Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 007:004.056
DOI: https://doi.org/10.24867/20BE22Kovacevic

АНАЛИЗА И ПОРЕЂЕЊЕ АЛАТА ЗА ФАЗ ТЕСТИРАЊЕ IoT СИСТЕМА

ANALYSIS AND COMPARISON OF FUZZ TESTING TOOLS FOR IoT SYSTEMS

Тамара Ковачевић, *Факултет техничких наука, Нови Сад*

Област – РАЧУНАРСТВО И АУТОМАТИКА

Кратак садржај – *Интернет, стуб друштва без ког се више не може, створио је нови начин за комуникацију, коришћење и дељење информација. Брз развој интернета ствара нову визију свакодневних ствари, омогућава развој новог концепта - Интернет ствари (енгл. Internet of things, IOT). IoT омогућава размену мноштва корисних информација, али истовремено доводи и до забринутости у погледу безбедности и приватности. Једна од ефикаснијих метода за тестирање сигурности софтвера јесте тестирање фаз методом. У склопу овог рада анализирани су концепти фаз тестирања IoT уређаја, упоређивани су различити алати и дат је предлог решења њихових недостатака.*

Кључне речи: *безбедност, фаз тестирање, интернет ствари, алати*

Abstract – *The Internet, a pillar of society that we can no longer live without, has created a new way of communication, data usage, and information sharing. The rapid development of the Internet has created a new vision of everyday things and enabled the development of a new concept - the Internet of Things (IoT). IoT enables the exchange of a massive amount of useful information but also raises security and privacy concerns. Fuzz testing is one of the more effective methods for testing software security. This master thesis explores the concepts of fuzz testing for IoT and their limitations. Different tools were compared, and a proposal was conducted to solve their shortcomings.*

Keywords: *security, fuzz testing, Internet of Things, tools*

1. УВОД

Не тако давно само су нам телефони били "паметни", међутим данас постоје и паметни сатови, машине, куће па чак и градови. Када се једном нађу на интернету, те "ствари" могу да комуницирају, како међусобно, тако и са својим корисницима. Прегледом ових принципа долази се до појма Интернет ствари (енгл. Internet of Things, IoT). IoT уређаји олакшавају свакодневни живот и омогућавају пренос различитих информација. Врста и количина информација којима IoT уређаји имају приступ, са собом носе и забринутост по питању одржавања њихове безбедности. Како би се осигурали сигурност и квалитет софтвера, неопходно је вршити његово тестирање. Фаз метод јесте једна од актуелних метода за тестирање сигурности софтвера. Применом ове методе може се тестирати врло широк спектар различитих система, а могуће је пронаћи како грешке које представљају функционалне проблеме, тако и рањивости које могу бити искоришћене од стране злонамерних нападача. За потребе фаз тестирања користе се специјализовани алати, који у систем убризгавају различите параметре и прате његово даље понашање. Кроз овај рад анализиран је безбедносни аспект IoT система и начини фаз тестирања истих. Описано је неколико актуелних алата за извођење ове врсте тестирања, након чега је извршено њихово поређење и дат је предлог решења њихових недостатака.

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

2. КАРАКТЕРИСТИКЕ IoT УРЕЂАЈА

Данас су међусобно повезани уређаји попут паметних сатова, аутомобила, кућа, медицинске опреме итд. Да би се принципи IoT технологије остварили у пракси неопходно је да су испуњена два главна предуслова, комуникација и јединствена идентификација ресурса. За обезбеђивање успешне комуникације, трансфера информација и аналитике користе се различити протоколи. Осим тога, неопходно је да сваки IoT уређај поседује јединствени нумерички или симболички идентификатор. Савремени IoT уређаји се састоје из мноштва сензора путем којих прикупљају различите информације. Одликује их веома изражена динамичка самоадаптивност као и способност самоконфигураци-је. Способни су да се конфигуришу уз минимално учешће корисника, а поред тога се и прилагођавају и реагују на промене из окружења. Минијатуризација ових уређаја побољшава њихову преносивост и доприноси прилагодљивости окружењу [1].

2.1. Безбедност IoT уређаја

Када је у питању домен интернет ствари, безбедносне претње се односе првенствено на аутентичност, поверљивост и интегритет података који постоје у оваквим системима. Поједини нивои технологије због својих ограничених процесорских капацитета могу бити злонамерно искоришћени за иницијални приступ систему. Одсуство аутентификације и других безбедносних механизама у сензорској опреми представља велики проблем. Такође, интерфејси између платформи и крајњих корисника могу проузроковати проблеме због небезбедних комуникационих протокола. При преносу информација до облака у ком се складиште, али и при самом складиштењу користе

се различити механизми и платформе који стварају проблем небезбедних података [2].

3. ФАЗ МЕТОД ТЕСТИРАЊА

Једна од популарних динамичких метода за тестирање названа је фаз методом. Фаз начин тестирања истражује стање софтвера тако што га испитује случајним, псеудо-случајним или делимично невладиним подацима. Примене ове методе су далекосежне, погодна је за тестирање веб-апликација, протокола, функција итд. Иако фаз тестирање не може да замени потребу за основним тестирањем софтвера, комбинације више техника заједно са фаз методом доприноси бољој покривености кода [3].

3.1. Стратегије фаз тестирања

Како је опсег система које је могуће тестирати на овај начин готово неограничен, постоји и више начина извођења ове методе тестирања. У зависности од тога да ли је фаз систем свестан структуре улазних параметара, системе можемо поделити на неструктуиране и структуиране. Најједноставнији начин за генерисање улазних података за фаз тестирање јесте креирање истих од нуле при сваком тесту, на овој теорији се заснива фаз тестирање које ради на принципу генерисања нових података у свакој итерацији.

Насупрот томе, фаз метод заснован на мутацији података преузима од корисника један или више тестних улаза, а затим их насумично модификује. Модификација подразумева измену сличном вредношћу, брисање или дуплирање. Уколико је позната унутрашња структура система, користи се фаз метод "беле кутије", док се системи који се заснивају на непознавању унутрашње структуре називају системи "црне кутије" [4].

3.2. Фазе фаз тестирања

Како би тестирање функционисало што ефикасније, фаз системи своје извршавање деле у неколико различитих фаза [4]:

- Идентификација система
- Идентификација улазних параметара
- Генерисање фаз података
- Пропагација тест случајева као улаз програма
- Праћење одговора система.

4. СИСТЕМИ ЗА ИЗВРШАВАЊЕ ФАЗ ТЕСТИРАЊА

За потребе фаз тестирања користе се специјализовани алати - фазери. Иако међу њима постоје многе разлике, сви имају заједнички циљ. Теже да обезбеде брзо, флексибилно и хомогено развојно окружење. Добар алат за фаз тестирање треба да апстрахује и минимизује бројне заморне задатке.

На способности данашњих алата утиче употреба напреднијих манипулација заснованих на три основна типа, манипулација вредностима, манипулација протоколима и манипулација датотекама [3].

Главни изазов са којим се суочавају фаз алати данашњице јесте време потребно да се простор комбинација улазних параметара исправно покрије.

4.1. Откривање рањивости

Фаз алати најбоље функционишу за откривање рањивости које описују преливање бафера, XSS пропусте или DoS, SQL нападе. Поменуте нападе често користе злонамерни нападачи са циљем да изазову проблеме у софтверу. Са друге стране, фаз тестирање је мање ефикасно за суочавање са безбедносним претњама које не изазивају рушење програма, као што су различити шпијунски софтвери, вируси итд. [5].

4.2. Одређивање квалитета фаз тестирања

Постоји много простора за побољшање квалитета фаз тестирања. Мерило квалитета свакако није број пронађених грешака у јединици времена. Један од најважнијих фактора при тестирању јесте брзина извршавања. Што већи број тестних случајева се изврши у краћем временском периоду, већа је вероватноћа да ће се пронаћи грешка у систему. Уз то, идеја сваког алата за тестирање јесте да што је већа покривеност кода, програм је боље тестиран. Додавши уз то и минимизовање тестних случајева који не изазивају грешке и ефикасну класификацију грешака, убрзава се процес тестирања и долази се до система високог квалитета и велике ефикасности [5,6].

4.3. Предности и мане фаз тестирања

Фаз тестирање нуди широк спектар погодности по питању безбедности и квалитета програма. Пружа широку општу слику квалитета циљног система и софтвера. Користећи овај метод тестирања може се лако проценити робусност и безбедносни ризик система. Осим тога, има ниске трошкове, а уз то неколико машина може истовремено да тестира изворни код, што чини фаз тестирање високо скалабилном техником тестирања. Највећа предност ове методе тестирања је у томе што проналази специфичне грешке које се тешко откривају класичним методама тестирања [7].

Са друге стране, руковање софтвером уме да буде веома комплексан процес. Фаз алати отвореног кода захтевају много ручног напора да би се постигли ефикасни резултати тестирања. Велики проблем овог типа тестирања лежи у људским ресурсима. Недостатак стручњака за безбедност умногоме отежава интеграцију ове технике тестирања у компаније. Случајност улаза који се користе у фаз тестирању се у неку руку може сматрати недостатком, јер уочавање граничних вредности са случајним улазима је мало вероватно, при чему се може очекивати да ће сваки фаз алат пронаћи различит скуп грешака [7].

5. АЛАТИ ЗА ФАЗ ТЕСТИРАЊЕ IoT УРЕЂАЈА

Како би се заштитили IoT уређаји и како би се гарантовала њихова безбедност, неопходно је тестирати све делове система. Због ограничености IoT уређаја, потешкоћа у извлачењу информација и опонашању прилагођених софтвера, фаз тестирање методом црне кутије често је једина одржива опција [3].

5.1. Изазови алата за фаз тестирање IoT уређаја

На основу чињенице да нема доминантних и стандардизованих протокола за домен IoT уређаја, произвођачи имају тенденцију да формирају себи прилагођене формате података и да дизајнирају различите протоколе како би испунили све захтеве производа.

Осим тога, користе и нестандардизоване функције за шифровање порука. Стога, у циљу генерисања протоколом вођених и криптографски конзистентних порука, неопходно је да фазери прате формате прописане од стране произвођача.

Строге граматичке спецификације фазера, ограничавају способности метода заснованих на мутацији. Већина порука које су генерисане насумичном мутацијом крше правила синтаксе и бивају брзо одбијене. Поред наведеног, велики проблем представља и недостатак механизма повратних информација. Произвођачи IoT уређаја не дају довољно информација о софтверима који се налазе у уређају.

Без приступа самом фирмверу скоро је немогуће добити информације о интерном извршавању унутар уређаја, што је полазна тачка већине типичних фазера. Случајност у одговорима IoT уређаја је нешто што умањује ефикасност фазера. Поруке од уређаја могу да садрже и насумичне елементе попут временских ознака или различитих токена. У таквим случајевима за исти улаз IoT уређаји дају различите излазе [8].

5.2. IoTFuzzer

Алат за аутоматско извођење фаз тестирања, IoTFuzzer, има за циљ да пронађе меморијске рањивости у IoT уређајима без приступа њиховом фирмверу. Кључна идеја заснована је на запажању да се већина IoT уређаја контролише путем њихових званичних мобилних апликација. Апликације тог типа представљају контролну таблу уређаја, садрже информације о уређајима, коришћеним протоколима, различите URL адресе, као и мноштво шема за шифровање/дешифровање.

IoTFuzzer врши динамичку анализу и идентификује садржај унутар апликација које формирају поруке. Након ових корака врши аутоматску мутацију порука и исте испоручује до уређаја. На вишем нивоу архитектуре овог алата постоје две главне фазе: фаза анализе апликација и фаза фаз тестирања [9].

Први корак IoTFuzzer-a јесте анализа корисничког интерфејса мобилне апликације за управљање уређајем. Циљ ове фазе јесте анализа кода апликације и откривање свих компоненти корисничког интерфејса које доводе до слања мрежних порука. Након идентификовања потребних поља и протокола, следи фаза у којој IoTFuzzer динамички мутира поља од интереса и надгледа понашање и евентуални престанак рада IoT уређаја.

Овај алат поседује унапређену логику мутирања порука, водећи се насумичним мутирањем подскупа поља од интереса. Након слања података следи фаза у којој се прати статус покретања IoT уређаја са мутираним подацима [9].

5.3. Snipuzz

За разлику од постојећих приступа за фаз тестирање IoT уређаја Snipuzz примењује стратегију мутације порука засновану на исечцима одговора. Оно што издваја Snipuzz од осталих алата исте намене јесте начин оптимизације процеса одређивања почетних података. Посматрајући одговоре добијене од IoT уређаја Snipuzz користи различите алгоритме на основу којих закључује њихово порекло.

Коришћењем новог хеуристичког алгоритма открива улогу сваког бајта у поруци добијеној као одговор IoT уређаја. Мутацијом појединачних бајта поруке алат решава проблем разноликих формата порука. Осим тога, Snipuzz унапређује свој приступ користећи хијерархијску стратегију груписања порука, чиме смањује грешке у класификацији узроковане случајностима у одговорима. Сходно наведеним унапређењима, Snipuzz, као фаз алат који користи метод црне кутије и даље може ефикасно да тестира IoT уређаје без подршке различитих граматика и интерних информација о конкретном уређају [10].

5.4. SIoTFuzzer

SIoTFuzzer користи унапређен аутоматски механизам за генерисање порука са стањем (енгл. SMG) који побољшава ефикасност фаз тестирања. Уводи технике анализе фронт-енд дела апликације као и анализе статуса. Тестни циљеви овог алата су углавном рутери и IP камере, њихова предност у раду са овим алатом јесте то што поседују кориснички интерфејс. SIoTFuzzer анализира изглед страница за управљање уређајем, што обухвата упознавање са изгледом и функционалностима. Резултат ове фазе јесте формиран речник који повезује елементе корисничког интерфејса са одређеним правилима, као и преглед редоследа акција које воде ка одређеном циљу [11].

Анализом стања система и корисничког интерфејса може се доћи до секвенце порука које одговарају различитим операцијама. Добијене поруке се филтрирају и категоризују, а њиховом комбинацијом настаје почетна порука за тестирање IoT уређаја. Како је главна идеја SIoTFuzzera генерисање порука са информацијама о стању система, изазов при мутацији јесте оставити непромењеним податке везане за ауторизацију и верификацију система. Стога, SIoTFuzzer користи више фаза мутације, које обухватају одређену и насумичну мутацију порука. Одређена фаза мутације је подељена у две подфазе: мутација параметара и структурална мутација. Насумична фаза бира неколико акција из почетне фазе које се насумично спроводе над почетном поруком [11].

5.5 IoTFuzzer vs Snipuzz vs SIoTFuzzer

Као што је истакнуто у претходним поглављима, док IoTFuzzer оштећења проналази модификацијом програмске логике у пратећој мобилној апликацији, Snipuzz додатно анализира одговоре са циљног уређаја и тиме омогућава фаз тестирање засновано на повратним информацијама. За то време SIoTFuzzer анализира кориснички интерфејс апликација за управљање уређајем, протоколе комуникације и

користи аутоматски механизам за генерисање порука са стањем. Сваки од наведених алата специјализован је или за посебан тип уређаја, за комуникацију путем специфичних протокола, или за уређаје који имају подршку додатних система за управљање [9,10,11].

Иако IoTFuzzer гарантује исправну семантику улазних порука, због чињенице да се ослања на мобилне апликације за управљање уређајем, ограничења овог алата огледају се у врсти уређаја које подржавају. Наиме подржава само Android мобилне апликације, а осим тога ограничен је и WiFi комуникацијом [9].

Насупрот њему, нешто млађи Snipuzz нуди оптимизован процес мутације. Са друге стране, највећа мана овог алата јесте зависност од одговора, као и мала покривеност кода [10].

SIoTFuzzer успешно обрађује статус конекције и нуди успешно тестирање процеса аутентификације. Као и претходно поменути алати, и SIoTFuzzer има ограничења по питању протокола који може да тестира, као и у погледу уређаја који могу да се обраде [11].

6. ЗАКЉУЧАК

У овом раду дат је преглед IoT уређаја из безбедносног аспекта, предочене су одлике фаз методе за тестирање, након чега је дат преглед алата за фаз тестирање IoT уређаја и њихово поређење.

Чињеница је да домену фаз тестирања IoT уређаја недостају јавна, генеричка решења. Комплексност система, недовољна информисаност, недостатак опште прихваћених и специјализованих протокола, као и недостатак обученог кадра одлажу потенцијално генерисање система опште намене. Како постоји много домена из којих се IoT свет може посматрати, један универзални фаз алат високе ефикасности, делује као немогуће направити. Потенцијално решење јесте синхронизација више фаз система који би могли да тестирају специфичне домене и да синхронизују своја запажања. Будућа истраживања треба да имају за циљ да обједине постојеће технике у конкретни фаз систем који би елиминисао појединачне недостатке. Још један правац развоја ових система јесу и уграђени фазери. Процес тестирања може се аутоматизовати уграђивањем фаз тестирања у стварни хардвер и извлачењем информација из постојећих функционалности. Ограничења у хардверским компонентама и самом софтверу засигурно представљају велику препреку, међутим простор за истраживање и унапређење у овом домену, свакако да постоји.

7. LITERATURA

[1] J. Gubbi, et al. “Internet of Things (IoT) a vision, architectural elements, and future directions”, 2013

[2] Shashi Rekha, Lingala Thirupathi, Srikanth Renikuntac, Rekha Gangula, “Study of security issues and solutions in Internet of Things (IoT)”, 2021

[3] Sofia Bekrar, Chaouki Bekrar Roland Groz, Laurent Mounier ,“Finding Software Vulnerabilities by Smart Fuzzing“, 2011

[4] Department of telecomunications-Government of India, “Fuzz testing”, https://tec.gov.in (приступљено u августу 2022)

[5] “OWASP Top 10”, https://owasp.org/Top10 (приступљено u сепембру 2022)

[6] Josip Bozic, Franz Wotawa “XSS Pattern for Attack Modeling in Testing”, 2013

[7] George Klees, Andrew Ruef, Shiyi Wei, Michael Hicks, “Evaluating Fuzz Testing“,2018

[8] Junjie Wang, Bihuan Chen, Lei Wei, and Yang Liu, “Skyfire: Data-driven seed generation for fuzzing”, 2017

[9] Jiongyi Chen, Wenrui Diao et al. “IOTFUZZER: Discovering Memory Corruptions in IoT Through App-based Fuzzing“, 2018

[10] Xiaotao Feng, Ruoxi Sun, Xiaogang Zhu, “Snipuzz: Black-box Fuzzing of IoT Firmware via Message Snippet Inference“, 2021

[11] Hangwei Zhang , Kai Lu, Xu Zhou, Qidi Yin, Pengfei Wang and Tai Yue, “SIoTFuzzer: Fuzzing Web Interface in IoT Firmware via Stateful Message Generation”, 2021

Kratka biografija:

Тамара Ковачевић рођена је 16. октобра 1998. године у Новом Саду. Завршила је природно-математички смер Гимназије „Јован Јовановић Змај” у Новом Саду, 2017. Године. Исте године је уписала Факултет техничких наука у Новом Саду, смер Рачунарство и аутоматика. На трећој години студија определила се за смер Примењене рачунарске науке и информатика. Основне студије завршава 2021. године, након чега уписује мастер академске студије на студијском програм Рачунарство и аутоматика, смер Електронско пословање. Испунила је све обавезе и положила је све испите предвиђене студијским програмом.