметре прима дијаграм тока података у XML формату, XML датотеку са дефиницијом ресурса и XML датотеку са дефиницијом напада. Као резултат кориснику се враћа извештај о потенцијалним нападима.

СПН је исправио суштинске недостатке које постоје у TMT-у [11]:

- Немогућност извоза креираних ДТП-ова.
- Непостојање ресурса (енг. *Asset*) као фактора при анализи и креирању ДТП-а
- Ограниченост шаблона при анализи дијаграма

4. МОДЕЛ СИСТЕМА

Декомпоновање дијаграма тока података резултује листом шаблона. Сваки шаблон је представљен класом *DiagramPattern* (слика 1). Он садржи почетни и крајњи елемент тока података, као и ресурсе на тим елементима. Када се заврши анализа шаблона и пронађу потенцијални напади, они се смештају у листу *foundExploits*. Касније се, приликом увезивања дефиниција напада, вредности из ове листе замене описима напада и контрамерама за сузбијање истих.



Слика 1. *Класа шаблона*

Систем за управљање рањивостима у софтверу приликом генерисања потенцијалних напада пролази кроз две велике фазе:

I Анализа дијаграма и генерисање рањивости за комплексне елементе.

II Генерисање рањивости за цео дијаграм и креирање извештаја.

5. ИМПЛЕМЕНТАЦИЈА СИСТЕМА

У наредном поглављу бити представљена имплементација веб-апликације Система за управљање рањивостима у софтверу.

5.1. Имплементација базе знања

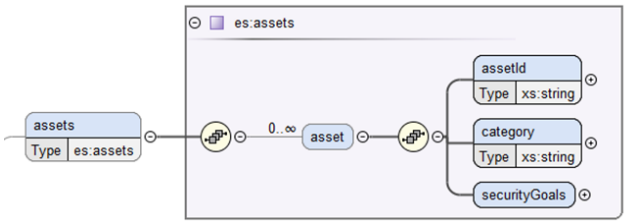
Сваки пронађени шаблон на ДТП бива прослеђен бази знања са циљем да се анализира и да се пронађу потенцијалне претње. Анализа шаблона резултује листом напада, који могу да се изврше над њим.

База знања система је имплементирана уз помоћ *Drools* технологија. База се састоји од једног документа у ком се налазе сва правила.

Правила су подељена на подгрупе у зависности од типа елемента, а то су:

- правила за процесе,
- правила за складишта података,
- правила за спољне елементе.

Елементи ДТП-а садрже листу ресурса. Сваки ресурс има ID, категорију ресурса којој припада и листу сигурносних циљева. На слици 2. налази се приказ дела шеме ДТП-а.



Слика 2. *Део шеме ДТП – приказ ресурса*

Сваки сигурносни циљ има назив и приоритет. Сигурносни циљеви су:

- поверљивост
- интегритет и
- доступност

Сигурносни циљеви ресурса се преносе на елементе. Када елемент садржи ресурс чија нпр. поверљивост треба да се заштити, онда се тај сигурносни циљ пребацује на елемент.

Ако елемент *складиште података* поседује ресурс *лозинке*, који има сигурносни циљ *поверљивост*, онда се штити поверљивост целог елемента тј. *складишта података*.

На листингу 1. налази се пример правила за детектовање *sniffing* напада. Овај напад се дешава у ситуацијама када се неовлашћено пресрећу пакети података.

Правило пролази кроз све ресурсе на елементу и проверава да ли је један од сигурносних циљева поверљивост. Ако се штити поверљивост ресурса, следи да се штити и поверљивост елемента. Уколико се користи HTTP протокол као веза између елемената, онда се веома лако може нарушити поверљивост података реализовањем *sniffing* напада.

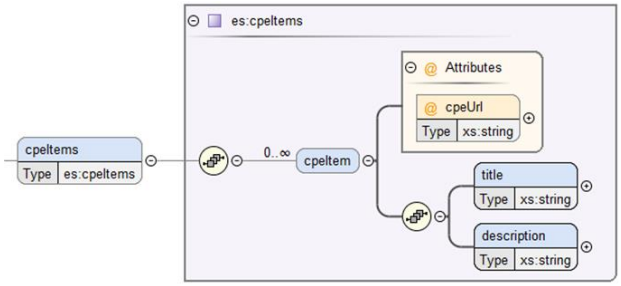
```
rule "AP:_Check_Confidentiality"
 agenda-group "ap_check_primary"
 when
  pattern: DiagramPattern($list : assets)
  $asset : Assets.Asset() from $list
       $securityGoals : Assets.Asset
                        .SecurityGoals() from
                         $asset.securityGoals
       $goal : Assets.Asset.SecurityGoals
                         .SecurityGoal() from
                  $securityGoals.securityGoal
  Boolean(booleanValue == true) from $goal
                       .name == "confidentiality"
  elementInTrace : Element() from pattern
                                      .trace
       DiagramPattern(elementInTrace instanceof
                                       Http)
 then
  pattern.addExploit("ed_sniffing");
end
```

Листинг 1. *Једно од правила које посматра цео ток података*

Ако је услов из правила задовољен тј. потенцијални напада је откривен, онда се извршава последица правила и идентификатор *sniffing* напада „*ed_sniffing*" се чува у оквиру тренутно анализираног шаблона.

5.2. Имплементација компоненте за откривање рањивости

Елементи ДТП-а, осим листе ресурса, могу да садрже листе дефинисаних CPE-ова (слика 3). Свака CPE ставка из листе садржи два елемента: наслов и опис. Најбитнији део CPE ставке је атрибут *cpeUrl*. У овом атрибуту ће се налазити путања, која идентификује CPE ставку и користи се приликом претраге базе рањивости.



Слика 3. *Део шеме ДТП – листа CPE-ова*

Систем може у сваком тренутку да контактира NVD и да затражи листу свих рањивости са последњим, најновијим изменама. Листа свих рањивости се добија у виду JSON датотеке. Прво се врши парсирање датотеке, а потом следи претрага листе за задати CPE.

5.3. Имплементација комплексног процеса

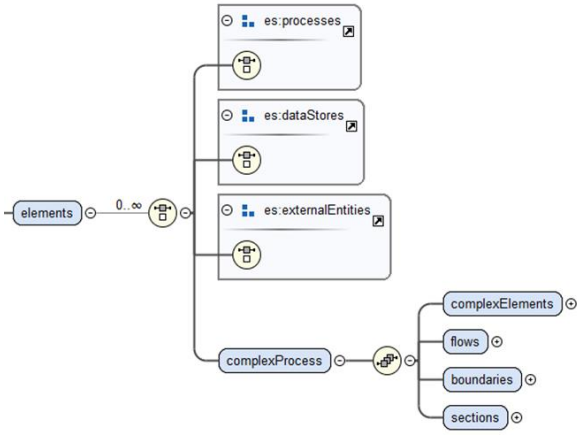
Шема ДТП-а поседује 4 елемента:

- елементи (енг. *elements*)
- токови (енг. *flows*)
- границе (енг. *boundaries*)
- области (енг. *sections*)

Елементи могу бити:

- процеси,
- спољни елементи,
- складишта података или
- комплексни процес.

Комплексни процес је нова врста елемента. То је чвор који поседује сопствени дијаграм тока података на ком је разложен у више целина. На слици 4. се налази део XML шеме ДТП на ком су приказани типови елемената.



Слика 4. *Део шеме ДТП – врсте елемената*

На почетку процеса анализе дијаграма, један од корака је декомпоновање система. Сваки пут када се пронађе комплексни процес, он се смешта у листу

комплексних процеса у оквиру дијаграма. Пронађени комплексни процеси се даље декомпонују на шаблоне, који се прослеђују бази знања на анализу.

Пошто се активирају одговарајућа правила из базе знања и идентификатори потенцијалних напада сачувају у шаблонима, учитава се XML датотека са дефиницијама напада. Следи замена идентификатора са целом дефиницијом напада.

Сви пронађени напади се смештају у листу и враћају кориснику. Корисник анализира потенцијалне нападе и одлучује које жели да пренесе на дијаграм. Напади који „опстану“ улазе у финални извештај.

5.4. Имплементација компоненте за генерисање извештаја

Финални извештај садржи:

- датум и време креирања извештаја,
- назив датотеке ДТП, који је анализиран,
- листу шаблона са пронађеним претњама,
- листу претњи које су пронађене на комплексним процесима,
- листу јавно идентификованих рањивости за пронађене CPE-ове и
- листу свих претњи са описом и контрамерама.

Добијени извештај у XML формату и креирани XSLT документ се прослеђују XSLT процесору, који врши трансформацију. Резултат трансформације је XHTML од ког се креира PDF.

6. ЗАКЉУЧАК

Тема овог рада је креирање система за управљање рањивостима у софтверу. Представљен је систем који на основу дијаграма тока података креира извештај са листом потенцијалних напада и листом јавно идентификованих рањивости у компонентама софтвера.

Објашњени су неки од механизама за откривање јавно идентификованих рањивости у софтверу. Описана је Листа рањивости (CVE), Национална база рањивости (NVD), као и ефикасан систем за оцењивање рањивости (CVSS).

У поглављу 3 описана су два алата: Threat Modelling Tool и Система за проналажење напада.

У поглављу 4 је објашњен модел система и кораци у раду.

Систем за управљање рањивостима у софтверу се ослања на СПН и врши његово унапређење. У поглављу 5 је приказана имплементација веб-апликације система и кључне новине, које је донела. Једна од новина је база знања, која више није базирана на ресурсима. Такође, имплементирана је и нова компонента, која служи за откривање јавно идентификованих рањивости. У овом поглављу је описан и комплексни процес, нови елемент ДТП. Поглавље се завршава приказом рада компоненте за генерисање извештаја, која генерише извештај у PDF формату.

Препорука за даљи развој веб-апликације је побољшање перформанси претраге јавно идентификованих рањивости. Тренутно претрага JSON датотеке са свим рањивостима, одузима највише времена у процесу анализе. Размотрити употребу *Elasticsearch*-а или неког другог алата.

Даљи развој веб-апликације може да иде у смеру увођења компоненте за графички интерфејс. Увођењем ове компоненте драстично би се унапредио рад система, јер улазни дијаграм тока података не би морао више да се описује у текстуалној датотеци.

7. ЛИТЕРАТУРА

[1] Rosziati Ibrahim, Siow Yen Yen, „Formalization of the data flow diagram rules for consistency check“, Internationanal Journal of Software Engineering & Applications (IJSEA), 2010

[2] Suvda Myagmar, Adam J. Lee, William Yurcik, „Threat Modeling as a Basic for Security Requirements“, National Center for Supercomputing Applications (NCSA)

[3] Adam Shostack, „Experiences Threat Modeling in Microsoft“, Microsoft

[4] Marwan Abi-Antoun, Daniel Wang, Peter Torr, „Checking Threat Modeling Data Flow Diagrams for Implementation Conformance and Security“

[5] https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir7695.pdf (приступљено у септембру 2019)

[6] CVE Official Specification, https://cve.mitre.org/

[7] https://nvd.nist.gov/general (приступљено у септембру 2019)

[8] Clement Elbaz, Louis Rilling, Christine Moris, „Towards Automated Risk Analysis of "One-day" Vulnerabilities“

[9] https://www.first.org/cvss/specification-document (приступљено у септембру 2019)

[10] https://docs.microsoft.com/en-us/azure/security/develop/threat-modeling-tool (приступљено у септембру 2019)

[11] Немања Миладиновић, Проналажење рањивости у софтверу на основу дијаграма тока података, Fakultet tehničkih nauka, Novi Sad, 2017.

Кратка биографија:

Марија Ковачевић рођена је у Сомбору 1995. год. Основну школу „Мирослав Антић“ у Оџацима завршила је 2010. године. Исте године уписује гимназију „Јован Јовановић Змај“ у Оџацима, општи смер. Гимназију завршава 2014. године као носилац Вукове дипломе и звања ђака генерације. Године 2014. уписује Факултет техничких наука у Новом Саду, смер Софтверско инжењерство и информационе технологије. Полаже све испите предвиђене планом и програмом са просечном оценом 9.88. 2018. године завршава основне студије и уписује мастер академске студије на истом факултету. Полаже све испите мастер студија предвиђене планом и програмом са просечном оценом 10.00.