Információrendszerek biztonsági kockázatainak vizsgálata a szoftverek nyíltsága szerint

Erdősi Péter Máté

PhD hallgató
Nemzeti Közszolgálati Egyetem Közigazgatás-tudományi Doktori Iskola
e-mail: erdosi.peter.kdi@office.uni-nke.hu

Dr. Horváth Attila

Vezető kutató
Információs Társadalomért Alapítvány
e-mail: horvath.attila@infota.org

Dr. Kiss Ferenc

Kutató
Információs Társadalomért Alapítvány
e-mail: kiss.ferenc@infota.org

Absztrakt

A hacktivizmus kialakulása óta számos sérülékenység jelenik meg nap mint nap az egyes szoftverek vonatkozásában és felmerül a kérdés, hogy létezik-e olyan módszer, olyan gyártó, akinek a termékei magasabb szintű biztonságot eredményeznek az azt használó szervezetek számára. Kutatásunkban több mint három év sérülékenységeit vizsgáltuk meg gyártói szempontból, arra a kérdésre keresve a választ, hogy létezik-e olyan gyártó vagy olyan fejlesztési módszer, mely jobban vagy kevésbé érintett a folyamatosan megjelenő sérülékenységek számára és hogyan érinti ez a biztonsági követelményeket.

Eredményünk azt támasztja alá, hogy a szoftverek között a biztonság tekintetében nincs kimutatható különbség, az egyes sérülékenységek sikeres kihasználásával mind a bizalmasság, mind a sértetlenség, mind a rendelkezésre állás sikeresen támadható. Az egyes termékek közötti különbségek a vonatkozó sérülékenységek számosságában jelennek meg, mely arányos a szoftverek penetrációjával – szélesebb körben elterjedt termékhez több sérülékenység jelent meg a vizsgált időszakban.

Kulcsszavak: *sérülékenység, zárt szoftver, nyílt szoftver, formális biztonság*

Bevezetés

A szoftverfejlesztés kérdése számos vitát generált amióta csak léteznek a szoftverek. Folyamatosan keresték arra a kérdésre a választ az elméleti kutatók és a gyakorlati szakemberek, hogy létezik-e még megbízhatóbb szoftverfejlesztési módszer, illetve az egyik módszernél mennyivel jobb a másik, milyen összehasonlítási bázis milyen eredményeket produkál a különböző szoftverfejlesztési metódusok vizsgálatában. A vizsgálatnak mindig voltak megbízhatósági és biztonsági aspektusai – ha nem is mindig tudatosan voltak ezek elkülönítve. Példának okáért az, hogy egy könyvelőprogram a bevitt számadatokat vajon helyesen adja-e össze (az átvitel-bit helyesen lett-e lekezelve), ez alapvetően megbízhatósági kérdés, hasonlóan ahhoz, hogy a CUDA[1] vagy az OpenCL[2] biztosít-e jobb performanciát a GPU (grafikai feldolgozó egység) programozásban [6]. A biztonságot akkor érinti, amennyiben a bizalmasság, a sértetlenség vagy a rendelkezésre állás ellen ható fenyegető tényezőként jelenik meg az adott tulajdonság – vagyis nevezhetjük nevén: a hiba. A hibás működés kihasználásával a szoftvert olyan tevékenységek elvégzésére lehet utasítani, melyre anélkül nem lenne lehetséges, például hozzáférés megadása külső ismeretlen személynek, vagy ha rendszerszintű a hozzáférés, akkor előbb-utóbb semmilyen akadály nem gördül a szoftverhibákat sikeresen kihasználó támadó elé. Vizsgálatunkat ez utóbbi irányra terjesztettük ki, vagyis azt elemeztük, hogy létezik-e olyan gyártó, melynek termékei mentesek vagy mentesebbek a hibáktól és ezzel kapcsolatosan azt is vizsgálat alá vettük, hogy publikáltak-e már olyan módszereket, melyek alkalmazásával a szoftverekben az ismeretlen sérülékenységek felfedezési valószínűsége növelhető.

A nyílt szoftver és a zárt szoftver esete

Az internet hajnalán is voltak olyan programozók, akik a közösségi szoftverfejlesztésben látták a jövő útját, fellelkesülve egy-egy sikeres open source projekten, mint például az Apache szerver projekt [20]. Az Apache fejlesztését 1995-ben kezdték el, ekkor a szoftverpiacot a zárt forráskódok uralták. Habár közel 400 programozó (388 fő) járult hozzá kódrészlettel a projekt sikeréhez, a folyamatosan jelen lévő programozók száma 5-15 között volt. A projekt érdekessége, hogy a programozói hibákat is megkísérelte megmérni (defect density) és összehasonlítani a dobozos termékekkel. Azt találták, hogy a rendszerteszt előtt jóval kevesebb hibát észleltek, mint a dobozos termékek esetében, amire egyik logikus magyarázat lehet a programozók heterogenitása, szélesebb tapasztalata. A nyílt forráskódú

[1] CUDA: Compute Unified Device Architecture, az NVIDIA által kifejlesztett és támogatott architektúra
[2] OpenCL: A Khronos csoport által menedzselt Open Computing Language

fejlesztés innovativitásának okát abban vélték felfedezni [16], hogy a zárt forráskódú fejlesztések csökkentik a kreativitást és a motivációt, az erős szerzői jogi védelem megteremtése és kikényszerítése által. Tény, hogy az ezredforduló tájékán számos sikeres open source fejlesztés történt és az internet növekedése folyamatosan új programozókat engedett további open source projektekkel foglalkozni, ami kihatással volt az egyes programok minőségére és az elterjedtségére is. Az MIT-n olyan cikkek láttak napvilágot, melyek a szoftvergyártókat egyenesen feleslegesnek tartották [10] és a közösségi szoftverfejlesztésben látták meg a jövő kulcsát [9].

Mások a dobozos szoftverek mellett törtek lándzsát és parázs diszkussziók folytak az online tudományos lapokban. Egyik ilyen korai eset Eric S. Raymond nevéhez fűződik, aki egy szellemes hasonlattal katedrális építéséhez hasonlította a zárt forráskódú termékek és bazárhoz a nyílt forráskódú termékek készítésének folyamatát [23]. Egy nagyon vehemens választ kapott erre [4], melynek egyik éles állítása szerint a Microsoftnak meg kellene semmisülnie („Microsoft need to be destroyed."). Az éles kritikát diplomatikus módon a következő lapszámban meg is válaszolta [24], a vita azonban azóta is fennáll.

A szoftverek fejlesztésére azóta további módszerek is létrejöttek (pl. globális szoftverfejlesztés, spirális szoftverfejlesztés stb.), amelyeket vizsgálat alá is vettek [13], [3], [5], [11], [12], [18]. A mi kutatásunk hatóköre ezekre a módszerekre nem terjedt ki, vagyis azt a kérdést explicit módon nem vizsgáltuk, hogy a sérülékenységekre gyakoroltak-e ezek a módszertani változások bármilyen hatást. A rosszindulatú szoftverkódok terjedésére irányuló vizsgálat erről azonban lesújtó képet festett [26].

A sérülékenységek számának alakulása

A szoftverek sérülékenységeinek vizsgálata és az incidensek számolása szinten az internet széles körben való elterjedésével együtt jelent meg az 1980-as évektől [19]. A CERT-ek (Computer Emergency Response Team) feladata a számítógépek működéséből adódó nem kívánt események detektálása, okainak feltárása és az újbóli előfordulás megakadályozása volt megalakulásuk óta. Figyelemre méltó, hogy az internet incidens taxonómia 1998-as kidolgozásakor [14] már tudatában voltak a kutatók annak, hogy a szoftverekben léteznek ismeretlen hibák (nulladik napi sérülékenységek) és ezek számosságát az ismert hibák számához képest már akkor is jelentősnek tartották. A taxonómiák olyan fejlődésen mentek keresztül [8], [1], [22], mely a folyamat-központúság mellett mára alkalmasak kibervédelmi feladatok szervezett koordinálására is [22], külön erre a célra létrehozott intézet támogatásával. A támadások ezzel párhuzamosan az egyedi, eseti jellegű incidensektől elmozdultak a folyamatos fenyegetettség állapota felé [7]. Ma már a lakosságnak is számolnia kell egy-egy kiberbiztonsági incidens következményeivel, ha a társadalom alapvető szolgáltatásait biztosító kritikus infrastruktúrát (mai szóval élve létfontosságú

rendszerelemeket) esetlegesen támadás éri [17], [2], a biztonságról már nem csak a vállalkozásoknak kell gondoskodniuk [27]. Emiatt ezek védelme is bekerült a társadalmi érdekek közé és meg is jelent az oktatásban is [21]. A szoftveres elemeket – különösen az internetes elérés biztosítottsága esetén – számos támadás érheti sérülékenységeiken keresztül.

A vizsgálatunk tárgyát képező adatbázis 2011. október 1. – 2015. április 30. között megjelent (jelentett) sérülékenységeket tartalmaz. Az adatbázis a jelzett időszakra vonatkozóan 1.101 darab sérülékenységet foglalt magában. Az adatbázist a (2013-ban megszűnt) Puskás Tivadar Közalapítvány Nemzeti Hálózatbiztonsági Központ (PTA-CERT) és az Információs Társadalomért Alapítvány (INFOTA) együttműködése hozta létre.

Az adatbázis alapját a PTA-CERT munkatársainak gyűjtőmunkájával létrehozott sérülékenység-vektorok jelentették, melyhez – az adatbázis specifikációra vonatkozó interjúk alapján – három forrást vettek igénybe:

1. US-CERT adatbázis[3]
2. Secunia advisories adatbázis[4]
3. CVE adatbázis[5]

A fenti adatbázisokban megtalálható sérülékenységek között a PTA-CERT előszűrést végzett, így az adatbázisba csak azok a sérülékenységek kerültek bele, amelyek a PTA-CERT munkatársainak megítélése szerint a kormányzati szektort érintették vagy érinthették, továbbá valódi (proof of concept, exploitálható) sérülékenységek voltak, nem pedig feltételezettek. Túlreprezentáltak voltak a távolról kihasználható sérülékenységek, de néhány esetben lokális jelenlétet igénylő formák is előfordultak. A PTE-CERT adatbázis az US-CERT által riportolt sérülékenységekkel ki lett egészítve a 2013. január 1. és 2015. április 30. közötti adatokkal, de csak a valódi fenyegetést jelentő sérülékenységekkel (a téves riportokat kiszűrtük).

Az adatbázisban végrehajtva a nyílt-zárt fejlesztési módszer alapján történő csoportosítást, a következő eredményt kaptuk, melyet az alábbi táblázat szemléltet:

[3] US-CERT elérhető: http://www.us-cert.gov/
[4] SECUNIA Advisories elérhető: http://secunia.com/community/advisories/
[5] CVE adatbázis elérhető: http://cve.mitre.org/

1. táblázat: Sérülékenységek számossága zárt szoftverfejlesztési metódus alkalmazásával

Gyártó (zárt szoftver)	Sérülékenység
Microsoft Corporation	129
IBM Corporation	81
Oracle Corporation	57
Google Inc.	47
Adobe System Incorporated	40
Apple Inc.	35
Cisco Systems Inc.	21
SIEMENS AG.	20
Hewlett-Packard Company	17
VMware Inc.	16
Dell Computer Corporation Inc	15
Novell Inc.	13
RealNetworks Inc.	12
Kevesebb, mint 10 sérülékenységben érintett gyártó	586
Összesen:	1089

A nyílt fejlesztésű szoftverek sérülékenységeinek számossága alacsonyabb volt (18,67%), szemben a zárt szoftverek sérülékenységeinek számosságával (81,33%), azonban minden ismert és közkedvelt – és sikersztoriként is értékelt – szoftver gyártója megjelent érintettként egy-egy sebezhetőség publikálásában, ez alól a SCADA[6] rendszerek sem voltak kivételek. Ez a tény ismét alátámasztja a funkcionalitás és a biztonság elválasztásának lehetségességét és szükségességét.

[6] SCADA: supervisory control and data acquisition, felügyeleti ellenőrző és adatgyűjtő program

2. táblázat: Sérülékenységek számossága nyílt szoftverfejlesztési metódus alkalmazásával

Gyártó (nyílt szoftver)	Sérülékenység
WordPress (open source GPLv2)	52
Mozilla Foundation (Mozilla Public License)	24
Moonchild Production (open source MPL/GPL/LGPL)	8
Apache Software Foundation	7
Joomla! Project (open source)	7
open source projekt	7
Red Hat Inc.	7
Debian GNU/Linux	6
PHP Group (open source)	6
VideoLAN non-profit organisation (Open Source)	6
5 vagy kevesebb sérülékenységben érintett fejlesztő	120
Összesen:	**250**

Összefoglalás

A kapott eredmények azt támasztják alá, hogy a szoftverek között a biztonság tekintetében nincsen lényegében kimutatható különbség, az egyes sérülékenységek sikeres kihasználásával a biztonság eredményesen támadható. Az egyes termékek közötti különbségek a vonatkozó sérülékenységek számosságában jelennek meg, melyek arányosnak látszanak a szoftverek penetrációjával – szélesebb körben elterjedt termékhez több sérülékenység jelent meg a vizsgált időintervallumban. Az, hogy egy szoftver fejlesztése nyílt vagy zárt licencelést követett-e, nem volt lényeges befolyásoló tényezője a sérülékenységek megjelenésének, az ismert platformok nyilvánosan elérhető szoftvereiben előfordulhattak és elő is fordultak sérülékenységek, függetlenül a fejlesztési módszertől. Érdekes vizsgálati kérdés lehet az, hogy a sérülékenységek nulladik napi állapotának időtartama függ-e a fejlesztési módszerektől, más szóval van-e kimutatható eltérés az egyes szoftvertermékek javításának átlagos időtartamában vagy sem, azonban erre a kérdésre ez a vizsgálat nem tért ki.

A szoftverfejlesztési módszer befolyásoló tényezője a termék funkcionalitásának, ahogyan erre Illési is rávilágít az igazságügyi informatikai vizsgálati szoftverek összehasonlító elemzésében [15], azonban a sérülékenységek szempontjából nem látszott preventív eszköznek egyik módszer sem. A szükséges biztonság elérésében a formális metódusok (FM) alkalmazása tűnik olyan eszköznek [25], mely alkalmasnak látszik arra, hogy az eddig rejtett módon megjelenő hibák létezésének valószínűségét csökkentse, a formalitás növelésével azokat a szabadsági fokokat redukálva, melyek elsődleges okai voltak a szoftverhibák létezésének. A formális módszerek alkalmazása azonban nem követelmény ma még minden szoftverfejlesztési garanciális szinten (a Common Criteria garanciális követelményei csupán az 5-7 szinteken tartalmaznak félformális vagy formális tervezési követelményeket a rendszerterv és a biztonsági politika számára), így ettől átütő eredményt csak akkor lehetséges majd elvárni, ha az elterjedése szélesebb körben és alacsonyabb garanciális szinten is megvalósulhat.

Köszönetnyilvánítás

Jelen tanulmány elkészítését a PD-109740 számú „IT és hálózati sérülékenységek tovagyűrűző társadalmi-gazdasági hatásai” című projekt támogatta a Nemzeti Kutatási, Fejlesztési és Innovációs Hivatal – NKFIH finanszírozásával.

Irodalomjegyzék

[1] Abbas, Ali - Saddik, Abdulmotaleb El – Miri, Ali. 2005. A State of the Art Security Taxonomy of Internet Security: Threats and Countermeasures. *Computer Science and Engineering* (19) 1: 27-36.

[2] Bányász, Péter – Orbók, Ákos. 2013. A NATO kibervédelmi politikája és kritikus infrastruktúra védelme a közösségi média tükrében. *Hadtudomány* (2013) elektronikus: 188-209.

[3] Basili, Victor R. - Selby, Richard W. – Hutchens David H.. 1986. Experimentation in Software Engineering. *IEEE Transactions on Software Engineering* (SE-12) 7: 733-743

[4] Bezroukov, Nikolai. 1999. Open Source Software Development as a Special Type of Academic Research (Critique of Vulgar Raymondism). *First Monday* (4) 10: 1

[5] Boehm, Barry W.. 1988. A Spiral Model of Software Development and Enhancement. *IEEE Computer* (21) 5: 61-72

[6] Fang, Jianbin – Varbanescu, Ana Lucia – Sips, Henk. 2011. A Comprehensive Performance Comparison of CUDA and OpenCL. *Proceeding ICPP '11 Proceedings of the 2011 International Conference on Parallel Processing.* USA: IEEE Computer Society

[7] Gyebrovszky, Tamás. 2014. Folyamatos fenyegetések a kibertérben. *Hadmérnök* (IX) 3: 137-153.

[8] Hansman, Simon – Hunt, Ray. 2004. A taxonomy of network and computer attacks. *Computer & Security* (24) 1: 31-43.

[9] Hippel, Eric von. 2001. Innovation by User Communities: Learning from Open-Source Software. *MIT Sloan Management Review* (42) 4: 82-86.

[10] Hippel, Eric von. 2001. Open Source Software: Innovation By and For Users – No Manufacturer Required! http://ebusiness.mit.edu/research/papers/133%20von%20hippel%20OSS%20innovation.pdf (2016. szeptember 25.)

[11] Hofmeister, Christine - Kruchten, Philippe - Nord, Robert L. - Obbink, Henk - Ran, Alexander – America, Pierre. 2005. Generalizing a Model of Software Architecture Design from Five Industrial Approaches. *Proceeding WICSA '05 Proceedings of the 5th Working IEEE/IFIP Conference on Software Architecture, November 6-9, 2005.* USA: IEEE Computer Society

[12] Holmström, H. - Ó Conchúir - E., Ågerfalk - P.J. - Fitzgerald, B.. 2006. Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance. *International Conference on Global Software Engineering (ICGSE2006), Costão do Santinho, Florianópolis, Brazil, October 16-19 2006.* USA: IEEE Computer Society

[13] Holmström, H. - Ó Conchúir, E. - Ågerfalk, P.J. - Fitzgerald, B.. 2006. Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance, *International Conference on Global Software Engineering (ICGSE2006)*, *Costão do Santinho, Florianópolis, Brazil, October 16-19 2006*. USA: IEEE Computer Society

[14] Howard, John D.. 1997. An Analysis of Security Incidents on The Internet 1989 – 1995 A dissertation submitted to the graduate school in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Engineering and Public Policy, Carnegie Mellon University. Pittsburgh, Pennsylvania, USA: Carnegie Mellon University

[15] Illési, Zsolt. 2012. *Információtechnológiai környezetben elkövetett támadások és bűncselekmények kriminál technikai vizsgálata.* Budapest: Nemzeti Közszolgálati Egyetem, Hadtudományi és Honvédtisztképző Kar Katonai Műszaki Doktori Iskola

[16] Kogut, Bruce – Metiu, Anca. 2001. Open Source Software Development and Distributed Innovation. *Oxford Review of Economic Policy* (17) 2: 248-264

[17] Kovács, László. 2012. Európai országok kiberbiztonsági politikáinak és stratégiáinak összehasonlító elemzése I. *Hadmérnök (*VII) 2: 302-311.

[18] Lamersdorf, Ansgar – Münch, Jürgen. 2010. Studying the Impact of Global Software Development Characteristics on Project Goals: A Causal Model. *The Open Software Engineering Journal (TOSEJ)* (2010) 4: 2-13.

[19] Landwehr, Carl E. - Bull, Alan R. - McDermott, John P. – Choi, William S.. 1994. A Taxonomy of Computer Program Security Flaws. *Computing Surveys* (26) 3: 211-254

[20] Mockus, Audris - Fielding, Roy T. – Herbsleb, James. 2000. A Case Study of Open Source Software Development: The Apache Server. *Software Engineering, 2000. Proceedings of the 2000 International Conference on 9 June, 2000*. USA: IEEE Computer Society

[21] Muha, Lajos – Krasznay, Csaba. 2014. Az elektronikus információs rendszerek biztonságának menedzselése. Budapest: Nemzeti Közszolgálati Egyetem, Vezető- és Továbbképzési Intézet

[22] Orbók, Ákos. 2015. Rövid áttekintés a Nemzeti Kibervédelmi Intézet megalakulásáról, működéséről és előzményeiről. *Hadmérnök* (X) 4: 247-251

[23] Raymond, Eric S.. 1998. The cathedral and the bazaar. *First Monday* (3) 3: 1

[24] Raymond, Eric S.. 1999. A Response to Nikolai Bezroukov. *First Monday*(4) 11: 1

[25] Schaffer, Kim – Voas, Jeffrey. 2016. What Happened to Formal Methods for Security? *IEEE Computer* (49) 08: 70-79

[26] Thonnard, Olivier – Dacier, Marc. 2008. A framework for attack patterns' discovery in honeynet data. *Digital Investigation* (2008) 5: 128-139

[27] Vasvári, György. 2011. Válogatott tanulmányok a vállalati biztonság témaköréből. Budapest: Információs Társadalomért Alapítvány