Hardwarové a softwarové řešení bezpečnosti

Daniel CVRČEK[1], Petr ŠVÉDA[2]

[1]Ústav informatiky a výpočetní techniky Vysokého učení technického
Božetěchova 2, 612 66 Brno
cvrcek@fit.vutbr.cz

[2]Fakulta informatiky Masarykovy univerzity
Botanická 68a, 602 00 Brno
xsveda@fi.muni.cz

Abstrakt

Úvodní část příspěvku se zabývá otázkou kritérií pro hodnocení bezpečnosti informačních technologií sloužícím jako měřítko pro hodnocení bezpečnosti produktů a systémů. Současně používaná společná kritéria jsou diskutována s ohledy na návaznost starších norem, terminologie, metodiky a následné použitelnosti v podmínkách komerčního sektoru. Dále je pozornost věnována bezpečnému hardware. Je vymezen pojem bezpečného hardwarového prvku, pojem důvěry a diskutován pojem důvěryhodného zařízení. Další část se zabývá možnou alternativní charakteristikou bezpečných zařízení, která je založena na lehce definovatelných vlastnostech. Poslední část se věnuje konkrétně nejsložitějším bezpečným zařízením – kryptografickým koprocesorům. Jsou popsány problémy spojené s používáním těchto bezpečných zařízení a je ukázáno nové nebezpečí spojené s jejich praktickým používáním – složitost aplikačního rozhraní. Toto je demonstrováno na útocích, které byly objeveny a implementovány na reálných zařízeních.

Klíčová slova: bezpečný hardware, hodnocení bezpečnosti, kritéria hodnocení bezpečnosti informačních technologií, normy, společná kritéria, API, koprocesor.

1 Normy pro hodnocení bezpečnosti

Produkty (angl. *products*) jsou vyvíjeny pro nějaké obecné prostředí stanovené výrobcem na základě marketingových výzkumů, obchodních zkušeností a jiných komerčních faktorů. S ohledem na bezpečnost jsou produkty navrhovány s uvážením modelu obecných hrozeb. Tyto obecné modely vytvářejí standardizační instituce či pověřené vládní agentury podle norem hodnocení bezpečnosti. Na rozdíl od produktů jsou *systémy* (angl. *systems*) navrhovány a vytvářeny podle potřeb a požadavků konkrétních uživatelů či skupin uživatelů. Systém má jedinečné operační prostředí a jeho bezpečnostní hrozby jsou definovány přímo na základě analýzy rizik podle požadavků konkrétního uživatele.

Systém se může skládat z jediného produktu, avšak v praxi jsou mnohem běžnější případy kdy se systémy skládají z řady různých produktů. Systémový integrátor je pak zodpovědný za konkrétní spolupráci všech částí systému z funkčního i bezpečnostního hlediska. Integrace certifikovaných produktů je netriviální záležitostí. Aby celý systém mohl splnit bezpečnostní nároky musí bezpečnostní politiky a mechanismy jednotlivých subsystémů být plně kompatibilní. Celý systém musí rovněž mít koncepční návrh.

2 Kritéria hodnocení bezpečnosti informačních technologií

Kritéria pro hodnocení bezpečnosti IT (dále jen "kritéria") slouží především jako měřítko používané k hodnocení informačních technologií s ohledem na jejich bezpečnost, na konkrétní aplikace služeb a na opatření k zajištění bezpečnosti. S ohledem na praktickou realizaci zájmů zainteresovaných stran v Severní Americe a Evropě se zástupci EU, Kanady a Spojených států (NSA i NIST) podíleli na vývoji *Společných kritérií* (angl. *Common Criteria*, zkráceně též *CC*) [1], přijatých také jako norma ISO/IEC 15408 (rovněž přijaté jako ČSN, první část v českém jazyce).

Proces hodnocení bezpečnosti IT podle CC prokazuje úroveň důvěryhodnosti, s jakou bezpečnostní funkce produktu nebo systému IT splňují stanovené požadavky. Stanovuje *míru zaručitelnosti bezpečnosti* (angl. *Evaluation Assurance Level*, zkráceně též *EAL*) udělované těmto bezpečnostním funkcím. Kritéria CC definují hierarchicky uspořádané úrovně zaručitelnosti bezpečnosti. Množiny požadavků na splnění jednotlivých mír zaručitelnosti bezpečnosti, a tím pádem i míry zaručitelnosti bezpečnosti, jsou uspořádané do hierarchické soustavy podle těchto úrovní. Výsledkem hodnocení je výrok o prokázání úrovně důvěryhodnosti, s jakou bezpečnostní funkce produktu nebo systému IT a míry zaručitelnosti bezpečnosti udělené těmto bezpečnostním funkcím splňují zavedené požadavky. Výrok sděluje, kterou úroveň zaručitelnosti produkt nebo systém IT splňuje. Vzájemnou korespondenci CC a dříve používaných kritérií ITSEC [2] a TCSEC [3] ukazuje následující tabulka.

CC [1]	ITSEC [2]	TCSEC [3]
EAL1*	E0	D
EAL2	F-C1, E1	C1
EAL3	F-C2, E2	C2
EAL4	F-B1, E3	B1
EAL5	F-B2, E4	B2
EAL6	F-B3, E5	B3
EAL7	F-B3, E6	A1

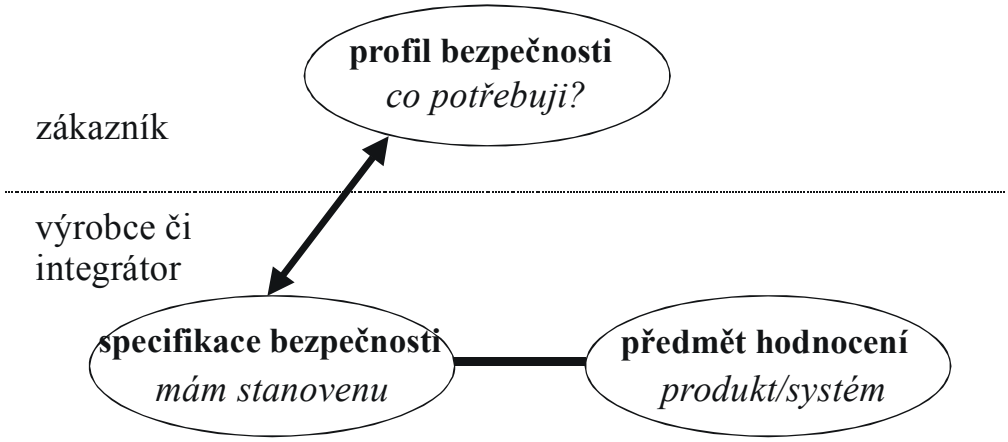
ITSEC definuje 10 funkčních tříd, z nichž prvních pět odpovídá funkčním požadavkům dle TCSEC (označované jako F-C1 až F-B3).
** Požadavek funkčního testování v rámci CC staví úroveň záruk EAL1 nepatrně výše než E0 resp. D*

2.1 Terminologie společných kritérií

Společná kritéria pracují se třemi typy hodnocení:

- hodnocení *profilu bezpečnosti* (angl. *protection profile*, zkráceně *PP*) – cílem je prokázat, že daný profil úplně, uceleně a důsledně pokrývá stanovené potřeby;
- hodnocení *specifikace bezpečnosti* (angl. *secutity target*, zkráceně *ST*) – cílem je posoudit teoretickou podobu bezpečnosti konkrétního předmětu hodnocení;
- hodnocení *předmětu hodnocení* (angl. *target of evaluation*, zkráceně *TOE*) – cílem je ověřit, zda daný předmět hodnocení splnil všechny požadavky uvedené v jeho bezpečnostní specifikaci.

Podrobněji se touto problematikou zabývá například [4]. Stručně schematicky lze vzájemný vztah jednotlivých typů hodnocení znázornit obrázkem:



2.2 Reálné možnosti verifikace a proveditelnost

Současný stav věci v oblasti hodnocení produktů a systémů je v komerční sféře dosti neutěšený. Vzhledem k nedostatku firem a expertů schopných provádět hodnocení a certifikace jsou vlastní náklady na tento proces velmi vysoké. Z komerčního hlediska je tedy úroveň záruk EAL4 (metodický návrh, testování, kontrola) dle CC tou nejnáročnější úrovní, jejíž dosažení je z hlediska nákladů únosné. Obvykle je certifikace pro tuto úroveň získávána na základě detailního popisu návrhu a zdokumentováním odolnosti proti útokům s omezenými zdroji. Vyšší úrovně záruky se již opírají o existenci formálního modelu a vyžadují použití speciálních metod vývoje.

Podstatné zjednodušení atestace by měla přinést metodika CEM [5]. Hodnocení by se tak nemělo opírat o procházení jednotlivých tříd požadavků na bezpečnostní záruky, ale mělo by probíhat na základě kroků stanovených metodikou. Ta by rovněž měla upravovat způsob správy vydaných certifikátů. Dále by metodika měla obsahovat postup pro atestaci novějších verzí již dříve validovaných produktů. Ve své momentální podobě metodika pokrývá pouze nízkou a střední třídu záruk (do úrovně EAL4).

V celém schématu záruk je třeba rovněž zohlednit vyjádření výrobce ohledně budoucí údržby a vývoje předmětu hodnocení, aby prováděné funkční úpravy neznamenaly zásadní ústupky na úkor bezpečnostních funkcí. Dodržení postupů pro uchování záruk významně snižuje náročnost a složitost hodnocení nových verzí předmětu hodnocení.

3 Co je to bezpečný hardware?

Pojmem *bezpečný hardwarový prvek* (angl. *secure hardware device*) rozumíme hardware, firmware či případnou kombinaci obou, která je schopna provádět kryptografické operace nad vstupními daty na základě dat uschovaných v paměti prvku. Všechny operace se dějí uvnitř zařízení, které má pevně definovanou nepropustnou hranici (po fyzické i logické stránce).

Označením *narušení* (angl. *tampering*) se v oblasti bezpečnosti IT rozumí neautorizovaná modifikace, která změní správnou funkci zařízení nebo systému takovým způsobem, jenž může ovlivnit poskytovanou bezpečnost nebo funkčnost. Pojem *odolnost vůči narušení* (angl. *tamper resistance*) popisuje část systému, který je chráněn proti neautorizované modifikaci způsobem poskytujícím podstatně vyšší úroveň ochrany než ostatní části systému. Obvykle se jedná o zvláštní modul obsahující vlastní procesor a paměť, který je odolný vůči fyzickému narušení (např. je uložen v trezoru, nebo má zvláštní pouzdro), a který zničí uschované chráněné informace v případě kdy k fyzickému narušení dojde. Teoretická rovina a realita jsou

však mnohdy dva odlišné světy, viz. [6]. Vlastní požadavek odolnosti vůči narušení je velmi silný a v praxi se mnohdy řada aplikací omezuje na pouhou detekci narušení.

3.1 Důvěra

Pojem *důvěry* (ang. *trust*) je často zmiňován bez podrobnější specifikace či dokonce definice. Obecně ji lze formulovat jako víru, že daný systém splňuje dané (bezpečnostní) požadavky a specifikace. Ale existují i kontroverzní podoby definice důvěry. Například podle NSA [7] existuje možnost, kdy důvěryhodný systém či komponenta může porušit bezpečnostní politiku bez zanechání průkazných stop. Důvěra je tedy subjektivní veličina. Poměrně rozsáhlou diskusi na téma možných definic důvěry v rámci termínů teorie informací lze nalézt na [8].

V rámci jednotlivých produktů lze jejich důvěryhodnost hodnotit na základě certifikací dle obecně uznávaných kritérií a norem. Bohužel reálné systémy nejsou stavebnice, jejichž komponentami jsou pouze certifikované produkty. Při integraci hardwarových prvků je tedy nutné softwarové části systému navrhovat a tvořit tak, aby vyhovovaly certifikaci minimálně dle shodné či vyšší bezpečnostní třídy. Otázkou zůstává zda jsou na software v rámci směrnic pro certifikaci hardwarových prvků kladeny dostatečné nároky. Je pro software spolupracující s hardwarovým prvkem certifikovaným dle FIPS 140-1 jako dostatečný požadavek pouhého metodického návrhu a testování?

3.2 Důvěryhodné zařízení

Nedostatkem řady hardwarových prvků pro úschovu soukromých klíčů uživatelů je například absence jakéhokoliv rozhraní pro přímou komunikaci s uživatelem. Z bezpečnostního hlediska čipová karta nemá zabezpečenu důvěryhodnou autentizační cestu. Uživatelské heslo či PIN chránící soukromý klíč je tedy nutné zadat prostřednictvím klávesnice počítače. Tato situace přímo vybízí k útokům ve formě trojských koní či virů.

Otázka důvěry tudíž není postavena pouze čistě na vlastnostech hardwarových prvků. Nemalý podíl pramení z historického vývoje, vžitých zásad a tradic. Například bankovnímu ústavu důvěřují lidé obecně více než obchodníkovi. V jakém systému lze data považovat za bezpečně uložena? Z hlediska uživatelské bezpečnosti lze jako důvěryhodné považovat řešení, kdy:

- data jsou dokonale zabezpečena (například šifrováním všech dat na zařízení s omezeným počtem pokusů na zadání hesla, kdy po jejich vyčerpání dojde ke zničení všech uživatelských dat);
- vlastníkem zařízení je uživatel a spadá pod jeho neustálou okamžitou kontrolu;
- zařízení obsahuje čip či paměťový modul odolný vůči narušení (pro uchování citlivých dat).

3.3 Rozdělení bezpečného hardwaru

Bezpečný hardware může být obecně vnímán minimálně ve dvou významech. První kategorii tvoří zařízení, která slouží k přenosu nějakého tajemství skrz nezabezpečené prostředí, k uchovávání informací, které potřebujeme k uložení informací nutných pro práci s IS, a které nejsme schopni si sami zapamatovat. Do této kategorie patří různé magnetické karty, případně čipové paměťové karty. Druhou kategorii tvoří zařízení, která jsou schopna převzít část úloh programů, nebo obecně informačních systémů. Toto jsou zařízení, která nás v dalším výkladu budou zajímat a která budeme nazývat bezpečným zařízením (hardwarem).

Bezpečná zařízení se z pohledu uživatele liší od běžného hardwaru tím, že jsou mnohem jednodušší a díky tomu by mělo být možné provést ověření správnosti jak jejich činnosti, tak i funkčnosti programů, které jsou v jejich rámci implementovány.

Jestliže začneme mluvit o tom, jak používat bezpečný hardware [17, 18], tak před sebou máme dva extrémní případy. Buď můžeme vše nechat v programovém řešení, nebo vše přesunout do bezpečného

zařízení. Jistě cítíme, že takovéto rozdělení funkcí mezi bezpečné a běžné zařízení není ideální, je třeba definovat něco mezi tím. Pokusíme se nyní vytvořit určitou klasifikaci bezpečných zařízení podle jejich schopností, přičemž budeme vycházet z myšlenky P. Gutmanna [9], který se pokusil o rozdělení zařízení podle jejich funkčnosti. Nejde v žádném případě o pokus vytvořit nová kritéria na způsob FIPS-140, ale spíše nástin vlastností, podle kterých je možné rychle získat poměrně přesnou představu, o jaké zařízení jde a jakým způsobem je možné jej použít. V určitém smyslu jde o značně jemnější chápání vlastností než jak je používáno ve FIPS 140, což by ovšem bylo nevýhodou při vytvoření "jednočíselné" stupnice pro vzájemná porovnání zařízení. Na druhou stranu může takovéto rozdělení ulehčit výběr pro konkrétní nasazení, s konkrétními požadavky.

Při takové klasifikaci bezpečných zařízení můžeme použít mnoho kritérií. Prvním a patrně nejdůležitějším kritériem, které nás bude zajímat je funkčnost – schopnosti zařízení. V této oblasti jsme identifikovali čtyři základní stupně:

1. uložení kritických informací;
2. provádění kritických operací (např. kryptografických);
3. provádění složitějších funkcí (typicky komunikační protokol);
4. provádění celých aplikací.

Funkčnost sama o sobě ještě nedeklaruje bezpečnost zařízení, i když je prvním z důležitých parametrů, které určují možnosti jeho použití. Další rovinou, kterou je třeba uvažovat je kontrola přístupu k funkcím bezpečného hardwaru:

1. bez kontroly přístupu;
2. kontrola přístupu k zařízení;
3. kontrola přístupu k jednotlivým funkcím;
4. kontrola přístupu ke vstupním datům funkcí – funkce je možné používat jen na určitá data.

Další podstatnou informací je určitě míra závislosti bezpečného hardwaru na okolí. To je další parametr, který nás musí zajímat při hodnocení vlastností zařízení a který velkou měrou ovlivňuje bezpečnost používání.

1. bez závislosti;
2. vstup a výstup operačních dat;
3. závislost ovlivňující samotnou činnost zařízení (zdroj napájení, hodinový kmitočet, apod.);
4. periferní zařízení.

Takovýchto podobných kritérií je možné vymyslet velké množství. Jaké budou, může být značně závislé na tom, z jaké strany se na činnost bezpečného zařízení budeme dívat. Kromě toho je důležité si uvědomit, že nezáleží jen na tom, co všechno zařízení umí, ale také jakým způsobem je daný rys implementován. U kontroly přístupu se nabízí popis způsobu zadávání autentizačních informací (přes jiný HW, PIN, jiný bezpečný HW připojený přímo k bezpečnému zařízení).

4 Krypto-API

Původní představa bezpečných kryptografických prostředků byla tvořena čipovými kartami, které byly schopny uchovat kryptografické klíče a provádět jednoduché kryptografické operace. Potřeby některých odvětví ovšem vyžadovaly mnohem složitější bezpečná zařízení, což vedlo ke vzniku vysoce bezpečných zařízení (schopnost implementovat celé aplikace, závislost na úrovni vstupních/výstupních dat, kontrola přístupu je zjemněna až na úroveň vstupních dat). Příkladem mohou být certifikační autority, nebo banky a jimi implementované protokoly pro práci s bankovními kartami. Typickým příkladem odpovědi na tyto požadavky jsou např. moduly od IBM (3848, 4758), nebo VSM (Visa Security Modul). Tyto koprocesory

jsou schopny provádět i poměrně složité operace bankovních protokolů, a některé z nich jsou schopny provádět obecné zákaznické aplikace.

Se vrůstající složitostí bezpečných hardwarových modulů je ovšem stále obtížnější provést validaci jak správné činnosti těchto zařízení, tak i funkcí, které jsou v těchto zařízeních implementovány. V tomto kontextu je navíc obrovský rozdíl mezi validací zařízení jako takového a validací aplikací, které jej používají jako prostředku zajišťujícího vysokou bezpečnost. Ukázkovým příkladem může být kryptografický koprocesor IBM 4758. Toto zařízení je ohodnoceno nejvyšším možným stupněm bezpečnosti podle FIPS 140 (Level 4). Kryptografický modul je ovšem dodáván s tzv. Common cryptographic architecture (CCA) [14, 16], což je programové vybavení, které umožňuje bezpečné používání algoritmů DES, 3DES a RSA. Kromě nich ovšem implementuje i složité bankovní operace.

Zákazník obvykle předpokládá, že při používání zařízení s implicitním programovým zařízením se mu nemůže nic stát – je extrémně bezpečné. Opak je ovšem pravdou. Jak ukážeme na dalších řádcích, tak tato konkrétní implementace kryptografických funkcí umožňuje celou řadu útoků. Kdybychom chtěli zabránit těmto útokům režimem přístupu, tak bychom výhody používání bezpečných zařízení značně degradovali.

4.1 Složitost programového vybavení

Jestliže zůstaneme např. u protokolu SET, tak jeho specifikace je popsána na stovkách stran dokumentace. Navíc se ukazuje, že díky způsobu, kterým dokumentace vznikla si jednotlivé dokumenty definující protokol protiřečí [19]. Ačkoliv je nejpřesnější popis v dokumentaci pro programátory, jako referenční je doporučena formální definice protokolu, která je neúplná. Některé problémy však nejsou popsány vůbec nikde (je možné vytvořit různé certifikáty pro jeden klíč, kolik certifikátů může mít jeden klient?). Jestliže se pokusíte vyjmout z implementace veškerá volitelná rozšíření, tak zjistíte, že implementace nebude fungovat, protože některé volitelné informace jsou povinné v jiné části protokolu. Jestliže nejsme schopni formálně ověřit protokol, jakým způsobem pak můžeme ověřovat správnost hardwarové implementace.

Z jedné strany tedy narážíme na omezení v použitelnosti bezpečných prostředků s omezenou funkčností. Z druhé strany ovšem hrozí skryté chyby v implementacích složitých funkcí, jež budou všemi považovány za bezpečné, jen proto, že byly validovány podle uznávaného standardu. Jako příklad je možno uvést protokol Needham-Schroeder [20], který byl dlouho považován za bezpečný, ačkoliv jeho specifikace pokryla méně než deset stran.

Posledním aspektem, který je nutné brát do úvahy je existence dalších standardů, které spoléhají na bezpečná kryptografická zařízení. Jako příklad můžeme uvést oblast elektronického podpisu, kde česká vyhláška definuje takové pojmy jako bezpečná komponenta pro prohlížení podepsaných zpráv, vytváření zpráv, či zobrazení certifikátu. Složitost implementace těchto funkcí je ovšem taková, že ověření jejich implementace by bylo extrémně složité.

Aby mohlo zařízení získat ohodnocení FIPS140 Level 4, tak je nutné vytvořit např. i formální model stavového prostoru. Jinými slovy je třeba vytvořit formální důkaz korektnosti chování zařízení. Tento přístup je ovšem velmi problematický u programů, což je vyjádřeno i nároky na ně – metodický návrh, testování. Formálním dokazováním správnosti programů se poprvé zabýval Dijkstra [15]. Problémem je obrovský nárůst stavového prostoru, který je nutné prozkoumat i u programů v délce desítek řádků. Programové vybavení bezpečných koprocesorů je ovšem mnohem rozsáhlejší a obvykle není prováděna žádná formální, či jiná důkladná, analýza jeho bezpečnosti. Abychom mohli popsat některé z útoků aplikovatelných na programové rozhraní, je třeba si nejprve říci několik slov o základních vlastnostech kryptografických koprocesorů.

4.2 Příkazová sada

Příkazové sady kryptoprocesorů je možné rozdělit na několik částí:

1. výkonné příkazy – tyto příkazy umožňují zpracování dat (šifrování/dešifrování, vytváření/ověřování MAC, či podpisu); příkazy vyžadují použití klíčů, které jsou uloženy v samotném koprocesoru;
2. příkazy pro správu klíčů – zařízení umožňují vytváření struktury klíčů uložených v koprocesoru a import/export klíčů mezi jednotlivými spolupracujícími zařízeními;
3. administrační příkazy – tato množina je značně závislá na konkrétním zařízení, ale minimálně umožňuje počáteční inicializaci zařízení, správu uživatelů, a příp. další nutnou administraci zařízení.

4.3 Kontrola přístupu

Jelikož koprocesory obsahují velmi citlivé údaje, je nutné zajistit, aby se k příkazům, jenž provádějí citlivé operace nad daty, nebo které mohou měnit významným způsobem vnitřní stav zařízení dostali pouze oprávnění uživatelé. Proto je u těchto zařízení implementována duální kontrola, a/nebo schémata sdílení tajemství *m*-z-*n*.

Složitost kontroly přístupu se velmi různí a u nejsložitějších zařízení je možné rozlišit jednotlivé uživatele a přidělovat oprávnění podle jejich funkce. Uživatelé pak získávají oprávnění provádět konkrétní operace implementované v bezpečném zařízení.

4.4 Hierarchie klíčů

Kryptografické koprocesory jsou obvykle využívány pro větší množství činností. Abychom minimalizovali následky případného prozrazení některého z klíčů, tak je obvykle pro každou činnost vytvořen zvláštní klíč. Abychom mohli bezpečně provádět správu klíčů - vytváření nových, import/export existujících klíčů, tak jsou klíče uvnitř těchto koprocesorů spravovány na základě určité hierarchie klíčů. Tato hierarchie se skládá z minimálně tří úrovní:

1. hlavní klíče (master keys) – tyto klíče se nikdy nedostanou ven ze zařízení a jsou používány pro generování dalších klíčů;
2. klíče pro šifrování klíčů (key-encryption-keys) – jestliže chceme vyexportovat některý klíč, např. pro přenesení do jiného bezpečného zařízení, tak tyto klíče použijeme pro jejich zašifrování;
3. operační klíče (operational keys) – klíče, které se používají pro samotné zpracování dat.

Způsob, jakým je možné jednotlivé klíče používat je definován v příslušných kontrolních vektorech. Kontrolní vektor je informace, popisující vlastnosti příslušného klíče a je s tímto klíčem jednoznačně svázána, tj. jsme schopni zjistit, jestli patří k určitému klíči, nebo zda došlo ke změně kontrolního vektoru.

5 Útoky

V této části se budeme zabývat několika typy útoků, jež je možné použít na různé existující kryptografické moduly. Jsou to útoky, které byly objeveny na nejvyšší úrovni softwarového vybavení kryptografických modulů. Chyby vedoucí k popsaným útokům byly nalezeny zkoumáním vztahů mezi jednotlivými funkcemi kryptografických modulů, aniž by byla prováděna analýza jejich vlastní implementace. Následující útoky pocházejí v převážné míře z university v Cambridge [10, 11, 12, 13].

5.1 Klíče se vztahem

Pojmem klíče se vztahem označujeme klíče, u nichž je znám jejich vzájemný vztah. Díky takovéto znalosti je útočník schopen při objevení, jednoho klíče schopen snadno odvodit klíč jiný, mnohdy

mnohem citlivější. Takovýto vztah může umožnit odvození i jen části druhého klíče. V tomto případě pak útočník může použít útok hrubou silou na mnohem slabší klíč. Jestliže je takový vztah znám pouze útočníkovi, tak není možné ho žádným způsobem odhalit.

Znalost vztahů mezi klíči je možné využít např. pro útok přetypováním. Jelikož některé systémy používají kombinování kontrolního vektoru s klíčem např. pomocí operace XOR, je možné při vhodné volbě vztahu s tím související změně kontrolního vektoru vložit útočníkovi známý klíč, který je ovšem koprocesorem považován za bezpečný.

Snadnou cestou vytvoření vztahu mezi klíči je využití mechanismu přenosu klíče rozděleného na několik částí pomocí operace XOR. Při dvojím přenosu téhož klíče lze pak snadno vytvořit klíče s předem zvoleným vztahem.

5.2 Vázání částí klíče

Jestliže jsou klíče větší než velikost bloku algoritmu, který je určen pro jejich ochranu, může být zvolen postup, kdy jsou chráněné klíče rozděleny na několik částí, které pak mohou být používány samostatně.

Konkrétním příkladem mohou být 3DES klíče, které jsou šifrovány algoritmem DES. V této situaci je 3DES klíč rozdělen na dvě části, se kterými je možné samostatně manipulovat. Jestliže se nám podaří jednu z částí nahradit klíčem podle vlastní volby, nebo se známou hodnotou, tak efektivně snižujeme náročnost útoku hrubou silou na původní silný klíč. V [21] je popsán postup, který najde 56b klíč algoritmu DES v rozmezí několika hodin (7 až 35).

V některých případech může kryptoprocesor umožnit požití silnějšího algoritmu tak, aby byl kompatibilní se slabší verzí. U algoritmu DES je možné dvojím použití jednoho bloku degradovat 112b klíč na rovnou polovinu 56 bitů. 3DES totiž funguje podle schématu $E_{k1}(D_{k2}(E_{k1})) = E_k$, jestliže $k=k_1=k_2$.

5.3 Víceúčelový typ

Tento útok byl nalezen při analýze vlastností modulu VSM. Základem útoku je zjištění, že jeden typ klíčů (kontrolní vektor) je použit u klíčů určených ke dvěma rozdílným činnostem. Konkrétně jde o šifrování komunikačního klíče a výpočet PINu pro čísla účtů.

Jestliže potřebujeme do bankomatu přenést klíč pro šifrování spojení mezi bankomatem a bankou (KC), tak jej zašifrujeme hlavním klíčem terminálu. KC slouží k ochraně integrity přenášených dat a na jeho použití nejsou kladeny žádné požadavky – může být použit k šifrování/dešifrování jakékoliv zprávy.

Další dvě vlastnosti, které potřebujeme, jsou možnost vložit čistý KC do modulu, který jej vrátí zašifrován hlavním klíčem modulu (KMC) a možnost změnit již ustavený KC (obvyklá činnost, která může být prováděna pravidelně). Abychom mohli v bankomatu změnit klíč KC, je třeba ho zašifrovat klíčem, který je i v bankomatu. Existuje tedy operace, která (uvnitř modulu) odšifruje KC a zašifruje ho libovolným hlavním klíčem terminálu (každý bankomat-terminál má vlastní klíč).

Poslední vlastností, kterou potřebujeme je právě fakt, že klíč pro odvození PINů (KP) může být exportován jako hlavní klíč terminálu, protože oba mají stejný kontrolní vektor. Nyní můžeme započít útok, kterým získáme PIN k libovolnému číslu účtu, aniž bychom k tomu měli oprávnění.

Připomeňme, že $PIN=E_{KP}\{PAN\}$, kde PAN je číslo účtu.

1. Nejprve vyexportujeme klíč pro výpočet PINu, který je samozřejmě zašifrovaný - $E_{KM}\{KP\}$.
2. Pak vložíme PAN jako nový KC, takže po zašifrování získáme $E_{KMC}(PAN)$.
3. Posledním krokem necháme modul, přešifrovat PAN novým klíčem, ale použijeme vyexportovaný $E_{KM}(KP)$. Tím získáme $E_{KP}(PAN)$, což je PIN k danému číslu účtu.

Operace, které jsou v tomto útoku použity jsou skutečně implementovány. Důvody jejich implementace jsou jak nutnost zajistit efektivní používání modulu, tak ale i nepříliš rozumná snaha řešit jiné systémové problémy. K těm druhým patří např. předcházení případné nespolehlivosti sítě pro ověřování PINů karet. Aby se předešlo problémům s ověřováním PINů při výpadku části bankovní sítě, tak se zajistí aby ověření PINů karet každé banky bylo schopno provést několik modulů.

5.4 Přetypování

Přetypování je typ útoku, který využívá buď vlastností zajišťujících kompatibilitu se staršími zařízeními, nebo znalosti vztahů mezi klíči. Bodem útoku je zde opět špatné zajištění vazby kontrolní vektor – klíč.

Příkladem může být dvoufázový útok využívající dvou způsobů importu klíčů. V první fázi je naimportován klíč s použitím operace XOR pro rozdělení klíče na několik částí – v tomto případě neexistuje kontrolní vektor, takže libovolný z účastníků přenosu klíče může změnit klíč podle volby s předem známým výsledkem. V druhé části, při importu klíčů s kontrolním vektorem (PINKEY, DATAKEY) využijeme toho, že v jednom zařízení je klíč změněn.

1. 1. $K_{ORIG} = KP_A \oplus KP_B \oplus KP_C$ - $K_{MOD} = KP_A \oplus KP_B \oplus KP_C \oplus CV_{MOD}$
   CV_{MOD} je rozdíl mezi přenášeným a požadovaným kontrolním vektorem
2. $E_{KORIG \oplus PINKEY}(PKEY)$, PINKEY – způsob přenosu klíčů u modulu IBM 4758, kde PINKEY je kontrolní vektor
   $E_{KORIG \oplus PINKEY}(PKEY)$, DATAKEY = $E_{KMOD \oplus CVMOD \oplus PINKEY}(PKEY)$, DATAKEY =
   = $E_{KMOD \oplus DATAKEY}(PKEY)$, DATAKEY

Výsledkem je, že klíč pro kontrolu PINů je v novém zařízení uložen jako klíč pro šifrování dat. Takový klíč je ovšem možno použít k šifrování libovolných dat, tedy i libovolných čísel účtů. Takto jsem schopen vypočítat PIN pro libovolný účet podle volby.

Hlavním problémem je, že útočník je schopen změnit výsledný složený klíč podle svého přání. Abychom tomuto zabránili, je potřeba použít nové schéma pro rozdělení tajemství – klíče. Jedním z příkladů může být schéma, kdy tajemství zašifrujeme *n-1* klíči a poslední část je tvořena výsledkem šifrování – to je počáteční hodnota pro složení klíče: $S_3 = E_{S2}\{E_{S1}\{K\}\}$.

Asi nejdůmyslnější útok, jaký byl zatím objeven pro modul IBM 4758 vyžaduje minimální oprávnění pro přístup ke kryptomodulu (vlastní je poměrně široký okruh osob) a nastavení příznaku exportovatelnosti klíčů. Výsledkem útoku je získání libovolného exportovatelného klíče za předpokladu 10 minut přístupu k zařízení a zhruba 2 dnů potřebných pro výpočet DES klíčů a získání veškerého citlivého exportovatelného materiálu [22].

Není ani tak znepokojivé, že byly objeveny útoky na bezpečná zařízení, mnohem horší se mi zdá minimum informací, které k nim bylo potřeba získat. Když si projdete ještě jednou krátký výběr útoků, které jsme stručně popsali, tak udivuje, že i nové, vysoce bezpečné produkty trpí stále stejnými problémy, které se vyskytly již před mnoha lety. Nebylo potřeba zabývat se vnitřní implementací zařízení (jako je tomu např. u útoků na operační systémy, nebo běžné aplikace), stačilo pouze využít definovaných rozhraní.

6 Závěr

Systémy se obecně neskládají pouze z jediného produktu či jen certifikovaných produktů. A právě chyby v rámci integrace bezpečnostních politik jednotlivých částí systémů vedou k novým možnostem potenciálních útoků. Obecně lze pro hardwarové prvky v porovnání se softwarovými komponentami snadněji sestavit formální návrh a otestovat jej (respektive v částce určené na vývoj hardwarového

produktu netvoří formální metody návrhu a verifikace tak zásadní položku, jako je tomu v případě nízkorozpočtových softwarových projektů).

Pouhým nasazením hardwarového prvku nelze zajistit úplně bezpečný systém. Jeho primárním účelem je výrazné znesnadnění a prodražení potenciálního útoku. Při současném stavu technologií lze reverzní inženýrství libovolného hardwarového prvku úspěšně provést, avšak za cenu velmi vysokých nákladů (finančních, speciálních znalostí atd.). Při analýze rizik je nutné jako klíčový prvek uvažovat hodnotu chráněných dat a na tomto základě volit způsob zabezpečení.

Četné příklady, z nichž některé jsme uvedli, však ukazují, že mnohdy ani není potřeba provádět vysoce náročné destruktivní útoky. Tak jako byly objeveny efektivní nedestruktivní útoky na čipové karty (zařízení poměrně málo bezpečná), tak existují "levné" útoky i na zařízení s nejvyšší bezpečností. Tato zařízení totiž používají kryptografické knihovny, které nedokážou plně využít vlastností těchto zařízení a dokonce degradují jejich bezpečnostní vlastnosti.

Jestliže je bezesporné, že formální dokazování programů je velmi obtížným problémem, tak díky komerčním tlakům, které požadují řešení "all in one" se neřešitelným problémem stalo dokázání správnosti programů na úrovni jejich rozhraní. Smutným následkem tohoto faktu je, že zařízení s jinak vysokou bezpečností jsou degradována jak co do použitelnosti, tak i obecné důvěryhodnosti. Co to je za bezpečnost, když existuje tolik úspěšných útoků na zařízení, které je tak bezpečné jak jen vůbec může být.

V tomto kontextu se zdá být nutností vybírat bezpečná zařízení podle požadavků a nároků konkrétních aplikací. Pro tento účel by mohlo být užitečné používat i rozdělení bezpečných zařízení podle jejich schopností a vlastností tak, jak bylo principiálně popsáno. Současně s tím se ale zdá nutností klást mnohem vyšší nároky na programové vybavení těchto zařízení.

7 Reference

[1] *Information technology – Security techniques – Evaluation criteria for IT security, ISO/IEC 15408*

[2] *Information Technology Security Evaluation Criteria*, available at http://www.cordis.lu/infosec/src/crit.htm

[3] *Trusted Computer System Evaluation Criteria*, available at http://csrc.nist.gov/secpubs/rainbow/std001.txt

[4] L. Novák, *Společná kritéria*. DSM 2/2001

[5] *Common Evaluation Methodology*, available at http://www.commoncriteria.org/cem/cem.html

[6] R. Anderson, M. Kuhn, *Tamper Resistance – a Cautionary Note*. Sborník 2nd USENIX Workshop on Electronic Commerce, USENIX Association, 1996.

[7] http://www.cl.cam.ac.uk/Research/Security/Trust-Register/book.html

[8] http://www.sandelman.ottawa.on.ca/spki/html/1998/winter/msg00077.html

[9] Gutmann P. *An Open-source Cryptographic Coprocessor*, The Proc. of the 10th USENIX Conference, 2000.

[10] Bond M., *Attacks on Cryptoprocessor Transaction Sets*, The Proceedings of the 3rd International Workshop on Cryptographic Hardware and Embedded Systems (CHES 2001), Paris, France, May 14-16, 2001, pp. 220-234.

[11] Bond M., Anderson R. *API-Level Attacks on Embedded Systems*. Computer, Vol. 34, No. 10, October, 2001, pp. 67-75.

[12] Anderson R, *The Correctness of Crypto Transactions Sets*. Security Protocols – 8th International workshop, April, 2000, pp. 125-127.

[13] Bond M., Anderson R. *API-attacks on Embedded systems*.

[14] IBM, *IBM 4758 PCI Cryptographic Coprocessor – CCA Basic Services Reference and Guide, Release 1.40*. available through http://www.ibm.com/security/cryptocards

[15] Dijkstra E.W. *Discipline of Programming*. Prentice Hall, Eaglewood Cliffs, New Jersey, 1976

[16] Dyer J., Perez R., Smith S., Lindemann M., *Application Support Architecture for a High-Performance, Programmable Secure Coprocessor*. 22nd National Information Systems Security Conference, October 1999.

[17] Smith S., *Secure Coprocessor Applications and Research Issues*. Los Alamos, Technical Report LA-UR-96-2805, 1996.

[18] Yee B., *Using Secure Coprocessors*, Carnegie Mellon University, PhD thesis, CMU-CS-94-149, 1994.

[19] Bella G., Massacci F., Paulson L. C., Tramontano P., *Making Sense of Specification: The Formalization of SET (extended abstract)*. Lecture Notes in Computer Science, Security Protocols 2000, Cambridge.

[20] Needham R. M., Schroeder M., *Using Encryption for Authentication in Large Networks of Computers*. Communications of the ACM, 21(12):993-999, 1978.

[21] Bond M., Clayton R., *A Low Cost Hardware Birthday Attack on DES*. presentation from Computer Laboratory seminar, University of Cambridge, United Kingdom.

[22] http://www.cl.cam.ac.uk/~rnc1/descrack/