Zaštita *embedded* sistema od neautorizovane modifikacije softvera i praktična primena na set-top-boksu

Bojan Lazić, Nebojša Mirić, Tihomir Anđelić, Velibor Mihić, RT-RK

***Apstrakt*— Kao sto je slučaj sa softverom za desktop računare, *embedded* sistemi takođe imaju softver koji je podložan napadima i izmenama od strane neautorizovanih lica. U ovom radu biće ukratko objašnjena potreba za zaštitom sistema, kao i posledice slabo osmišljenih mehanizama zaštite. Dat je pregled elemenata kriptografije koji su najčešće u upotrebi, enkripcije/dekripcije, *hash* funkcija, digitalnog potpisa. Opisano je kako se kriptografske metode mogu upotrebiti u zaštiti softvera *embedded* sistema. Dat je primer primene predloženih metoda zaštite kod set-top-boks uređaja, u procesu sigurnog podizanja sistema i ažuriranja softvera.**

***Ključne reči*— *embedded* sistem, kriptografija, ažuriranje softvera, set-top-boks**

I. UVOD

Pojam *embedded* sistema se odnosi na elektronske sisteme čiji je centralni deo mikroprocesor (bilo u okviru mikrokontrolera ili zasebno) koji je povezan sa različitim periferijama (GPIO pinovi, tajmeri, AD/DA konvertori, UART, SPI) i čiji je zadatak da nadzire ili upravlja delom većeg pripadajućeg sistema. Prisutni su u auto, avio, vojnoj, telekomunikacionoj industriji i deo su uređaja potrošačke elektronike, televizora, kamera, mobilnih telefona itd. Ovi uređaji su često u stanju da komuniciraju jedni sa drugima ili sa udaljenim serverima.

Softver koji se izvršava unutar *embedded* sistema je, kao i drugi tipovi softvera, podložan napadima hakera. Bezbednosnim aspektima razvoja softvera se mora posvetiti posebna pažnja, kako bi se sprečile zloupotrebe. Prema istraživanju [1] CVE (*Common Vulnerabilities and Exposures*) baze podataka, čiji je pokrovitelj američka vlada (*Department of Homeland Security*), 3 glavne slabe tačke *embedded* sistema su: greške u programiranju, loše zaštićene internet veze i slabe metode kontrole pristupa i autentifikacije. Greške u programiranju izazivaju promenu kontrole toka izvršavanja programa. Primer je *buffer overflow* napad uzrokovan lošim parsiranjem ulaznih podataka. Mane u *web* interfejsu uređaja mogu se zloupotrebiti za ubacivanje neželjenog softvera. Mehanizmi kontrole pristupa se mogu zaobići ukoliko se koriste šifre i ključevi koji se retko menjaju i koji su slabo zaštićeni. Posledice hakerskih napada mogu biti onemogućavanje usluga (*Denial of Service*), izvršavanje malicioznog softvera, pribavljanje zaštićenih podataka, nedozvoljeno korišćenje uređaja.

U prošlosti su *embedded* uređaji isporučivani bez mogućnosti naknadnih izmena hardvera i softvera. Danas je softver odvojen proizvod, a tehnologija fleš memorije omogućuje njegovo lako ažuriranje. Prednosti daljinskog ažuriranja softvera ispravka *bug*-ova koji nisu uočeni tokom razvoja, kao i naknadno omogućavanje dodatnih funkcionalnosti. Postupak ažuriranja mora biti robusan, u smislu da sistem mora da nastavi da funkcioniše čak i u slučaju da ažuriranje nije uspelo.

Za inicijalni softver koji se nalazi u tek isporučenim uređajima se moze smatrati da je autentičan, jer je proces programiranja sproveden u sigurnom fabričkom okruženju. U ostalim slučajevima, kada je nova verzija softvera razvijena, proizvođač dostavlja binarni fajl koji se šalje preko nezaštićenih komunikacionih kanala. Da bi se sistem dodatno ojačao, uvode se se mehanizmi sigurnog podizanja sistema (*secure boot*) i sigurnog ažuriranja sistema (*secure update*).

Osnovni bezbednosni aspekti koji se moraju razmatrati pri razvoju softvera su poverljivost, autentičnost i integritet [2]. Poverljivost podrazumeva da intelektualni sadržaj (u ovom slučaju programski kod, ključevi, dokumenti) nije dostupan neautorizovanim licima. Autentičnost znači da proizvod zaista potiče od autorizovanog lica i da će imati svoju nameravanu funkciju. Proverom autentičnosti se potvrđuje izvor, ali i validnost uređaja u koji se novi softver upisuje. Integritet podrazumeva da su svi delovi softvera onakvi kao sto je proizvođač obezbedio, tj. da nijedan deo nije modifikovan.

Kvalitet zaštite sistema zavisi od stepena sigurnosti kriptografskog algoritma, koji se sastoje od kombinacije osnovnih elemenata kriptografije. Pregled kriptografskih elemenata i njihovi pripadajući bezbednosni aspekti prikazani su u tabeli I.

TABELA I
BEZBEDNOSNI ATRIBUTI KRIPTOGRAFSKIH ELEMENATA

	Atribut		
	Tajnost	Autentičnost	Integritet
Enkripcija/dekripcija	✓	✓	
Hash, digest			✓
*MAC**		✓	✓
Digitalni potpis		✓	✓

(**Message Authentication Code* – autentifikacioni kod poruke)

Kriptografija je oblast koja se konstantno razvija. Pronalaze se novi algoritmi, dok upotreba starih postaje nebezbedna. U narednom poglavlju biće prikazan pregled osnovnih pojmova u kriptografiji.

II. KRIPTOGRAFSKE MERE ZAŠTITE

U procesu enkripcije, originalni sadržaj se konvertuje u proizvoljni broj bita uz pomoć ključa i određenog algoritma kriptovanja. Algoritam je takav da se pri svakoj ponovnoj upotrebi proizvodi različiti izlaz u zavisnosti od vrednosti

Bojan Lazić – Istraživačko-razvojni Institut RT-RK, Beograd, Bulevar Milutina Milankovića 19b (e-mail:bojan.lazic@rt-rk.com).

Nebojša Mirić – Istraživačko-razvojni Institut RT-RK, Beograd, Bulevar Milutina Milankovića 19b (e-mail: nebojsa.miric@rt-rk.com).

Tihomir Anđelić – Istraživačko-razvojni Institut RT-RK, Novi Sad, Narodnog fronta 23a (e-mail: tihomir.andjelic@rt-rk.com).

Velibor Mihić – Istraživačko-razvojni Institut RT-RK, Novi Sad, Narodnog fronta 23a (e-mail: velibor.mihic@rt-rk.com).

ključa. Na prijemnoj strani se uz pomoć istog algoritma i odgovarajućeg ključa kriptovana poruka konvertuje u originalnu. U zavisnosti od broja ključeva koji se koriste u procesu enkripcije i dekripcije razlikuju se simetrična i asimetrična kriptografija. Kod simetrične kriptografije se koristi jedan isti (tajni) ključ i za enkripciju i za dekripciju. Osnovni problem u ovom slučaju je obezbediti sigurnu vezu za prenos tajnog ključa kako bi bio dostupan na obe strane. Algoritmi koji se najviše koriste su AES, DES, 3DES, blowfish, RC4, RC6. Prema istraživanjima [3], AES algoritam ima bolje performanse od ostalih. To se pre svega odnosi na brzinu izvršavanja, otpornost na poznate metode napada i potrošnju snage.

Kod asimetrične kriptografije koristi se par ključeva, pri čemu se jedan ključ koristi za enkripciju, a drugi za dekripciju. Postoje dva oblika asimetrične kriptografije, PKE (*Public Key Encryption*) i digitalni potpis. U prvom slučaju, ključ koji se koristi za enkripciju se naziva javni ključ, dok je ključ za dekripciju privatni i čuva se u tajnosti. Ovaj oblik asimetrične kriptografije se koristi kada je namera da svako može da enkriptuje neki sadržaj, a enkriptovani sadržaj može se dekriptovati samo ako na prijemnoj strani postoji privatni ključ. Najpoznatiji algoritmi sa asimetričnim ključem su RSA, DSA, ECC, Diffie-Hellman, ElGamal. RSA je najrasprostranjeniji trenutno, međutim ECC algoritam pruža isti nivo zaštite u odnosu na ostale, upotrebom ključa manje dužine [4]. Osobine ECC algoritma su brže vreme izračunavanja, manje koričćene memorije što je bitan faktor u resursno ograničenom okruženju.

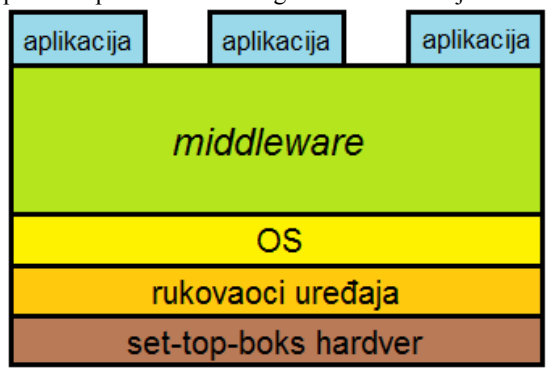
Kriptografske *hash* funkcije su javno dostupne, determinističke funkcije koje se mogu izračunati nad proizvoljnim ulaznim podacima. da bi se kao rezultat dobila hash vrednost, tj. string fiksne dužine. Proces generisanja *hash*-a treba da bude ireverzibilan, tj. da se originalna poruka ne može rekonstruisati na osnovu hash-a. Verovatnoća da dve poruke imaju isti *hash* (tzv. kolizija) treba biti što manja. Parametri koji se razmatraju prilikom poređenja različitih *hash* funkcija su vreme izračunavanja, lavinski efekat i verovatnoća kolizije. Najpoznatije *hash* funkcije su MD2, MD4, MD5, SHA0, SHA1, SHA2, SHA3.

Pomoću *hash* funkcija, može se utvrditi integritet poruke, ali ne i autentičnost. *HMAC* (*Hash-based Message authentication code*) je blok podataka fiksne veličine generisan pomoću tajnog ključa, koji je poznat na predajnoj strani i na prijemnoj strani. Na prijemnoj strani se pomoću tajnog ključa verifikuje *HMAC*, čime se istovremeno potvrđuje integritet, ali i poreklo, tj. autentičnost.

Ukoliko ne postoji potpuno poverenje između predajne i prijemne strane, metoda digitalnog potpisa prevazilazi taj problem korišćenjem privatnog ključa za enkripciju *hash*-a. Na prijemnoj strani se pomoću javnog ključa dekriptuje sadržaj *hash*-a i poredi se sa izračunatom hash vrednošću poruke. Ako je rezultat upoređivanja potvrdan, potpis se smatra autentičnim, u protivnom se smatra da je potpis lažan ili da je poruka modifikovana.

III. Softver set-top-boks uređaja

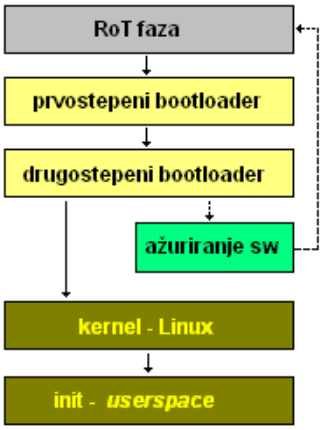
Set-top-boks uređaji služe za prijem TV signala (kablovskog, zemaljskog ili satelitskog), njegovu obradu i prosleđivanje na monitor za prikaz slike i zvuka. Na slici 1 je prikazan primer softverskog steka STB uređaja.



Sl. 1. Struktura softvera STB uređaja

Uloga rukovalaca uređajima (*device drivers*) je da posreduju u komunikaciji između hardvera i operativnog sistema. Uloga operativnog sistema je da obezbedi multitasking zahteva DTV (digitalna televizija) softvera, upravljanje procesima, memorijom i sistemskim resursima. Najrasprostranjeniji OS kod STB uređaja je Linux. Najveći sloj u steku je *middleware*, koji obezbeđuje usluge višeg nivoa i omogućava izvršavanje naprednih DTV aplikacija. Može se smatrati ekstenzijom operativnog sistema.

Pre operativnog sistema, obično se izvršava poseban program *bootloader*, koji se može izvršavati u više faza. Sekvenca izvršavanja programa pri podizanju sistema prikazana je na sl. 2.



Sl. 2. Faze pri podizanju sistema

Po dobijanju napajanja izvršava se kod od poverenja (*Root of Trust* - RoT). Ovaj deo koda se ne ažurira, nalazi se u postojanom delu memorije (ROM, fleš) na dobro poznatoj lokaciji. Očitavaju se vrednosti OTP (*one time programmable* – moguće programiranje samo jedanput) registara i *bootstrap* pinova i u zavisnosti od njihovih vrednosti određuje se dalji tok izvršavanja. Sledeći se izvršava prvostepeni *bootloader* koji se takođe može nalaziti u ROM memoriji. Zatim se u eksternu RAM memoriju učitava drugostepeni *bootloader*, u kome se mapira fizička memorija, inicijalizuju mrežni interfejsi, UART. Proverava se vrednost flega koji daje informaciju o tome da li postoji novi softver za ažuriranje. Ako ne postoji, proverava se validnost *image*-a (pod *image*-om se podrazumeva binarni fajl koji se sastoji iz izvršnog fajla aplikacije sa pridodatim zaglavljem i potpisom i ovaj pojam će se koristiti u daljem

tekstu) operativnog sistema i podiže se Linux. Ako postoji novi softver za ažuriranje, prelazi se na izvršavanje posebne aplikacije za tu svrhu (*Secure Software Updater* - SSU). Uloga SSU-a je ažuriranje softvera: Linux-a, fajl sistema i aplikacije. SSU se moze smatrati ekstenzijom bootloader-a. Kanali za preuzimanje novog softvera su OAD (*Over-the Air-Download*), TCP/IP i USB. Za OAD metod je karakteristično da se binarni fajl koji predstavlja novi softver ubaci u pakete koji se šalju u okviru transportnog toka, kojim se paketski prenose i podaci vezani za TV program (audio/video, dodatne informacije o kanalu). Transportni tok je formiran prema MPEG (*Moving Picture Experts Group*) standardu za kompresiju i transmisiju audio i video signala.

Glavne funkcionalnosti SSU-a su:

- Mogućnost preuzimanja nove verzije softvera
- Mogućnost da se upotrebom kriptografskih algoritama utvrdi da je softver za preuzimanje korektan
- Mogućnost pohranjivanja preuzetog binarnog fajla u fleš memoriju
- Mogućnost startovanja *bootloader*-a (reset)

U narednom poglavlju biće dat predlog kako se elementi kriptografije predstavljeni u drugom poglavlju mogu primeniti za zaštitu softvera set-top-boks uređaja.

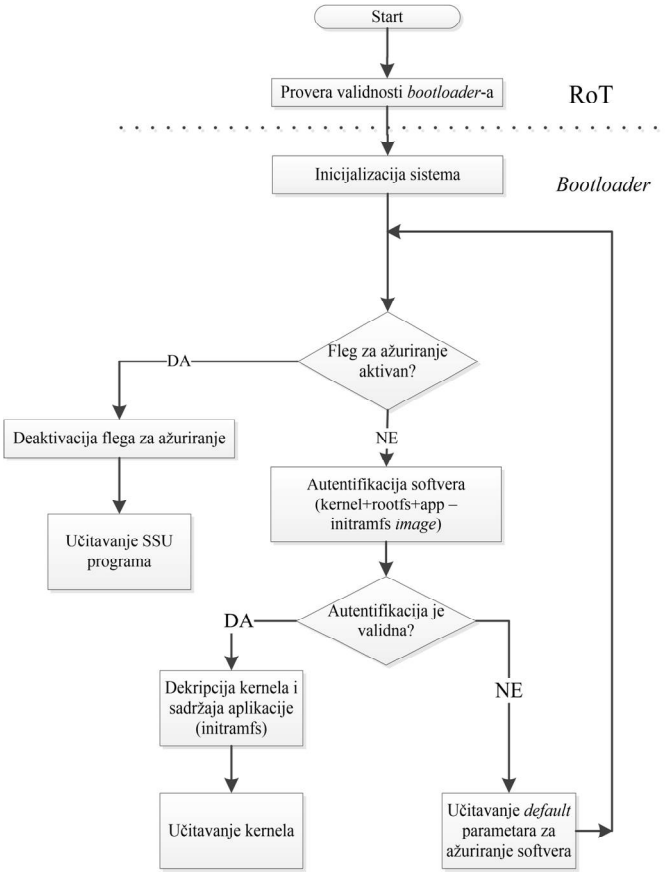
IV. SIGURNOSNI ASPEKTI PODIZANJA SISTEMA I AŽURIRANJA SOFTVERA SET-TOP-BOKSA

Polazna tačka svake sigurne softverske platforme je kreiranje programskog koda bez *bug*-ova i *malware*-a (maliciozni softver koji se dodaje sa ciljem da spreči normalan tok izvršavanja programa). Kada je kod spreman za puštanje u proizvodnju, binarnim podacima koji predstavljaju kompajliran i povezan kod, mogu se dodati zaglavlje i potpis (slika 3).

Sl. 3. Struktura *image*-a spremnog za isporuku

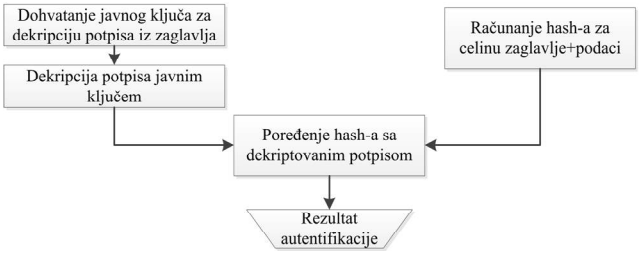
U zaglavlju se nalaze podaci o verziji softvera, verziji hardvera za koju je softver namenjen, kao i javni ključ za dekripciju potpisa. Korisni podaci enkriptovani su tajnim ključem koji je generisan na osnovu identifikacionog broja familije procesora i isti je za celu procesorsku familiju. Ključ se generiše tajnim algoritmom koji je poznat samo proizvođaču. Radi se o algoritmu simetrične enkripcije i u softveru set-top-boksa može se pozivom odgovarajuće funkcije izvršiti dekripcija dela *image*-a sa korisnim podacima. Digitalni potpis enkriptovan je privatnim ključem koji je poznat isključivo proizvođaču i čuva se i sigurnoj bazi podataka.

Na slici 4 prikazan je postupak sigurnog podizanja sistema. Ovaj postupak garantuje da se neautentifikovani softver neće izvršavati.



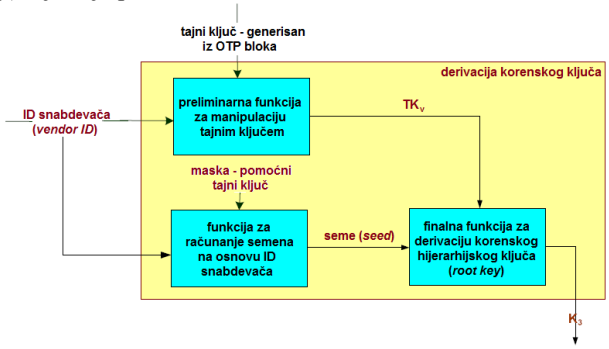
Sl. 4. Postupak sigurnog podizanja sistema (*secure boot*)

Autentifikacija softvera se vrši proverom digitalnog potpisa pomoću javnog ključa koji se nalazi u zaglavlju *image*-a. Dijagram provere potpisa je prikazan na slici 5.



Sl. 5. Postupak provere potpisa

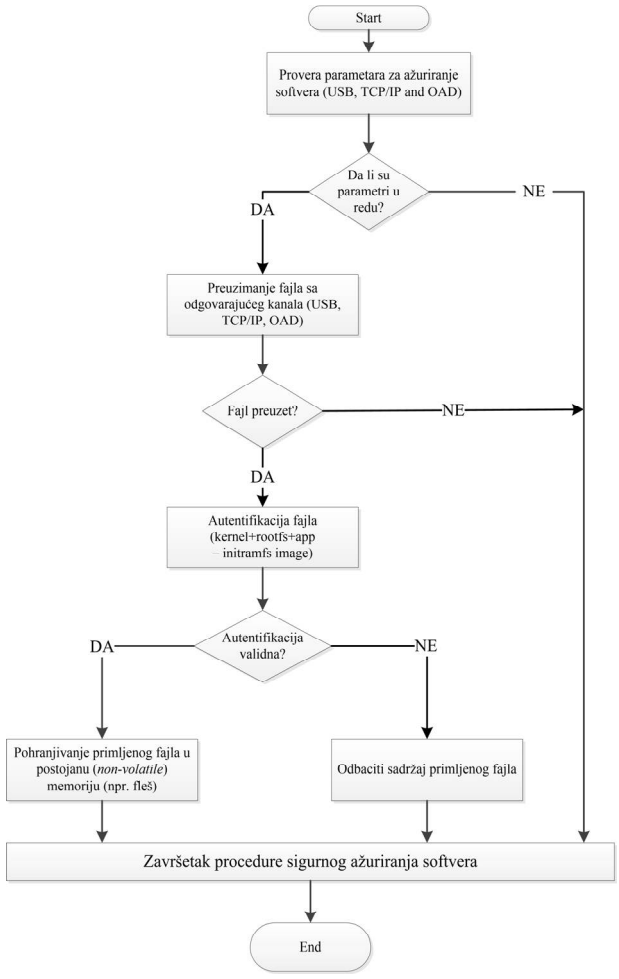
Jedno od mogućih rešenja za generisanje tajnog ključa je *key ladder* mehanizam[5]. Proces generisanja tajnog (*root key*) ključa je prikazan na slici 6.



Sl. 6. Postupak generisanja tajnog korenskog ključa

Rezultat procedure prikazane na slici 6 je ključ K_3. On je dobijen na osnovu vrednosti tajnog ključa koji se računa na osnovu podataka iz OTP memorije, zatim na osnovu podataka o snabdevaču (Vendor ID), kao i pozivanjem funkcija čija se implementacija takođe čuva u tajnosti.

Pomoću *key ladder* mehanizma kreira se hijerarhija ključeva (K_3, K_2, K_1) sa ciljem da se dobije konačni ključ K_0, koji centralni procesor ima pravo da koristi za dekripciju sadržaja od značaja. Ključ K_3 se koristi za dekripciju ključa K_2, K_2 se koristi za dekripciju K_1, dok se K_1 koristi za dekripciju ključa K_0. U slučaju SSU mehanizma ažuriranja K_0 se koristi za dekripciju *initramfs image*-a. Initramfs je u obliku cpio ("*copy in and out*") arhive. Nakon dekripcije i dekompresije *initramfs* fajla, kernel se učitava u RAM, fajl sistem se podiže u korenskom (*root*) folderu, nakon čega je *init* proces spreman za izvršavanje.



Sl. 7. Mehanizam ažuriranja softvera

Prilikom izvršavanja aplikacija na set-top-boksu, periodično se vrši provera da li postoji novi *image*-a za preuzimanje. Npr. OAD metode ažuriranja u tu svrhu koristi monitoring PSI tabela. PSI (*Program Specific Information*) tabele su deo transportnog toka za prenos podataka u digitalnoj televiziji.

Na osnovu odgovarajućeg PID-a (*Packet Identifier* – identifikacioni broj paketa) koji je definisan MPEG standardom, određuje se koji se tačno podaci u transportnom toku odnose na fajl za preuzimanje. Ako postoji fajl za preuzimanje, na utvrđenu lokaciju se upisuje fleg koji pruža informaciju o postojanju novog *image*-a. Zatim se vrši reset i izvršavaju se programski moduli kao na slici 2. U drugostepenom *bootloader*-u se izvršava provera flega i ako je fleg aktivan, deaktivira se i prelazi se na izvršava SSU. Na slici 7 je prikazan mehanizam preuzimanja i provere nove aplikacije za preuzimanje.

V. Zaključak

Cilj rada je bio da se, nakon pregleda postojećih kriptografskih mera zaštite, predoči jedno moguće rešenje za zaštitu softvera kombinacijom opisanih metoda. Dato rešenje se može iskoristiti kao polazna tačka za dizajn softvera kod set-top-boks uređaja, ali i kod sličnih uređaja koji zahtevaju česta ažuriranja.

Zahvalnica

Ovaj rad je delimično finansiran od strane Ministarstva za prosvetu, nauku i tehnološki razvoj Republike Srbije, na projektu broj TR32029.

Literatura

[1] Dorottya Papp, Zhendong Ma, Levente Buttyan, "Embedded Systems Security: Threats, Vulnerabilities, and Attack Taxonomy" , Thirteenth Annual Conference on Privacy, Security and Trust (PST), 2015.
[2] Texas Instruments, "Secure In-Field Firmware Updates for MSP MCUs", Application report, 2015
[3] Mansoor Ebrahim, Shujaat Khan, Umer Bin Khalid, "Symmetric Algorithm Survey: A Comparative Analysis" International Journal of Computer Applications (0975 – 8887) Volume 61– No.20, 2013.
[4] Prashant Kumar Arya, "Comparative Study of Asymmetric Key Cryptographic Algorithms" International Journal of Computer Science & Communication Networks,Vol 5(1),17-21
[5] ETSI standards "Access, Terminals, Transmission and Multiplexing (ATTM); Integrated Broadband Cable and Television Networks;K-LAD Functional Specification", 2010

Abstract

As it is the case with software for desktop computers, embedded systems also have software which is prone to attacks and changes from unauthorized users. This paper briefly explains the need for system protection, as well as the consequences of poorly designed protection mechanisms. An overview of cryptography elements that are commonly in use, encryption/decryption, hash function, digital signature is provided. It is explained how encryption methods can be used to protect the embedded systems software. It is given an example of the proposed methods of protection with the set-top-box devices in the secure boot and software updates.

Protection of embedded systems from unauthorized modification of software and the practical application of the set-top-box

Bojan Lazić, Nebojša Mirić, Tihomir Anđelić, Velibor Mihić