Raport ştiinţific si tehnic pentru proiectul *Canal securizat între dispozitivele I/O şi unitatea de procesare pentru extinderea securităţii software (SABOTORE)*

Etapa I – Studii de piaţă interne şi internaţionale, privind securitatea transferului de date (I/O) între un sistem de calcul şi sistemele periferice acestuia şi posibilitatea folosirii platformelor de tip Intel SGX

L1.1 - Raport de cercetare privind tehnologiile şi implementările mecanismelor tehnice de securizare a transferului de date între sistemul de operare şi dispozitivele periferice

Cuprins

1 Introducere

Protejarea datelor personale a fost mereu una dintre cele mai importante componente atunci când vorbim de securitate în mediul online. Spre exemplu, conceptul de virtualizare a fost introdus nu doar pentru a putea utiliza simultan un dispozitiv de calcul, dar și pentru a putea separa datele fiecărui utilizator. Astăzi, într-o lume interconectată, securitatea datelor a devenit chiar și mai importantă prin utilizarea conturilor în tot mai multe platforme online. Proiectul de față își propună să aducă o îmbunătățire a modului în care sunt folosite datele confidențiale din cadrul unei companii, cum ar fi un fișier care conține unul dintre contractele foarte importante ale companiei.

Majoritatea sistemelor de securitate au o bază de încredere („root of trust") în software, spre exemplu unul dintre cele mai folosite protocoale de securitate, Transport Layer Security (TLS), se folosește de existența în sistemele de operare a unei chei publice (am simplificat foarte mult pentru o explicare mai facilă). Această cheie publică va fi folosită ulterior la tot ce înseamnă o comunicare securizată în Internet, iar compromiterea ei duce la compromiterea întregii securități.

Principala motivație pentru care securitatea are ca bază unelte software este dată de ușurința cu care un produs software se dezvoltă față de un produs hardware. Spre exemplu, apariția unui nou protocol criptografic este prima dată implementată în software, și apoi în hardware, unde evident volumul de date care poate fi procesat este mai mare.

Proiectul SABOTORE vine în contextul în care majoritatea calculatoarelor sunt echipate cu echipamente hardware dedicate ce oferă Execuție Protejată („Trusted Execution Environment - TEE), Intel furnizând începând cu anul 2014 procesoare cu suport Intel Software Guard Extensions (SGX) iar procesoarele AMD oferă din 2016 Secure Encrypted Virtualization (SEV)[1][2]. Aceste soluții hardware oferă garanția execuției unui program în contextul unui sistem de operare malițios, putând fi folosite ca baza de încredere.

Mai mult, în domeniul Internet of Thinks (IoT) se poate observa o creștere a numărului de echipamente disponibile, estimările actuale arată că până în anul 2025 vom avea peste 75 de miliarde de astfel de echipamente conectate la Internet [3]. Odată cu această explozie a echipamentelor, costurile de fabricare au scăzut, fiind posibilă construirea de hardware particularizat cu o ușurință mai mare.

SABOTORE își propune să se folosească de aceste noi tehnologii pentru a permite o comunicare sigură. Echipa Universității Politehnica din București împreună cu cei de la CertSign au dezvoltat trei posibile scenarii în care tehnologiile prezentate mai sus pot oferi o securitate ridicată:

1. Pentru a proteja accesul la rețea în mediile de lucru unde acest acces trebuie controlat. Spre exemplu, placa de rețea va trebui să folosească un secret stocat în TEE pentru a se autentifica la rețea.
2. Pentru a folosi la distanță token-ul SABOTORE. Mai concret, am putea pune un astfel de la intrarea într-o imprimantă sau într-un video proiector și să permitem accesul la aceste dispozitive doar de la dispozitive autorizate și autentificate.

3. Pentru a cripta datele de pe dispozitive, locale sau la distanță. Spre exemplu, un stick USB poate fi criptat cu cheia care rezidă în TEE, iar decriptarea se poate face doar folosind același calculator.

Etapele principale în cadrul proiectului sunt:

1. Studii de piață interne și internaționale, privind securitatea transferului de date (I/O) între un sistem de calcul și sistemele periferice acestuia și posibilitatea folosirii platformelor de tip Intel SGX
2. Elaborare cazuri de utilizare și cerințe tehnice
3. Arhitectura de referință și etică
4. Dezvoltare, testare și validare prototipuri
5. Transferul tehnologic al rezultatelor de dezvoltare experimentală și diseminarea rezultatelor

În prezentul raport ne propunem să arătăm care este stadiul de dezvoltarea al tehnologiilor ce vor fi folosite, cum este în prezent securizată comunicarea între două dispozitive și cum pot fi folosite dispozitive externe pentru a realiza autentificarea.

2 Descrierea etapei și a activităților

Obiectivele acestei etape sunt:

1. Integrarea cât mai simplă a SABOTORE în tehnologiile de securitate existente
2. Familiarizarea cu tipurile de atacuri asupra dispozitivelor I/O

Cele 3 activități din Etapa I descrise in cadrul acestui raport sunt:

1. A1.1 Studiu privind stadiul de dezvoltare tehnologică a soluțiilor de securizare a datelor folosind extensii software ale unităților centrale de procesare (CPU)
2. A1.2 Studiu privind vulnerabilitățile sistemelor de operare în procesul de prelucrare a datelor si transferul acestora intre unitățile centrale de procesare și dispozitivele periferice
3. A1.3 Studiu privind tehnologiile si implementările mecanismelor de autentificare a dispozitivelor periferice si de securizarea transferului de date intre acestea si unitatea centrala de procesare (CPU)

Au fost obținute următoarele livrabile:

1. L1.1 Raport de cercetare privind tehnologiile și implementările mecanismelor tehnice de securizare a transferului de date intre sistemul de operare si dispozitivele periferice

3 Stadiul de dezvoltare tehnologică

În cadrul acestui capitol accentul este pus pe metodele deja dezvoltate care sunt corelate cu SABOTORE, atât cele academice dar și cele industriale, care au fost sau sunt deja lansate pe piață.

3.1 *Soluții existente în mediul academic*

Mediile de execuție de încredere vizează reducerea complexitatea proiectării sistemelor sigure și, astfel, există numeroase cercetări în acest domeniu. Soluțiile variază de la hipervizoare sigure

software [6],[7], abordări specifice platformei care valorifică caracteristicile hardware [8], [9] [10] la abordări centrate pe hardware [11], [12], [13], [14].

Problema stabilirii unei căi Input/Output de încredere între dispozitivele externe și TEE-urile este prezentată ca o direcție de cercetare complementară, ale cărei soluții pot fi partiționate în două categorii: abordări exclusiv software și cele care necesită dispozitive hardware separate sau modificări arhitecturale.

În ZHOU, Zongwei și colab. [4], autorii prezintă abordarea bazată pe hipervizoare a izolării periferice. Ei definesc provocările pe care trebuie să le depășească un hipervizor pentru a oferi I / O sigure, cum ar fi izolarea întreruperilor, prevenirea DMA atacurile suprapuse și complexitatea dezvoltării izolate driverele de dispozitiv și chiar sugerează soluții arhitecturale la problema căii de încredere. SGXIO [15] propune o altă soluție folosind un hipervizor, de data aceasta vizând în mod specific Intel SGX tehnologie. De asemenea, necesită un hardware TPM pentru a furniza servicii de boot de încredere pentru legarea enclavei la o mașină. În [5], se implementează un nucleu de separare GPU care adaugă un acces la stratul de control la obiectele GPU.

În ceea ce privește abordările bazate pe hardware, am găsit mai multe soluții specializate. ZTIC [16] este un USB dongle cu propriul său afișaj și butoane, concepute pentru comunicarea cu un server prin canal criptat. Bumpy [17] folosește dispozitive USB pentru criptarea datelor de intrare (tastatură) și afișarea de încredere pe un sistem mobil cu TEE exclusiv [8].

O altă lucrare, BASTION-SGX [18], folosește protocolul Bluetooth și modificări ale firmware-ului controlerului pentru activare I / O de încredere cu dispozitivele wireless care îl utilizează. Deși acest lucru nu utilizează niciun hardware extern (gazdele Bluetooth moderne) au cipuri dedicate cu firmware reprogramabile datorate la complexitatea protocolului), necesită sprijin din partea producătorul controlorului (firmware-urile sunt deseori surse închise).

În ProximiTEE [19], autorii vorbesc despre utilizarea unui dispozitiv extern pentru a demonstra execuția codului pe un Enclavă SGX în apropiere (aceeași platformă pe care este conectată) folosind limitare de latență. De asemenea, vin cu ideea că un astfel de dispozitiv ar putea fi utilizat pentru a facilita un canal de Input / Output de încredere prin utilizarea unui scheme de aprovizionare în care utilizatorul pornește a un kernel minim pentru a schimba chei asimetrice și a stabili o rădăcină de încredere.

3.2 Produse existente în mediul privat

În mediul privat cercetarea noastră nu a identificat foarte multe produse comercializate, asta probabil datorită faptului că tehnologia SGX este relativ nouă, iar puterea de procesare a dispozitivelor IoT este încă destul de mică pentru a putea efectua calcule criptografice complexe. Totuși există foarte multe patente care folosesc tehnologiile menționate anterior, iar în această secțiune le vom prezenta pe cele mai relevante.

Compania Fortanix este una dintre puținele care are produse ce folosesc tehnologia SGX [20]. Unul dintre cele 3 produse prezentate, denumit Self-Defending Key Management Service, poate fi folosit pentru a gestiona într-un mod sigur cheile sau certificatele folosite de protocoalele care asigura

comunicare sigura. Mai mult, compania pune la dispoziție și un API prin care se pot genera aceste chei în interiorul enclavei (denumirea dată de Intel pentru TEE).

Unul dintre cele mai importante patente referitoare la raportul curent este „Technologies for secure device configuration and management" [23]. În acest patent Intel descrie procesul prin care o aplicație care rulează în mod sigur, spre exemplu într-o enclavă, poate bloca un dispozitiv de Intrare/Ieșire (Input/Output – I/O) astfel încât toată informația primită de acel dispozitiv să fie disponibilă doar programului. Pentru a putea face această trecere este nevoie de un monitor care să primească comenzi de la programul din enclavă și care să aibă acces unic la sistemul de configurare al dispozitivului I/O, permițând astfel o blocare globală. Important de menționat este că programul în execuție care cere acces unic la dispozitivul I/O poate să fie și el autentificat, permițând astfel un acces al controlului mai granulat și mai sigur.

Compania Symantec propune folosirea mediilor de execuție de încredere pentru analiza traficului de rețea în patentul „Decrypting network traffic on a middlebox device using a trusted execution environment„[22]. Mediul de execuție de încredere se află în dispozitivul de rețea, iar principiul de funcționare este următorul: un program executat de către client realizează în mod normal un canal de comunicare criptat direct cu serverul de la care dorește să transfere date. Datorită acestui canal, dispozitivele de rețea nu pot analiza datele, ele fiind criptate. În patentul propus de Symantec, canalul de comunicare este divizat în două, permițând interpunerea unui dispozitiv de rețea. Totuși, pentru a menține securitatea, dispozitivul de rețea este echipat cu un mediu de execuție de încredere, permițând astfel clientul să realizeze mai întâi autentificarea acestuia și apoi trimițând datele către el.

Compania MITRE propune crearea unui tunel securizat între un dispozitiv Universal Serial Bus (USB) și o mașină virtuală [24]. Dispozitivul propus poate avea două moduri de operare, de încredere și transparent. Atunci când rulează în modul transparent, dispozitivele I/O care sunt conectate trimit date direct către sistemul de operare, fără a oferi nici o protecție. În modul de încredere, datele primite de la orice dispozitiv extern sunt criptate și ele vor putea fi descifrate doar de către mașina virtuală țintă.

Microsoft propune o validare a datelor de intrare folosind un mediu de execuție de încredere în patentul denumit „Secure policy ingestion into trusted execution environments" [25]. Datele nesigure dintr-un calculator, spre exemplu un fișier PDF, vor fi afișate utilizatorului prin intermediul unui dispozitiv I/O sigur, iar utilizatorul le va putea valida. Dispozitivul I/O are un canal de comunicare sigur direct cu mediul de execuție de încredere, și va trimite validarea (sau nu) primită de la utilizator direct către programul sigur. Acesta din urmă, în funcție de răspunsul primit, va adăuga o semnătură proprie datelor, ele fiind astfel marcate ca sigure și vor putea fi prelucrare de către altcineva într-un mod sigur.

4 Autentificarea dispozitivelor

În această secțiune vom prezenta principalele metode prin care se poate realiza autentificarea (mutuală) între două dispozitive. Reamintim că în proiectul SABOTORE este necesară crearea unui canal sigur între dispozitivul IoT și o enclavă SGX, iar, așa cum vom vedea în secțiunea următoare,

acest canal nu poate fi realizat fără autentificarea celor două entități. Există 2 metode principale de autentificare: (1) folosind criptografia simetrica, prin utilizarea de chei partajate anterior sau (2) folosind criptografia asimetrica, prin utilizarea de perechi de chei publice/private. În continuare vom detalia aceste metode.

Atunci când se folosește o cheie partajată anterior, cele două entități trebuie să aibă posibilitatea de a trimite un secret pe un canal deja securizat. Există mai multe metode de a realiza acest lucru: (a) cheia partajata este scrisă deja în codul programului care va fi executat, (b) se folosește un alt canal prin care se poate trimite un secret (spre exemplu se trimite o parola prin SMS), sau (c) cheia este introdusă manual de către administrator înainte de a se negocia canalul securizat.

Atunci când se folosesc chei publice/private pentru autentificare, unul dintre cei doi participanți la realizarea conexiunii trebuie să dețină cheia publică a celuilalt. Și aici, ca și în cazul precedent, există două metode generice folosite: Public Key Infrastructure (PKI) sau Pretty Good Privacy (PGP).

În cazul PKI cheile publice/private sunt salvate sub forma unui certificat digital împreună cu alte informații despre deținătorul acestora, spre exemplu numele serverului. Acest certificat poate fi trimis de la server către client, cheia publică va fi extrasă și folosită pentru autentificare. Totuși, această schemă funcționează doar dacă se poate garanta autenticitatea certificatului primit, pentru aceasta este nevoie de o Autoritate de Certificare, iar fiecare client trebuie să aibă deja cheia publică a acestei autorități. Apoi, certificatul serverului este validat de către Autoritate (prin diverse mijloace cum ar fi Domain Name System (DNS)), și se adaugă o semnătură din partea autorității pe certificatul serverului. Cu aceasta semnătură clientul poate valida și folosi certificatul primit.

În cazul protocolului PGP, autenticitatea cheilor publice/private se realizează prin transferul direct între două entități. Spre exemplu, dacă Alice vrea să vorbească cu Bob, cei doi se pot întâlni și realiza un transfer direct de chei. Dacă, apoi Bob se întâlnește cu Trudy, el va face schimb de chei cu el. Prin tranzitivitate, Trudy are încredere în Bob, Bob în Alice, și astfel Trudy poate autentifica cheia lui Alice. În felul acesta între utilizatorii rețelei PGP se poate crea o „pânză de păianjen” cu ajutorul căreia utilizatorii se pot autentifica unul pe celălalt.

În patentul denumit „Techniques for remote SGX enclave authentication”, Intel detaliază metodele care pot fi folosite pentru autentificarea la distanță a unei enclave [21]. Orice program poate dovedi că este executat în interiorul SGX prin generarea unei perechi de chei public-private, obținerea unui certificat care va conține doar cheia privată și care va fi apoi semnat de către procesorul Intel pe care rulează. Semnătura procesorului poate fi verificată prin validarea de către un serviciu de atestare. Acest proces de atestare la distanță poate fi folosit de către o altă entitate, spre exemplu un dispozitiv IoT, și prin autentificare cu cheia publică aflată în certificat se poate crea un canal de comunicare securizat cu programul care se execută în enclavă.

5 Securitatea canalelor de comunicare

În această secțiune sunt prezentate sumar principalele mecanisme de crearea a unui canal de comunicare securizat între două sau mai multe entități.

Cel mai important protocol pentru securizarea unui canal de date este Diffie-Hellman (DH), publicat în anul 1976 în jurnalul IEEE Transactions on Information Theory. Acest protocol răspunde la una dintre cele mai grele întrebări ale vremii, cum pot Alice și Bob comunica sigur atunci când Trudy are acces la toată comunicația lor. Fără a intra în detaliile matematice ale protocolului, versiunea inițială a protocolului folosește problema algoritmilor discreți, și comunicând public p (prim foarte mare) și g (1 < g < p), Alice și Bob pot genera o cheie secretă.

Totuși, protocolul DH poate fi atacat dacă nu există autentificare între Alice și Bob deoarece nici unul dintre cei doi nu poate garanta că informațiile primite (ex., p și g) provin de la celălalt, astfel Trudy ar putea realiza un canal de comunicare securizat cu Alice, și altul cu Bob, interpunându-se astfel între cei doi. Soluția pentru această problema este simpla, Alice și Bob trebuie să poată realiza autentificarea primului mesaj, spre exemplu printr-o parola pe care doar cei doi o cunosc sau folosind certificate digitale.

Protocolul Transport Layer Security (TLS), versiunea precedentă este denumită Secure Sockets Layer (SSL), este unul dintre cele mai cunoscute protocoale de comunicare securizată. Protocolul folosește certificate și Autorități de Certificare („Certificate Authorities" – CA) pentru autentificarea entităților. Cel mai cunoscut caz este acela în care un client dorește să transfere date de la un server. Pentru aceasta, primul pas pe care îl face clientul este să descarce certificatul serverului pentru a realiza autentificarea. Un certificat conține cheia publică a serverului, și o semnătură asupra autenticității acesteia din partea unui CA. Clientul poate verifica veridicitatea semnăturii CA-ului deoarece el deține cheia publică a mai multor CA-uri. După verificarea CA-ului, se poate verifica semnătura acesteia din certificat, obținându-se astfel cheia publică a serverului, deci o metodă de autentificare. După acest pas, TLS folosește protocolul DH pentru a genera o cheie de criptare cu care se poate proteja canalul de comunicare.

Un alt protocol important dezvoltat de către Internet Engineering Task Force (IETF) este IP Security (IPSec). Fără a detalia toate modurile de funcționare, menționăm că protocolul are două nivele importante de securitate: Encapsulating Security Payload (ESP) – poate fi folosit pentru a proteja toate datele din pachetele de rețea, și Authentication Header (AH) – garantează doar integritatea datelor transmise, nu și confidențialitatea acestora. De asemenea, protocolul IPSec poate funcționa în modul tunel sau în modul transport, principala diferență fiind că atunci când se folosește modul tunel este protejat și antetul IP, respectiv sursa și destinația comunicării.

6 Concluzii

În aceasta etapa au fost analizate principalele unelte ce pot fi folosite pentru îmbunătățirea prototipului de la care pornește SABOTORE, cum ar fi metodele actuale de autentificare sau protocoalele pentru securizarea unui canal de comunicare. De asemenea, au fost identificate principalele soluții similare atât din mediul academic cât și din meniu privat.

Site-ul proiectul este în curs de dezvoltate și va fi actualizat cu informații privind progresul proiectului.

7 Referințe Bibliografice

[1]	https://software.intel.com/content/www/us/en/develop/topics/software-guard-extensions.html
[2]	https://developer.amd.com/sev/
[3]	https://www.cisco.com/c/en/us/solutions/internet-of-things/future-of-iot.html
[4]	Z. Zhou, V. D. Gligor, J. Newsome, and J. M. McCune, "Building verifiable trusted path on commodity x86 computers," in 2012 IEEE Symposium on Security and Privacy. IEEE, 2012, pp. 616–630.
[5]	M. Yu, V. D. Gligor, and Z. Zhou, "Trusted display on untrusted commodity platforms," in Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. ACM, 2015, pp. 989– 1003.
[6]	J. Criswell, N. Dautenhahn, and V. Adve, "Virtual ghost: Protecting applications from hostile operating systems," in ACM SIGPLAN Notices, vol. 49, no. 4. ACM, 2014, pp. 81–96.
[7]	O. S. Hofmann, S. Kim, A. M. Dunn, M. Z. Lee, and E. Witchel, "Inktag: Secure applications on an untrusted operating system," in ACM SIGPLAN Notices, vol. 48, no. 4. ACM, 2013, pp. 265–278.
[8]	J. M. McCune, B. J. Parno, A. Perrig, M. K. Reiter, and H. Isozaki, "Flicker: An execution infrastructure for tcb minimization," in ACM SIGOPS Operating Systems Review, vol. 42, no. 4. ACM, 2008, pp. 315–328.
[9]	A. Vasudevan, J. McCune, J. Newsome, A. Perrig, and L. Van Doorn, "Carma: A hardware tamper-resistant isolated execution environment on commodity x86 platforms," in Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security. ACM, 2012, pp. 48–49.
[10]	A. M. Azab, P. Ning, and X. Zhang, "Sice: a hardware-level strongly isolated computing environment for x86 multi-core platforms," in Proceedings of the 18th ACM conference on Computer and communications security. ACM, 2011, pp. 375–388.
[11]	D. Champagne and R. B. Lee, "Scalable architectural support for trusted software," in HPCA-16 2010 The Sixteenth International Symposium on High-Performance Computer Architecture. IEEE, 2010, pp. 1–12.
[12]	G. E. Suh, D. Clarke, B. Gassend, M. Van Dijk, and S. Devadas, "Aegis: architecture for tamper-evident and tamper-resistant processing," in ACM International Conference on Supercomputing 25th Anniversary Volume. ACM, 2014, pp. 357–368.
[13]	E. Owusu, J. Guajardo, J. McCune, J. Newsome, A. Perrig, and A. Vasudevan, "Oasis: On achieving a sanctuary for integrity and secrecy on untrusted platforms," in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, 2013, pp. 13–24.
[14]	F. McKeen, I. Alexandrovich, A. Berenzon, C. V. Rozas, H. Shafi, V. Shanbhogue, and U. R. Savagaonkar, "Innovative instructions and software model for isolated execution." Hasp@ isca, vol. 10, no. 1, 2013
[15]	S. Weiser and M. Werner, "Sgxio: Generic trusted i/o path for intel sgx," in Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy. ACM, 2017, pp. 261–268.
[16]	T. Weigold, T. Kramp, R. Hermann, F. Horing, P. Buhler, and ¨ M. Baentsch, "The zurich trusted information channel–an efficient defence against man-in-the-middle and malicious software attacks," in International Conference on Trusted Computing. Springer, 2008, pp. 75–91

[17]	J. M. M. A. Perrig and M. K. Reiter, “Safe passage for passwords and other sensitive data,” in Proceeding of the 16th Annual Network and Distributed System Security Symposium, 2009.
[18]	T. Peters, R. Lal, S. Varadarajan, P. Pappachan, and D. Kotz, “Bastionsgx: bluetooth and architectural support for trusted i/o on sgx,” in Proceedings of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy. ACM, 2018, p. 3
[19]	A. Dhar, I. Puddu, K. Kostianen, and S. Capkun, “Proximitee: Hardened ˇ sgx attestation and trusted path through proximity verification,” 2018
[20]	https://fortanix.com/
[21]	https://patents.google.com/patent/US20180241572A1
[22]	https://patents.google.com/patent/WO2019156741A1
[23]	https://patents.google.com/patent/US20190311123A1
[24]	https://patents.google.com/patent/US20130179685
[25]	https://patents.google.com/patent/US20190268161A1/
[26]	
[27]	