HRVATSKA AKADEMSKA I ISTRAŽIVAČKA MREŽA
CROATIAN ACADEMIC AND RESEARCH NETWORK

Zaštita softvera

CCERT-PUBDOC-2004-04-71

CARNet CERT u suradnji s LS&S

Sigurnosni problemi u računalnim programima i operativnim sustavima područje je na kojem CARNet CERT kontinuirano radi.
Rezultat toga rada ovaj je dokument koji je nastao suradnjom CARNet CERT-a i LS&S-a, a za koji se nadamo se da će Vam koristiti u poboljšanju sigurnosti Vašeg sustava.

CARNet CERT, www.cert.hr - nacionalno središte za **sigurnost računalnih mreža** i sustava.

LS&S, **www.lss.hr**- laboratorij za sustave i signale pri Zavodu za elektroničke sustave i obradbu informacija Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu.

Sadržaj

1.UVOD....4

2.ZAKONSKA ZAŠTITA SOFTVERA....5

2.1.Autorsko pravo....5
2.2.Patent....5
2.3.Licenca....6
2.4.Vrste softvera....6

3.METODE ZAŠTITE SOFTVERA....8

3.1.Dongle....8
3.2.Zaštita pomoću instalacijskog medija....8
3.3.Zaštita CD-a protiv kopiranja....9
3.4.Fiksna registracijska šifra....9
3.5.Promjenjiva registracijska šifra....9
3.6.Serijski broj ovisan o hardveru....10
3.7.Zaštita programa pomoću registracijske datoteke....10

4.METODE ZA PROBIJANJE SOFTVERSKE ZAŠTITE....11

4.1.Debugging....11
4.2.Disassembling....11
4.3.Decompiling....11

5.KOMERCIJALNI SOFTVER....12

5.1.FlexLM....12
5.2.HASP....13
5.3.PC Guard....14

6.ZAKLJUČAK....16

7.REFERENCE....16

1. Uvod

Softverska industrija danas se iznimno brzo razvija i potrebe za novim aplikacijama sve su veće. Istovremeno Internet veze postaju sve brže, a računalna mreža postaje novi medij za jednostavnu i brzu distribuciju softvera. Takva situacija pruža zanimljive mogućnosti, ali istovremeno nameće i velike pravne i sigurnosne probleme proizvođačima softvera.

Softver je proizvod koji je jako teško i skupo razvijati, što dovodi do njegove visoke tržišne cijene, no istovremeno ga je zbog niske cijene CD/DVD medija te Internet pristupa lako umnožavati i distribuirati. Zbog toga se softver često ilegalno kopira i distribuira što se popularno naziva softversko piratstvo. Zbog tog piratstva softverske kompanije danas trpe ogromne gubitke. Prema nekim procjenama, danas na svaku legalnu kopiju softvera dolaze 3-4 ilegalne kopije.

Kako bi zaštitili intelektualno vlasništvo razvijenog softvera te svoje financijske interese, proizvođači softvera su počeli implementirati razne hardverske i softverske metode koje onemogućavaju neovlašteno kopiranje i distribuciju programa te njihovo korištenje. Te zaštite uključuju provjeru serijskog broja isporučenog softvera, provjeru raznih parametara računala na kojem je softver instaliran, obveznu autentikaciju svake zasebne kopije softvera, provjeru ispravnosti isporučenih licenci, enkripciju izvršnog koda programa te razne druge metode.

Još uvijek ne postoji metoda za zaštitu softvera koju bi bilo nemoguće probiti. Svaka metoda koja je do sada primijenjena u praksi na kraju je ipak razbijena te sav softver nakon nekog vremena postane dostupan u piratskoj verziji.

U takvoj situaciji nameće se pitanje kakvog uopće smisla ima ulagati vrijeme i sredstva u razvoj takvih zaštita, kad je s današnjom tehnologijom jednostavno nemoguće pobijediti ljude s dovoljno vremena, sredstava i motivacije za razbijanje softverskih zaštita. Razlog za to je potpuno komercijalan jer ako softver nema zaštitu postat će dostupan u besplatnoj verziji za svega nekoliko dana. Međutim, ako softver ima donekle kompliciranu zaštitu, moguće je da ona neće biti probijena nekoliko mjeseci ili čak godina, pa će za to vrijeme biti odgođeno pojavljivanje piratske verzije tog programa. Na taj način je moguće spriječiti značajniju financijsku štetu softverskoj kompaniji jer nakon tog razdoblja softver će ionako zastarjeti te više neće donositi značajniju dobit.

Osim toga, u softver je moguće ugraditi višestruke zaštite i provjere u različitim dijelovima programskog koda, tako da je teško ispravno razbiti zaštitu. Kod takve zaštite je moguće da piratske verzije programa prividno rade (tj. moguće je pokrenuti softver), ali u nekim fazama rada program se jednostavno ugasi, počne se ponašati nestabilno ili nema neke funkcionalnosti koje ima legalna verzija. Naravno, nakon dovoljno vremena će se sigurno pojaviti ispravna verzija piratskog programa, ali do tada može proći dosta vremena. Osim toga, kad se takav program i pojavi, na Internetu će postojati mnogo verzija piratskog softvera od kojih će većina biti neispravna. Ta činjenica može mnoge korisnike natjerati da kupe program, čak i nakon objavljivanja piratskih verzija, jednostavno zato da izbjegnu gnjavažu te gubitak vremena i novca, pokušavajući koristiti neispravan softver.

Na Internetu je moguće pronaći mnogo stranica sa detaljnim uputama za razbijanje softverskih zaštita. Tu su objašnjeni razni postupci te napisane opširne upute za korištenje specijalnih alata za razbijanje takvih zaštita. Na žalost, ne postoji ništa slično za edukaciju programera. Zbog toga mnogi programeri uopće nisu svjesni koliko su jednostavne zaštite koje oni primjenjuju te da su vjerojatno već odavno probijene.

Razlog tome je djelomično i okruženje i motivacija ove dvije skupine ljudi. Programeri su ljudi koji svoj posao rade za novac i navikli su da je tajnost podataka najbolji način za održavanje prednosti pred konkurencijom i zarađivanje što veće svote novaca. Pirati, s druge strane, ne rade za novac te zbog toga i objavljuju besplatno na Internetu sve rezultate svog rada. Piratska verzija softvera može donijeti značajne novčane uštede njegovim krajnjim korisnicima, ali pirati razbijaju softverske zaštite isključivo iz znatiželje, izazova ili ugleda koji će mu razbijanje nove i kompliciranе zaštite donijeti unutar piratske zajednice. Pirati su najčešće spremni provesti tjedne i mjesece pred računalom pokušavajući probiti nove softverske zaštite, a programeri u pravilu podcjenjuju njihov broj i motivaciju.

2. Zakonska zaštita softvera

Osnovni preduvjet za borbu protiv softverskog piratstva je donošenje zakona koji će regulirati prava autora softvera te prava njegovih korisnika. Problem sa softverom je oduvijek bila činjenica da ga je bilo teško svrstati u bilo koju od tradicionalnih kategorija intelektualnog vlasništva. Softver se može promatrati na dva načina. S jedne strane, izvorni kod programa te način implementacije različitih algoritama u sintaksi konkretnog programskog jezika se mogu promatrati kao literarno djelo i zbog toga ga je moguće zaštititi pomoću autorskog prava. S druge strane, softver je također mehanizam koji, kad se izvrši, može obaviti neki koristan posao, a takve stvari se tradicionalno štite pomoću patenata. Oko zakonskih odredbi za zaštitu softvera se u zadnje vrijeme vodi dosta rasprave i nije potpuno jasno koju količinu zaštite bi softver trebao imati. Programeri naravno žele svoj softver zaštititi što bolje tako da od njega mogu imati što veću financijsku korist, dok s druge strane korisnici žele da softver bude što dostupniji, otvoreniji i pristupačniji širokoj javnosti. U praksi je potrebno pronaći odgovarajuću razinu zaštite koja će omogućiti dovoljno zaštite programerima da se stimulira daljnji razvoj softvera te koja će osigurati da softver bude dovoljno pristupačan korisnicima.

2.1. Autorsko pravo

Autorsko pravo je najraširenији način zaštite bilo kojeg intelektualnog vlasništva. Autorsko pravo štiti autora od nelegalnog korištenja njegovog djela. Točnije, zaštita djela koju omogućuje autorsko pravo se može definirati pomoću sljedećih prava:

- pravo umnažanja djela;
- pravo na izmjene i dopune u djelu;
- pravo distribucije djela;
- pravo javnog izvođenja i prikazivanja djela;
- pravo autora na korištenje njegovog imena uz svaku kopiju djela te zaštita od korištenja imena uz tuđe radove ili uz iskrivljene verzije njegovog djela;
- pravo autora da spriječi iskrivljavanje ili uništavanje svojeg djela.

Autorsko pravo načelno štiti originalnu implementaciju i način prikaza neke ideje, a ne samu ideju. U softverskoj industriji to znači da je moguće autorskim pravom zaštititi izvorni i izvršni kod programa, strukturu i organizaciju koda programa, dijelove ili cijelo korisničko sučelje te sve priručnike, upute i ostalu dokumentaciju u digitalnom ili pisanom obliku. Autorsko pravo međutim ne štiti razne programske algoritme ili metode te matematičke postupke koji su korišteni u realizaciji softvera.

Autorsko pravo štiti od neovlaštenog kopiranja ili oponašanja koda, međutim, autorsko pravo ne štiti od konkurencije koja samostalno i nezavisno (bez uvida u izvorni kod konkurencije) razvije sličan softver. Dapače, drugi autor može čak dobiti autorsko pravo za svoj program bez obzira na sličnost s već postojećim softverom. Autorsko pravo se često koristi jer je primjenjivo na skoro svaki oblik softvera, a moguće ga je lako, brzo i jeftino dobiti.

Svako djelo zaštićeno autorskim pravom mora imati vidljivu oznaku, a mora biti označena i godina izdavanja te ime autora ili naziv tvrtke koja je nositelj autorskih prava.

2.2. Patent

Patent je zaštita izuma koju izdaje vlada neke države, a sprječava druge osobe ili organizacije da proizvode i prodaju isti ili sličan proizvod. Patentna zaštita se može primijeniti na svaki koristan princip, mehanizam, proizvodni proces i sl. koji je nov, nije očigledan i nije sadržan u nijednom prethodno objavljenom patentu. Za razliku od autorskog prava, patent zabranjuje objavu bilo kakvog sličnog rada pa makar bio i nezavisno razvijen.

Za razliku od autorskog prava koje štiti prezentaciju neke ideje i oblik izražavanja, patent štiti samu ideju. U softverskoj primjeni patent štiti ideje, algoritme i matematičke postupke korištene u programu, a ne sam programski kod.

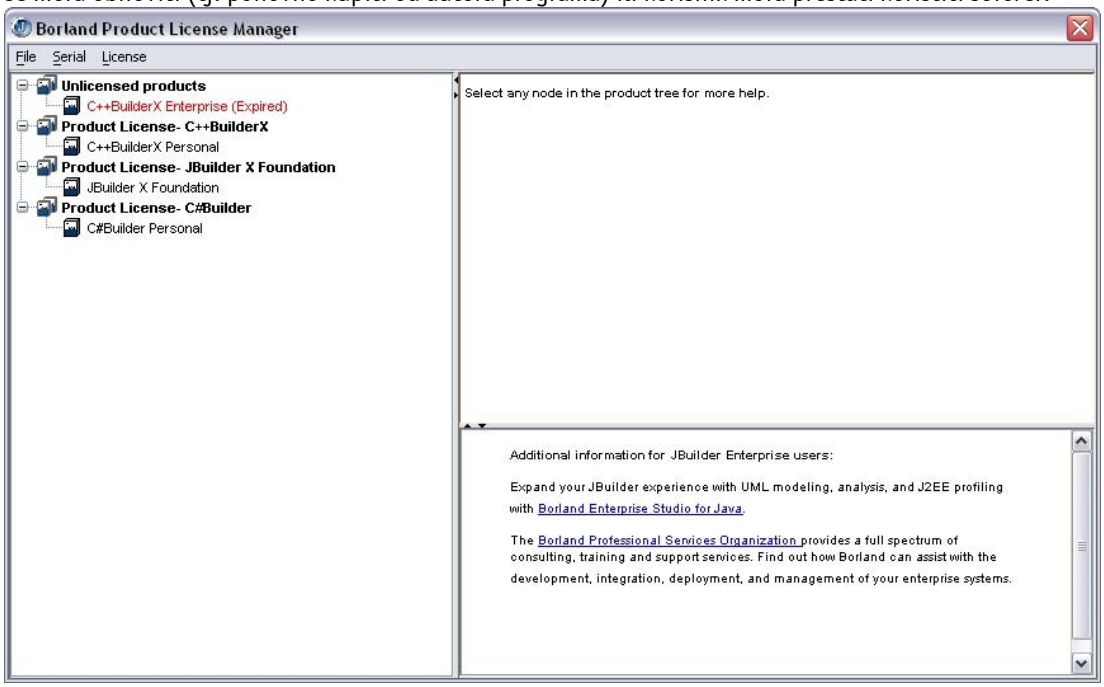
Nedostatak patenata je visoka cijena njihovog izdavanja i dugo vrijeme (obično nekoliko godina) koje mora proći od predaje zahtjeva pa do eventualnog odobrenja za izdavanje patenta. Osim toga, da bi se formule, metode ili algoritmi zaštitili patentom prvo ih je potrebno objaviti i dati na uvid komisiji za dodjeljivanje patenta, a na taj način se uglavnom gubi na sigurnosti koju pruža tajnost istih ideja.

Kako se softverska industrija nevjerojatno brzo razvija, postoji jako malo softverskih rješenja koja mogu opstati na tržištu više od nekoliko godina, koliko je potrebno za izdavanje patenta. Zbog toga je

traženje patenta za softver rijetko isplativo i koristi se samo za neke algoritme i postupke za koje se smatra da su dovoljno fundamentalni da bi mogli ostati u upotrebi desetak ili više godina.
Patent je najmoćniji zakonski oblik zaštite softvera jer, kao što je već spomenuto, zabranjuje objavu programa koji ima slične funkcije ili koristi iste softverske metode pa makar bio nezavisno razvijen. Ta činjenica je ujedno razlog velikih kontroverzi i žestokih rasprava koje se vode oko patentiranja softverskih rješenja.
Softverska industrija zagovara patentiranje softvera jer, osim nelegalnog kopiranja i očitog kopiranja koda, želi zaštititi što više inovativnosti i originalnosti koja je utrošena u razvoj softvera. Konkurencija može razviti sličan softver jednostavnim proučavanjem funkcionalnosti originalnog rješenja, čak i bez poznavanja izvornog koda, a patent je jedini način da se to spriječi. S druge strane, postoji rašireno mišljenje da bi patentiranje softverskih algoritama zaustavilo razvoj softvera, širenje znanstvenih ideja te stvorilo razne monopole u softverskoj industriji.
Danas je softver moguće patentirati u SAD-u, ali ne i u Europi.

2.3. Licenca

Većina softvera se danas izdaje pod nekom vrstom licenčnog sporazuma. Licenca je ustvari proširenje autorskog prava na neki softver. Kod licenciranog softvera korisnik uopće ne kupuje softver već kupuje samo licencu za njegovo korištenje, a vlasnik softvera ostaje autor. Licenca je posebna dozvola u kojoj je točno definirano na koji način korisnik može koristiti taj softver.
Licenca definira na koliko računala se softver smije instalirati, u koje svrhe se smije koristiti (komercijalne, osobne, obrazovne svrhe itd.) te koliko dugo je licenca važeća. Nakon isteka licence ona se mora obnoviti (tj. ponovno kupiti od autora programa) ili korisnik mora prestati koristiti softver.



***Slika 1**: Softver za nadzor nad softverskim licencama*

2.4. Vrste softvera

Prema zakonskom obliku zaštite softver se može podijeliti u sljedeće kategorije:

- *Public domain*;
- *Open Source*;
- *Freeware*;
- *Shareware*;
- Komercijalni softver;
- Komercijalni softver (licencirani).

Public domain je softver s kojim korisnik može raditi sve što želi. Dopušteno je korištenje, umnožavanje, distribucija pa čak i prodavanje bez ikakve dozvole od autora.
Open Source je softver koji je besplatan za korištenje, umnožavanje i distribuciju, a dozvoljeno je raditi promjene u izvornom kodu te takav softver dalje distribuirati. Jedini uvjet koji se obično nameće korisniku je da promijenjeni softver i dalje bude *Open Source*. Takav softver se distribuira zajedno s licencom u kojoj su definirana sva prava i obaveze korisnika. Ako korisnik distribuira takav softver, bilo u izvornom obliku bilo izmijenjen, uvijek se mora distribuirati pod istim licenčnim sporazumom.
Freeware je besplatan za korištenje i distribuciju, ali se ne smije mijenjati. Taj softver se također izdaje pod posebnom licencom sa definiranim pravilima korištenja. Autor zadržava autorsko pravo na ovakav softver.
Shareware je sličan kao i *freeware*, ali se u licenčnom sporazumu obično traži da korisnik pošalje autoru određenu svotu novca. Kod takvog softvera obično postoji definirano vrijeme u kojem korisnik može besplatno koristiti program, a ako ga želi koristiti i nakon tog razdoblja mora ga platiti. Novčane svote za takav softver su obično simbolične i to je zgodan način za distribuciju softvera koji inače ne bi opstao na tržištu komercijalnog softvera.
Komercijalni softver je softver koji korisnik mora kupiti te ga smije samo koristiti, a ne kopirati, distribuirati ili mijenjati. Postoje dva tipa komercijalnog softvera, softver koji korisnik može kupiti te licencirani softver. Ako korisnik kupi kopiju programa, može ga koristiti na način koji je definiran zakonom o autorskim pravima. Danas je skoro sav softver licenciran i u tom slučaju korisnik kupuje samo licencu za korištenje programa. Korisnik može koristiti takav softver samo na način koji je u skladu s licenčnim sporazumom te zakonom o autorskim pravima.

3. Metode zaštite softvera

Glavni problem s zakonskim mjerama koje su opisane u prethodnom poglavlju je njihova primjenjivost u praksi. Bez obzira na koji način je softver zakonski zaštićen, u praksi je teško otkriti prekršitelje i još teže na sudu dokazati krivicu. Zbog takve situacije je proizvođačima softvera najčešće lakše i jeftinije spriječiti piratstvo ugradnjom tehnoloških zaštitnih mjera u svoj softver, nego kasnije pokušavati krivično goniti prekršitelje licenčnog sporazuma.
Za zaštitu softvera se koriste različite hardverske i softverske metode. Zadaća svih tih metoda je da od korisnika zahtijevaju neku vrstu autentikacije prije nego što mu se omogući korištenje softvera. Korisnik mora imati poseban hardver koji se spoji na paralelni ili serijski komunikacijski port računala ili neku vrstu registracijske datoteke ili registracijsku šifru koju mora upisati prilikom instalacije programa. Osim toga sve te metode pokušavaju nekako šifrirati ili samo zakomplicirati izvršni kod programa tako da onemoguće pirate u pokušaju razbijanja zaštite.

3.1. Dongle

Dongle je uređaj koji je potrebno spojiti na serijski ili paralelni komunikacijski port računala da bi se mogla pokrenuti određena aplikacija. *Dongle* je relativno jednostavna naprava koja sadrži hardverski implementiran ključ za omogućavanje pristupa određenoj aplikaciji i koja je sposobna na zahtjev aplikacije poslati taj ključ.
Unutar same aplikacije se na više mjesta poziva funkcija koja komunicira s *dongle* ključem, aplikacija šalje razne upite na odgovarajući komunikacijski port i provjerava pristigle odgovore. Ako *dongle* nije prisutan ili ako ne šalje odgovarajući ključ, program će prestati s radom.
Osim zaštite od ilegalnog kopirana softvera, *dongle* se može koristiti i za onesposobljavanje određenih dijelova programa, ovisno o kupljenoj verziji softvera, odnosno o kupljenoj licenci. U tom slučaju se kupcima uvijek isporučuje ista verzija programa, ali se isporučuje *dongle* sa drukčijim ključem. Ako korisnik kupi najskuplju verziju softvera dobiti će *dongle* koji otključava sve funkcije programa, a ako kupi jeftiniju verziju dobiti će *dongle* koji omogućava rad programa, ali neke funkcije ostavlja zaključane.
Dongle ključ je prilično teško kopirati, ali je zato mnogo jednostavnije probiti funkcije za komunikaciju s *dongle* ključem unutar same aplikacije. Zbog toga je i nivo sigurnosti koji se postiže ovakvim načinom zaštite najčešće određen izvedbom same aplikacije, a ne izvedbom *donglea*. Međutim kod za provjeru ispravnosti *donglea* unutar aplikacije se može enkriptirati, a isto tako se može enkriptirati i sva komunikacija aplikacije i *dongle* uređaja.
Ako je provjera ključa ugrađena na više mjesta u programu te ako su ti dijelovi koda enkriptirani može biti jako teško pronaći i ukloniti sve provjere iz aplikacije.
Novije izvedbe *dongle* uređaja imaju ugrađenu EPROM memoriju u koju je moguće pohraniti programski kod nekih funkcija u programu. Funkcije u EPROM-u su enkriptirane i dekriptiraju se direktno u radnu memoriju kod izvršavanja programa. Na taj način je bez originalnog *donglea* praktički nemoguće probiti zaštitu jer dio koda jednostavno nedostaje.
Usprkos opisanim prednostima, *dongle* uređaji se danas skoro više uopće ne koriste, a razlozi su:

- kompliciranа instalacija, često je potreban poseban upravljački softver (engl. *driver*) za ispravan rad *dongle* ključa;
- visoka cijena, svakom korisniku je osim primjerka aplikacije potrebno isporučiti i poseban primjerak *donglea*;
- nemogućnost distribucije softvera preko Interneta.

3.2. Zaštita pomoću instalacijskog medija

Za ovaj način zaštite aplikaciju je potrebno distribuirati na mediju na koji je moguće pisati podatke kao što je meki ili tvrdi disk ili izbrisivi optički disk (CD-RW). U posebnoj datoteci na instalacijskom mediju je zapisan brojač instalacija programa. Nakon svake uspješne instalacije brojač se poveća i ponovno zapiše u istu datoteku. Kad se dostigne određeni broj instalacija, program više nije moguće instalirati s tog medija.
Za ispravan rad ove zaštite, datoteka s brojačem mora biti enkriptirana tako da joj je što teže promijeniti sadržaj, a sam medij mora biti što teže kopirati.

Ovaj mehanizam zaštite se rijetko koristi i to zbog očitih razloga kao što su vezanost uz određenu vrstu instalacijskog medija, nemogućnost distribucije softvera preko Interneta, visoka cijena i nepraktičnost.

3.3. Zaštita CD-a protiv kopiranja

Postoji više vrsta zaštite CD medija protiv kopiranja i svi programi koji se isporučuju na CD mediju koriste takvu zaštitu.

Najjednostavniji oblik zaštite je u obliku provjere da li se ispravan CD nalazi u CD čitaču. Na taj način se može spriječiti kopiranje i pokretanje programa s tvrdog diska računala, ali je također nemoguće razlikovati originalni od ilegalno kopiranog CD-a. Čak i takva jednostavna zaštita je korisna jer onemogućava softverske pirate da rade tzv. *ripanje* originalnog programa. *Ripanje* je postupak koji se najčešće primjenjuje za računalne igre, a njime se s originalnog CD-a uklone sve slike, animacije, zvukovi, DirectX upravljački programi i svi ostali dijelovi koji zauzimaju puno prostora, a nisu nužno potrebni za rad programa. Nakon toga se preostali sadržaj komprimira i distribuira preko Interneta. Tako nastala verzija programa puno je manja, što olakšava njeno skidanje s Interneta osobama sa sporom vezom.

Kompliciranije zaštite protiv kopiranja CD medija se obično rade tako da se na originalni CD snimi informacija koju je nemoguće kopirati pomoću komercijalno dostupnog softvera za kopiranje CD-ova. Veliki dio softvera provjerava naziv (engl. *Label*) CD-a koji se trenutno nalazi u čitaču. Ako se za naziv CD-a koriste nestandardni znakovi, to može zbuniti većinu softvera za kopiranje CD medija.

Zaštita CD-a se često izvodi tako da se provjeri da li su na CD-u prisutne sve datoteke. Posebno se provjeravaju datoteke koje se najčešće uklanjaju prilikom *ripanja* CD-a.

3.4. Fiksna registracijska šifra

Vjerojatno najjednostavniji oblik zaštite je ugrađivanje programske funkcije koja zahtijeva od korisnika unošenje određene šifre za registraciju. Ta vrijednost je uvijek ista i ne ovisi o nikakvim parametrima kao što su korisnički podaci ili podaci o korisnikovom računalu. Kod takve zaštite je dovoljno da pirat jednom pronađe registracijsku šifru i objavi je na Internetu pa da svatko može besplatno registrirati taj softver.

Usprkos svojoj jednostavnosti, ovaj tip zaštite ima neke prednosti nad ostalim mehanizmima. Najveća prednost je što ispravna šifra za registraciju ne mora biti pohranjena na disku već je pohranjena negdje u programskom kodu. Tu šifru je nemoguće saznati čitanjem programskog koda jer se obje šifre (šifra pohranjena u kodu i šifra koju upiše korisnik) najprije nekim složenim matematičkim operacijama pretvore u nove vrijednosti koje se tek tada uspoređuju. Ako su te operacije dovoljno komplicirane može biti jako teško shvatiti ispravan postupak računanja šifre.

Najčešći način uklanjanja te šifre je prepravljanje dijela koda za provjeru šifre tako da se potpuno zaobiđe provjera i program normalno nastavi s radom. Da se spriječi takvo probijanje zaštite, moguće je koristiti registracijsku šifru za dekriptiranje dijelova programskog koda. Ako je dio koda enkriptiran pomoću registracijske šifre onda će ga biti moguće dekriptirati samo pomoću te iste šifre. Čak i ako pirat uspije premostiti algoritam za provjeru šifre kod upisa program još uvijek neće ispravno raditi.

Enkriptirani dio koda se može dekriptirati u memoriji neposredno prije nego korisnik pokrene tu funkciju, i kada funkcija više nije potrebna ponovno se enkriptira. Ako program ima nekoliko enkriptiranih dijelova onda će maksimalno jedan od tih dijelova biti dekriptiran u memoriji u svakom trenutku, što piratima onemogućava snimanje dektiptiranog programskog koda iz memorije.

3.5. Promjenjiva registracijska šifra

Bolji način zaštite je generiranje registracijske šifre pomoću parametara koji su sakupljeni iz podataka o korisniku ili tajno prikupljeni iz parametara korisničkog računala.

Ovakav program generira šifru iz korisničkih podataka te koristi podatke poput korisničkog imena, adrese, naziva tvrtke i sl. Kada korisnik upiše svoju šifru, program provjerava da li se šifra poklapa s izračunatom šifrom.

Program koji koristi podatke sakupljene iz korisničkog računala radi na sličan način samo što umjesto korisničkih podataka koristi razne podatke o korisničkom računalu poput serijskog broja diska, raznih postavki Windows *Registrya* i sl. Iz prikupljenih vrijednosti program računa neku vrijednost i pohranjuje je u skrivenoj datoteci na tvrdom disku računala. Korisnik mora proizvođaču softvera poslati generirani

identifikacijski broj računala na osnovu kojeg mu proizvođač softvera vrati odgovarajuću šifru za registraciju softvera.

3.6. Serijski broj ovisan o hardveru

Ovo je najčešće korišten način hardverske zaštite računala. Prilikom instalacije programa generira se pseudo-slučajni serijski broj računala. Aplikacija šifrira taj broj i sakrije ga u posebnu datoteku, a može ga zapisati i u neku od datoteka operacijskog sustava. Serijski broj se generira pomoću serijskog broja isporučene kopije softvera te iz prikupljenih podataka o hardveru računala i konfiguraciji operacijskog sustava. Na taj način se postiže jedinstveni generirani broj.

Nakon instalacije program je potrebno registrirati. U procesu registracije, generirani broj se šalje proizvođaču softvera koji korisniku vraća registracijsku šifru koja odgovara generiranom serijskom broju hardvera.

3.7. Zaštita programa pomoću registracijske datoteke

Registracijske datoteke imaju istu ulogu kao i registracijske šifre, a prednost im je količina informacija koju je moguće spremiti u njih. Registracijska datoteka može sadržavati informacije o korisniku, registracijsku šifru za autentikaciju korisnika, ključeve za dekriptiranje enkriptiranih dijelova izvršnog koda aplikacije i sl. Registracijska datoteka je obično enkriptirana tako da je nemoguće pročitati i promijeniti njen sadržaj.

U takvoj datoteci može biti zapisan dio izvršnog koda aplikacije tako ako datoteka ne postoji ili je neispravno dekriptirana, aplikaciji nedostaje dio koda i ne može ispravno raditi. U registracijsku datoteku je moguće pohraniti podatke o hardveru korisnikovog računala i na taj način je potrebna jedinstvena datoteka za svako računalo.

Kao i kod registracijskih šifri, iz aplikacije je moguće ukloniti dio koda koji provjerava ispravnost registracijske datoteke. Kako bi se to otežalo, provjera ispravnosti datoteke se ugrađuje u više mjesta u programu i informacije u datoteci se koriste za šifriranje dijelova izvršnog koda.

4. Metode za probijanje softverske zaštite

Postoje dva glavna pristupa probijanju softverske zaštite. Prvi pristup je pokretanje programa unutar nekog alata za ispravljanje pogrešaka u kodu (engl. *debugging*), a drugi pristup je reverzno prevođenje programa iz izvršnog koda natrag u asemblerski kod (engl. *disassembling*) ili neki od viših programskih jezika (engl. *decompiling*).

4.1. Debugging

Alati za ispravljanje pogrešaka u kodu omogućuju interaktivno pokretanje programa. Program za *debugging* može zaustaviti izvršavanje programa između svake dvije instrukcije. Nakon što se program zaustavi, programer može vidjeti u kojem dijelu izvornog koda programa se proces trenutno nalazi. Osim dijela izvornog koda, programer u svakom trenutku može vidjeti sadržaj radne memorije, sadržaj procesorskih registara, podatke koji se zapisuju u datoteke, informacije o programskim nitima i mnoge druge informacije. Takve alate koriste programeri za vrijeme razvoja softvera jer im omogućuju detaljan uvid u rad programa i uklanjanje pogrešaka u programiranju. Sličan alat je ugrađen u svaki softverski paket koji je namijenjen razvoju aplikacija, poput Microsoft Visual Studia ili Borland C++ Buildera. Međutim alati za *debugging* se često rade i kao samostalne aplikacije i takvi alati su obično puno moćniji. Najbolji *debugging* alati mogu komunicirati direktno s jezgrom operacijskog sustava i zaobići pozive sustava. Na taj način su puno stabilniji u radu, mogu prepoznati više informacija u stanju sustava te čak omogućuju analizu hardverskih upravljačkih programa (engl. *drivera*).
Korištenjem takvih alata, pirati mogu zaustaviti program u trenutku kad on zahtijeva autorizaciju korisnika. Nakon što se program zaustavi, moguće je vidjeti dio koda koji je odgovoran za provjeru autorizacije i izmijeniti ga tako da prihvaća svaku šifru.
Najpoznatiji alat za *debugging* je SoftIce tvrtke Compuware, program može uhvatiti mnoge pogreške koje drugim alatima promaknu, pa čak prikazati pogrešku nakon blokiranja Windows operacijskog sustava.

4.2. Disassembling

Disassembling i *decompiling* su postupci koji izvršni kod programa pretvaraju natrag u izvorni kod, tako da ga je moguće analizirati. Alati za *disassembling* pretvaraju izvršni binarni kod u asemblerski kod. Kako je asembler jezik koji se direktno (jedna instrukcija asemblera odgovara jednoj binarnoj instrukciji) prevodi u binarni kod, *disassembling* alati mogu prevesti svaku aplikaciju, bez obzira na programski jezik u kojem je ona izvorno napisana.
Najbolji alati mogu generirati strukturiran kod koji je moguće lako pratiti i analizirati pa čak kreiraju i komentare za lakše snalaženje u programu.
Najčešće korišten alat za *disassembling* je IDA Pro tvrtke NuMega.

4.3. Decompiling

Alati za *decompiling* pretvaraju izvršni binarni kod programa u izvorni kod nekog od programskog jezika više razine kao što je C++ ili Java. Ovi alati mogu prevesti samo izvršni kod nekog određenog jezika za koji je taj alat razvijen. Na primjer, alat napravljen za programski jezik C++ ne može prevesti program napisan u Javi. Najveća prednost ovih alata nad alatima za *disassembling* je činjenica da većina programera i pirata puno bolje poznaje jezike više razine od asemblera i takav kod im je puno lakše analizirati.

5. Komercijalni softver

5.1. FlexLM

FlexLM (*Flex License Manager*) je softver tvrtke Macrovision. FlexLM je izuzetno kompliciran sustav nadzora nad softverskim licencama i predstavlja danas najrašireniје komercijalno rješenje za zaštitu softvera.

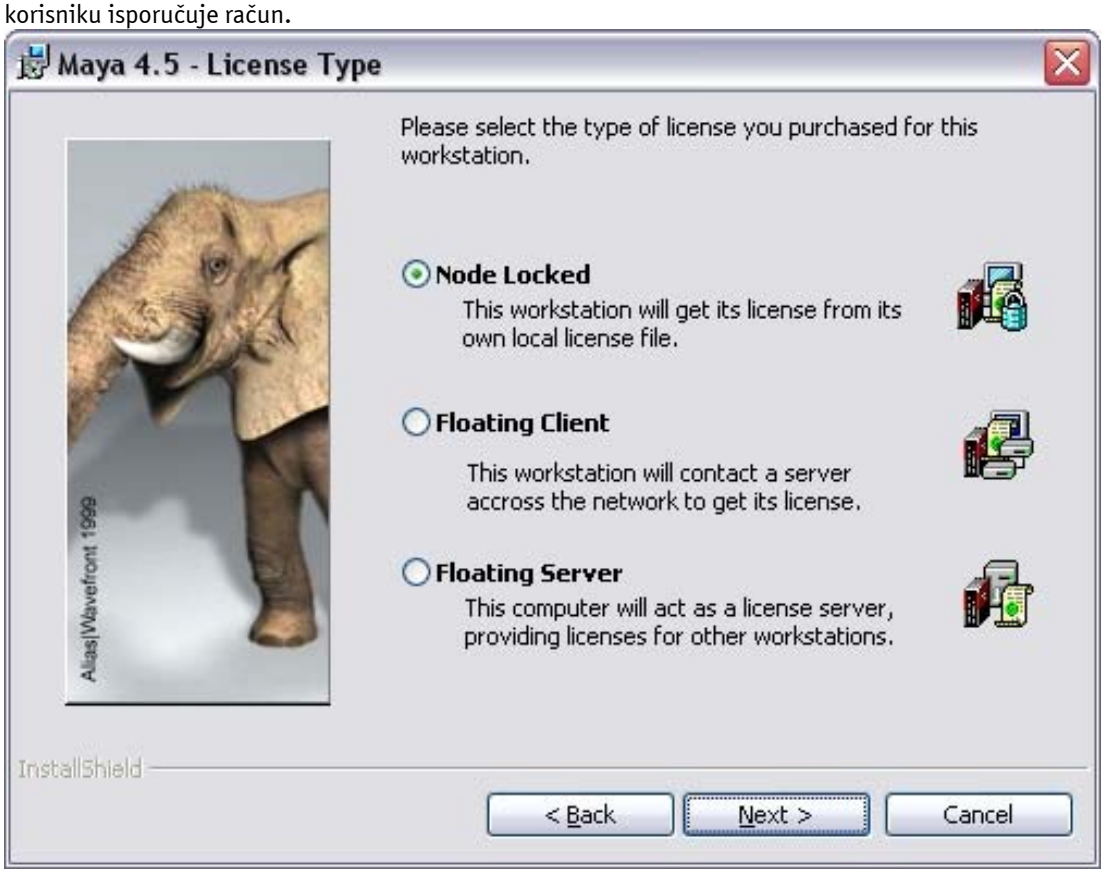
FlexLM softver uključuje zaštitu izvršnog koda aplikacije, kontrolu ispravnosti registracijskih datoteka na disku, softver za implementaciju na poslužiteljima koji nadziru korištenje licenci u lokalnoj mreži, sustav za lako obnavljanje licenci i centralnu administraciju cijelog sustava.

Za zaštitu samih aplikacija FlexLM softver koristi enkripciju izvršnog koda aplikacije, enkriptiranu registracijsku datoteku koja je ovisna o hardveru korisničkog računala, kontrolu poslužitelja kojom se neovlaštenim korisnicima zabranjuje pokretanje mrežnih aplikacija, a u nekim izvedbama se koristi i *dongle*.

Sustav nadzire korištenje softvera i ispravnost licenci, a moguće je izabrati različite metode licenciranja softvera. Softver podržava nekoliko desetaka različitih licenčnih sustava, a najpoznatiji su:

- *Node locked* – softver je moguće pokrenuti samo na računalu za koje je izdana licenca;
- *User based* – softver može pokrenuti samo korisnik koji je vlasnik licence (licenca je vezana uz osobu, a ne računalo kao kod *Node locked* sustava);
- *Site licensing* – softver mogu pokrenuti svi korisnici na određenoj lokaciji (firmi, odjelu itd.);
- *Floating license* – softver mogu pokrenuti svi korisnici u lokalnoj mreži, ali postoji ograničenje na maksimalni broj korisnika koji mogu istovremeno raditi sa programom.

Softver podržava i licenciranje prema vremenu utrošenom u radu s zaštićenim programom. FlexLM poslužitelj prati vrijeme provedeno u radu s softverom za sve korisnike u mreži te se na osnovu toga korisniku isporučuje račun.

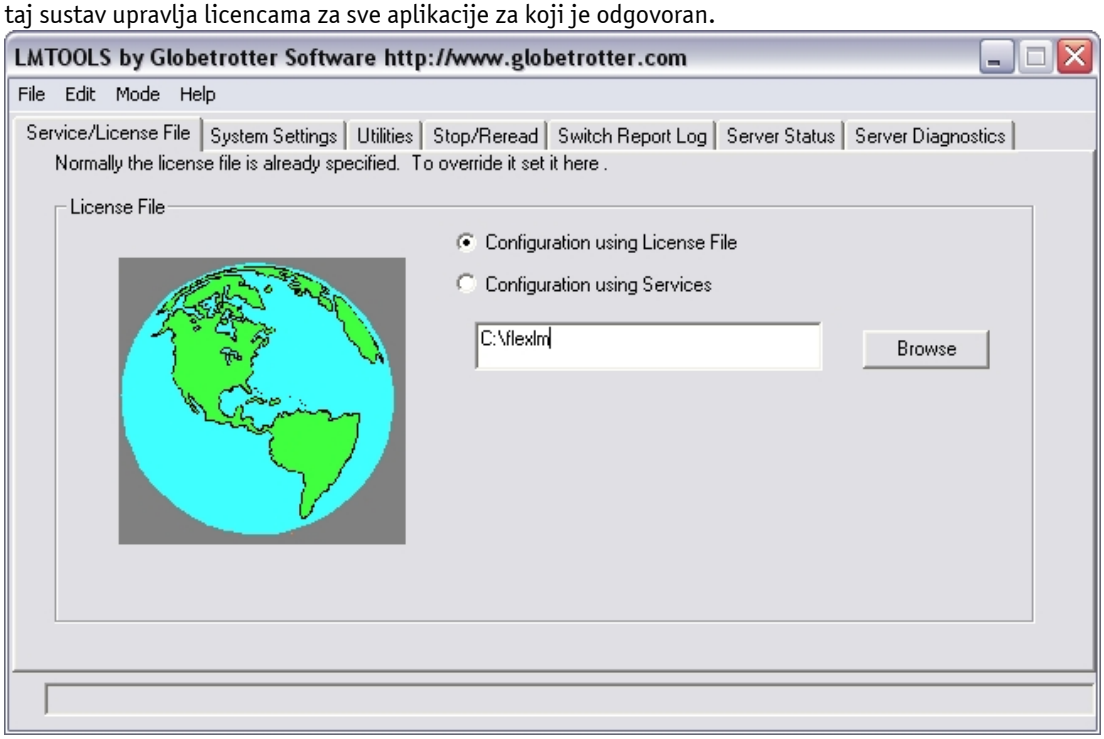


Slika 2: *Izbor vrste licenciranja*

FlexLM sustav se sastoji od 4 glavna dijela i to su:

- aplikacijski softver (engl. *application software*);
- registracijska (licenčna) datoteka (engl. *license file*);
- poslužitelj za kontrolu licenci (engl. *license manager deamon*);
- poslužitelj proizvođača softvera (engl. *vendor daemon*).

Aplikacijski softver je softver koji je zaštićen FlexLM sustavom. Softver ima ugrađene funkcije koje provjeravaju postojanje i ispravnost registracijske datoteke. Datoteka se mora nalaziti u određenom direktoriju, a u datoteci je naveden naziv i inačica softvera, podaci o vlasniku licence te razdoblje u kojem je licenca važeća.
U registracijskoj datoteci je zapisana i staza do poslužitelja za kontrolu licenci čija je zadaća nadzor licenci u lokalnoj mreži. Aplikacija se spaja na taj poslužitelj i od njega saznaje lokaciju poslužitelja softverske kompanije. Poslužitelj proizvođača softvera nadgleda korištenje licenci za sve mušterije tog proizvođača.
S obzirom da je FlexLM jako popularan softver koji koriste mnogi proizvođači softvera za zaštitu svojih aplikacija, moguće je da korisnik na svom računalu ima više aplikacija od različitih proizvođača, a koje su sve zaštićene FlexLM sustavom. U tom slučaju korisnik mora instalirati FlexLM softver samo jednom i taj sustav upravlja licencama za sve aplikacije za koji je odgovoran.



Slika 3: *FlexLM softver za kontrolu licenci*

Iako je ovaj sustav jako poznat i koriste ga stotine proizvođača softvera širom svijeta, njegova komplicirana zaštita je ipak probijena i vrlo brzo se pojavljuju piratske verzije softvera koji koristi ovakvu zaštitu.
FlexLM softver se instalira kao dijeljena DLL biblioteka funkcija i aplikacija koja je zaštićena FlexLM sustavom komunicira s tom bibliotekom svaki puta kada provjerava ispravnost licenci. Takva implementacija ustvari olakšava probijanje zaštite.

5.2. HASP

Softver HASP (*Hardware Against Software Piracy*), koji proizvodi tvrtka Aladdin, je sustav za zaštitu aplikacija koji primjenjuje *dongle* ključeve. Softver podržava *dongle* ključeve koji se spajaju na skoro sve moguće komunikacijske portove u računalu. Postoje izvedbe *donglea* za paralelni port, USB port, PCI sabirnicu (kartica koja se ugradi u računalo) te kao PCMCIA kartica.
Na slikama 4 i 5 su prikazani *dongle* uređaji za paralelni i USB port.

***Slika 4**: Dongle za paralelni port*

***Slika 5**: Dongle za USB port*

Svi *dongle* uređaji imaju vlastitu EPROM memoriju i procesor koji enkriptira i dekriptira podatke u memoriji. Kako je gotovo nemoguće pročitati podatke iz memorije bez poznavanja ispravnog ključa, postiže se nivo zaštite koji je jako teško razbiti.

Dongle ima dovoljno mjesta u memoriji za pohranu 112 različitih licenci za 112 različitih aplikacija, a s istim uređajem je moguće štititi aplikacije na Windows, Unix i Macintosh platformama.

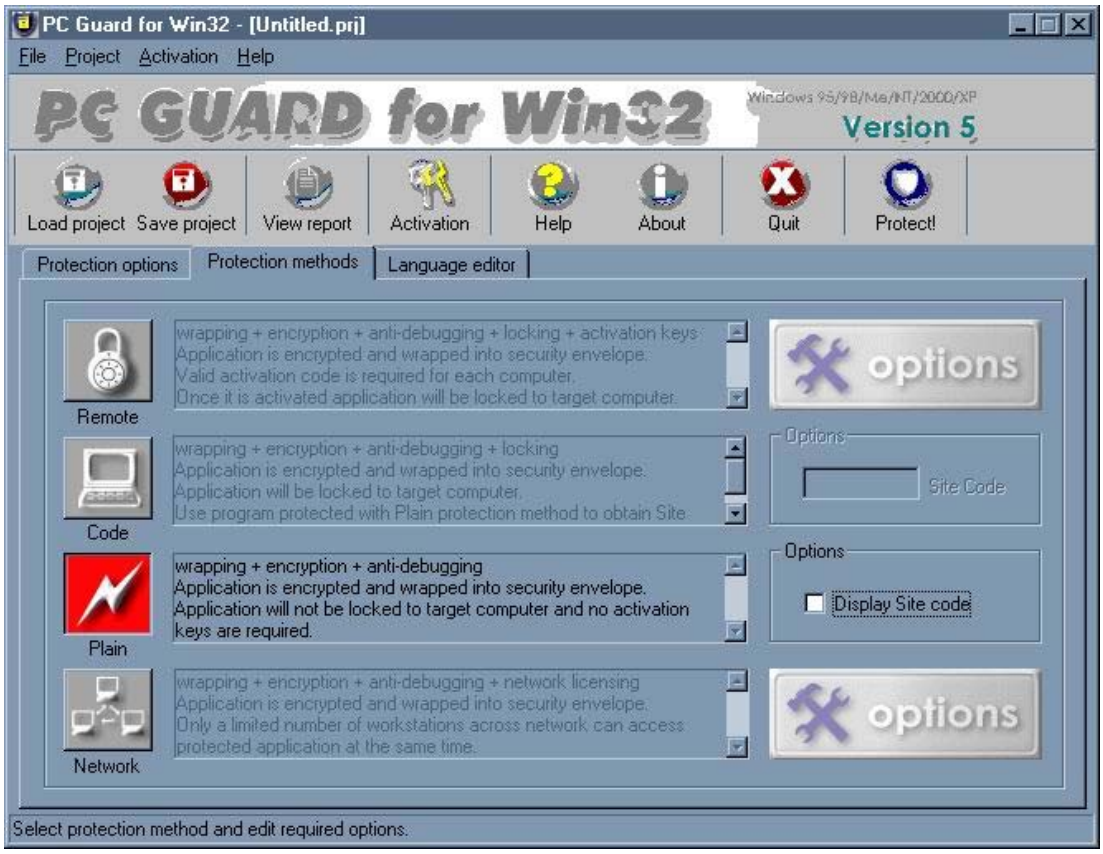
Softver koji se isporučuje s ovim uređajima ima dva načina rada. Korisničku aplikaciju je moguće zaštititi korištenjem takozvane omotnice ili promjenom izvornog koda aplikacije. Omotnica zaštićuje korisničku aplikaciju tako da enkriptira izvršnu datoteku i doda dio koda koji omogućuje izvršavanje aplikacije. Taj način zaštite je jednostavniji jer ne zahtijeva promjenu izvornog koda aplikacije, ali je manje siguran. Drugi način zaštite je da se prilikom programiranja u softver ugrade posebne funkcije iz biblioteke koja se isporučuje s Aladdin softverom.

5.3. PC Guard

PC Guard je softver kompanije Sofpro, namijenjen zaštiti softvera i kontroli licenci na Windows platformama. PC Guard štiti softver pomoću elektroničke omotnice ili API poziva u izvornom kodu aplikacije, slično kao što je to opisano za HASP softver. Zaštita se postiže enkripcijom izvršnog koda i detektiranjem pokušaja analize koda.

Za autentikaciju korisnika PC Guard koristi registracijsku šifru koja ovisi o hardveru korisničkog računala. PC Guard može detektirati registracijske oznake tvrdog diska, CD/DVD-a, BIOS-a, Ethernet kartice, procesora i operacijskog sustava na korisničkom računalu i te informacije koristi za kontrolu korisnikove registracijske šifre.

Postoji posebna verzija PC Guard softvera za klasične Windows aplikacije te posebna verzija za .NET aplikacije.



Slika 6: *Korisničko sučelje programa PC Guard*

Rad s programom PC Guard je vrlo jednostavan. Aplikaciju je moguće zaštiti tako da se upiše ime aplikacije, njena registracijska oznaka te izabere način licenciranja. Na slici 6 je prikazano korisničko sučelje programa PC Guard i to prozor u kojem se izabire jedan od 4 moguća načina licenciranja.

6. Zaključak

Iz ovog dokumenta je vidljivo da je zaštita softvera predstavlja veliki problem u modernom društvu. Zaštita softvera je proces povezan ne samo s informatičkom tehnologijom, već i sa zakonodavstvom i ekonomijom. Moderni sustavi za kontrolu softverskih licenci su narasli u velike, distribuirane, mrežne sustave kojima je zadaća ne samo borba protiv softverskog piratstva, već i stvaranje infrastrukture koja će omogućiti efikasniji način naplate korištenja softvera. U ne tako dalekoj budućnosti se softver vjerojatno više neće plaćati unaprijed kao danas, već prema vremenu korištenja, poput naplate struje ili vode, a sustavi za kontrolu licenci to moraju omogućiti.

S druge strane, sustavi za zaštitu softvera još uvijek ne uspijevaju obaviti svoju primarnu zadaću, a to je zaštita softvera od izrade piratskih verzija. Iako je na Internetu moguće pronaći tvrtke koje za svoje sustave zaštite tvrde da nikad nisu probijeni, praksa pokazuje da je razbijanje bilo kakve softverske zaštite samo pitanje vremena. Često je isplativije u softver ugraditi vlastiti mehanizam za zaštitu, nego kupovati poznati, komercijalni sustav. Iako komercijalni sustav vjerojatno ima bolje algoritme za zaštitu, iskusni pirati imaju uhodane procedure za probijanje takvih poznatih i često korištenih sustava za zaštitu.

Odluka o ugradnji zaštite u softver je uvijek ekonomska i potrebno je naći kompromis između cijene ugradnje takve zaštite i potencijalnih gubitaka od prodaje softvera ako se zaštita ne ugradi. Vrlo rijetko se isplati ulagati u najskuplje metode zaštite softvera jer svaka zaštita će s vremenom biti probijena i bitno je samo da se to odgodi dovoljno dugo kako bi proizvođač softvera ipak ostvario zadovoljavajući profit od prodaje softvera.

7. Reference

1. Stranica tvrtke Aladdin: http://www.ealaddin.com
2. Stranica tvrtke Macrovision: http://www.macrovision.com
3. Stranica tvrtke Sofpro: http://www.sofpro.com