Implementace kryptografického protokolu s využitím mobilní kryptografie

Petr Švenda

<xsvenda@fi.muni.cz>

Fakulta Informatiky
Masarykova universita, Brno

Abstrakt

Příspěvek se zabývá implementací autentizačního protokolu a protokolu pro výměnu důvěrných dat mezi stranami A a B v situaci s upravenými předpoklady útočníkových možností. Strana A může být v souladu se standardním modelem útočníka ovlivňována pouze prostřednitvím manipulace příchozích a odchozích zpráv protokolu. Předpoklady strany B jsou oslabeny. Předpokládáme, že kroky protokolu strany B jsou prováděny softwarovou aplikací, která je vykonávána ve výpočetním prostředí pod kontrolou útočníka. Dále předpokládáme, že útočník může sledovat vykonávané instrukce procesoru a číst paměť používanou aplikací. Bezpečnost probíhajícího protokolu tak může být narušena nejen analýzou a manipulací vyměňovaných zpráv, ale i manipulací samotného procesu zpracování a vyhodnocení zpráv protokolu stranou vykonávanou v prostředí pod kontrolou útočníka.

Cílem příspěvku je využít konceptu mobilní kryptografie [1] pro ochranu kódu aplikace, která provádí vyhodnocení protokolu. Implementace vychází z protokolu ISO9798-2 pro vzájemnou autentizaci sdíleným klíčem symetrické kryptografie a využivá alternativní implementaci šifrovacího algoritmu AES (White-Box Attack Resistant AES, navržená v [2], dále WBACRAES), která umožňuje ukrýt hodnotu používaného klíče a zpracovávaných dat i v případě, že má útočník přístup k průběhu šifrování.

Vlastní přínos je v návrhu mechanismů, které umožňují využít možnosti WBACR AES pro implementaci protokolu zajišťujícího autentizaci, důvěrnost a čerstvost dat tak, aby napadení aplikace s touto implementací bylo obtížnější. Jedná se o odstranění nutnosti provádět v průběhu protokolu porovnávací rozhodnutí, které jsou útočníkem snadno modifikovatelná. Je zavedena kontrola čerstvosti přijatých zpráv, která je obtížněji odstranitelná než použití testu na očekávanou hodnotu keksíku. Je navržen postup generování vstupního a výstupního kódování pro WBACRAES použitelný pro CBC režim, který umožňuje hlubší integraci s okolním aplikačním kódem. Kombinací WBACR AES a metody pro tvorbu hashovací funkce z blokového šifrovače je navržena implementace klíčovaného hashovacího algoritmu umožňující utajení hodnoty použitého klíče. Výsledkem je popis protokolu SEAUT (SEcure AUthenticated Transport protokol) včetně jeho doporučené implementace na straně umístěné v potencionálně nebezpečném prostředí.

Klíčová slova: autentizační protokol, mobilní kryptografie, obfuscation, softwarový agent, White-Box Attack Resistant AES.

1 Úvod

Mnoho reálných situací vyžaduje autentizaci a výměnu důvěrných dat mezi stranami, z nichž jedna je vykonávána v prostředí, které je pod kontrolou možného útočníka. Příkladem je využití autonomní softwarové aplikace pro rozhodování a správu přístupu k digitálním objektům v rámci Digital Rights Management [3] ve výpočetním prostředí koncového uživatele. Během rozhodování dochází ke komunikaci mezi aplikací vykonávané na zabezpečeném serveru poskytovatele digitálního objektu a autonomní softwarovou aplikací (dále softwarový agent) na počítači, jehož uživatel disponuje plnými právy pro přístup ke všem jeho výpočetním prostředkům a lze předpokládat, že má zájem na narušení korektnosti běhu softwarového agenta. Pro ochranu vyměňovaných dat má pak význam nejen odolnost protokolu proti útočníkovi s přístupem k vyměňovaným zprávám, ale i způsob implementace zpracování protokolu v kódu softwarového agenta. Dalším příkladem je spolupráce aplikace vykonávané na kryptografické čipové kartě (bezpečné prostředí) komunikující se softwarovým agentem na počítači pod možnou kontrolou útočníka. Provedení funkčnosti (použití podpisového klíče apod.) je vázáno nejen na zadaní správného PINu uživatelem, ale i na autentizaci sofwarového agenta, který data k podpisu zasílá.

Bezpečnost výpočetního prostředí je ve většině případů relativní vzhledem k prostředkům, které musí útočník vynaložit na její prolomení. Přesto však má toto dělení smysl v případě, že komunikující strany se nacházejí v prostředích s různou úrovní zabezpečení, z nichž jednu pokládáme za dostačující vzhledem k hodnotě zpracovávaných informací (strana A), zatímco druhou nikoli (strana B). Zavádíme následující model útočníkových možností: V souladu s běžnými předpoklady kladenými na kryptografický protokol má útočník přístup ke všem vyměňovaným zprávám mezi stranou A a B. Strana A se vůči útočníkovi jeví jako „černá skříňka“, kterou může ovlivňovat pouze manipulací příchozích a odchozích zpráv. Předpoklady kladené na stranu B jsou však oproti straně A oslabené. Předpokládáme, že útočník může provádět statickou a dynamickou inspekci programového kódu reprezentující stranu B s využitím ladících nástrojů (dissasembler, debugger), čtení a modifikaci kódu nebo dat a podvržení systémových funkcí a další. Pro prostředí tohoto typu přípěvek identifikuje místa snadno napadnutelná útočníkem a s využitím mobilní kryptografie navrhuje modifikace, které by prakticky znemožnily nebo alespoň ztížily napadnutelnost. V následujícím textu bude presentován protokol i jeho doporučená implementace na straně B s ohledem na specifické prostředí, ve kterém se nachází. Na implementaci zpracování protokolu na straně A nejsou kladeny žádné dodatečné požadavky, neboť se nepředpokládá, že útočník bude mít k průběžným hodnotám přístup. Například použité šifrovací klíče mohou být representovány polem s otevřenou hodnotou a podobně.

Důležitím prvkem je nezávislost softwarového agenta na jeho uživateli. Softwarový agent si sám nese svoje autentizační informace a v průběhu protokolu nekomunikuje s uživatelem, v jehož prostředí je spuštěn. Cílem je naopak autentizační informace softwarového agenta před uživatelem utajit resp. zajistit jejich integritu tak, aby je uživatel nebo zlovolný software na uživatelově počítači nemohl využít ve vlastní režii. Vycházíme z předpokladu, že WBACR AES (viz 3 umístěný v kódu softwarového agenta poskytuje dostatatečnou ochranu používané hodnotě klíče, nechrání však další hodnoty používané v průběhu protokolu. Mezi tyto hodnoty patří příznak o výsledku proběhlé autentizace, hodnoty keksíku pro zajištění čerstvosti a měnící se hodnota klíče relace, který nemůže být implementován formou WBACR AES. Navržená implementace protokolu nezajišťuje absolutní odolnost proti útočníkovi, měla by však zvýšit množství nutných změn v kódu pro úspěšný útok. Více změn vyžaduje více útočníkova času a zvyšuje pravděpodobnost, že dojde k detekci změny doplňkovými integritním mechanismy (například [5, 6]) pokud jsou použity.

Dělení textu je následující: Kapitola 2 uvádí funkční požadavky kladené na hledaný protokol a stanovuje vlastnosti, které by měla mít jeho vhodná implementace straně B. V kapitole 3 jsou shrnuty základní vlastnosti implementace šifrovacího algoritmu AES navržené v [2]. Kapitola 4 popisuje autentizační a transportní protokol navržený pro bezpečnou komunikaci mezi stranou umístěnou v bezpečném prostředí (chráněný server, kryptografická čipová karta) a stranou umístěnou ve výpočetním prostředí pod potenciální kontrolou útočníka (softwarový agent). V kapitole 5 jsou podrobněji rozebrány stavební prvky použité pro implementaci tohoto protokolu na straně B.

2 Požadované vlastnosti protokolu

Pro potřeby komunikace mezi stranami A a B byly stanoveny požadavky na vhodný protokol tak, aby zajišťoval vzájemnou autentizaci komunikujících stran i autentizaci, utajení a čerstvost vyměňovaných dat. Tyto požadavky odpovídají potřebám pro komunikaci typu klient/server. Nejprve je provedena (vzájemná) autentizace, na základě které se server rozhoduje, zda umožní klientovi přístup ke svým zdrojům. Klient se ujistí, že komunukuje s požadovaným serverem. Následně klient zasílá požadavky serveru, který po zpracování zasílá zpět odpověď.

Pokud bychom předpokládali, že útočník má možnost ovlivňovat komunikující strany A a B pouze manipulací vyměňovaných zpráv, pak existuje široké spektrum kryptografických protokolů, které výše uvedené požadavky splňují. Pro autentizaci lze využít protokolů typu výzva-odpověď založených na symetrické nebo asymetrické kryptografii. Autentizaci dat lze zajistit digitálním podepisováním nebo přidáním autentizačního kódu zprávy (MAC). Utajení dat lze dosáhnout šifrováním, čerstvosti použitím „keksíků“ na bázi čítače, náhodného čísla nebo časového razítka.

Oslabení předpokladů kladených na stranu B tak, že útočník může navíc pozorovat/modifikovat prováděné operace na úrovni instrukcí procesoru a kontrolovat využívanou paměť ukazuje, že implementace zvoleného protokolu běžným způsobem nepostačuje, pokud nejsou explicitně nechráněné hodnoty používaných klíčů a integrit procesu zpracování vstupních a výstupních zpráv protokolu. Útočník může provést tyto základní kategorie útoků:

- *Odhalení citlivých dat* – Pomocí disassembleru je lokalizováno přibližné místo použití, pomocí debuggeru během dynamické inspekce je z oprační paměti přečtena otevřená hodnota. Typicky jde o autentizační informace a hodnoty šifrovacích klíčů.

  Obranou je práce s citlivými daty D tak, aby nebyla nikdy přístupná v paměti v otevřené podobě, ale pouze v tvaru $G_{IOC}(D)$ transformovaném pomocí funkce G_{IOC}, kde G_{IOC} není známa útočníkovi. Je nutné příslušně upravit operace, které s takto transformovanými daty manipulují. Lze použít technik mobilní kryptografie známých jako počítání se šifrovanými daty a počítání se šifrovanou funkcí[1]. Implementace navrženého protokolu takto pracuje s hodnotami autentizačních klíčů (viz 3), hodnotou klíče relace (viz 5.3) a daty určenými k zašifrování (viz 5.1).
- *Modifikace vyhodnocení průběhu protokolu* - Cílem útočníka je úprava kódu vyhodnocující vyměňované zprávy tak, aby byl porušen některý z výše uvedených cílů protokolu, například autentizace strany, která ve skutečnosti nezná autentizační informace nebo zpracování příchozí zprávy jako čerstvé, přestože jde o opakovaně zaslanou zprávu. Modifikována mohou být dlouhodobě nesená data softwarového agenta, například veřejný klíč P_A je nahrazen klíčem $P_{A'}$, ke kterému útočník vlastní privátní klíč $S_{A'}$. Modifikována mohou být průběžná data jako pořadový čítač zpráv nebo příznak udávající stav autentizace. Modifikovány mohou být instrukce kódu, typicky instrukce porovnání (vůči očekávané hodnotě) a přiřazení (výsledku autentizace).

  Navržená implementace protokolu nevyžaduje porovnávací operace v autentizační části (viz 4.1). Nahrazení hodnot klíčů je ztíženo použitím vstupního a výstupního kódování WBACR AES tabulek (viz 3).
- *Využití části kódu ve vlastní režii* - Útočník lokalizuje v kódu softwarového agenta strany B úsek, ve kterém dochází k aplikaci funkce f na data D, například aplikace autentizačních informací na autentizační výzvu. Úsek vyjme a nahrazením vstupních dat D se autentizuje vůči straně A ve vlastní režii. Útočník je tak schopen vytvářet korektní odpovědi i bez znalosti obsažené funkce f. Typickým cílem v tomto kontextu jsou autentizační funkce a funkce pro šifrování/dešifrování vyměňovaných dat.

  Navržená implementace ztěžuje extrakci kódu zavedením vstupního a výstupního kódování pro WBACR AES tabulku použitelné v CBC režimu (viz 5.1) a zavedením „směrových“ klíčů (viz 4.1).

Vhodná implementace (zvoleného protokolu) by proto měla splňovat následující podmínky:

- Strana B neprovádí žádné porovnávací operace vzhledem k očekávaným hodnotám.
- Autentizační informace strany B a informace sloužící k utajení vyměňovaných dat nejsou čitelné při použití statické (disassembler) a dynamické (debugger) inspekce.
- Strana B explicitně zajišťuje integritu keksíku používaného pro kontrolu čerstvosti.
- Strana B neobsahuje funkci generující zprávy zaměnitelné za zprávy pocházející od strany A, a to ani v případě, že útočník manipuluje se vstupními daty této funkce.

3 White-Box Attack Resistant AES

Pro ochranu kódu a dat vykonávaných ve výpočetním prostředí pod kontrolou útočníka lze využít konceptu mobilní kryptografie [1], který umožňuje provést implementaci některých tříd funkcí[1] tak, že je výpočetně obtížné získat hodnotu použitých dat i v případě, že má útočník možnost monitorovat vykonávané instrukce procesoru.

Tato myšlenka je využita pro implementaci šifrovacího algoritmu AES [7] navržené v [2] (dále WBACR AES). Umožňuje šifrovat v prostředí pod kontrolou útočníka tak, aby zisk používané hodnoty klíče byl výpočetně obtížný. Konkrétní implementace WBACR AES může šifrovat pouze jednou, předem danou hodnotou klíče K. Na základě hodnoty K jsou vygenerovány tabulky o celkové velikosti 600KB (šifrovací i dešifrovací směr) s předpočtenými hodnotami odpovídajícími použití klíče K. Tabulky jsou dvojího typu. První typ provádí operace AddRoundKey(), SubByte() a část MixColumn() v rámci jedné rundy. Druhý typ dokončuje operaci MixColumn(). Hodnota expadovaného klíče K se v průběhu rund liší. Proto jsou pro každou z 10-ti rund přítomny odlišné tabulky. Vstupní data jsou transformována pouze prostřednictvím série operací „náhledu“ do těchto tabulek na základě aktuální hodnoty x části vstupu. Operace „náhledu“ do předpočtené tabulky pro operaci f se rozumí nahrazeni hodnoty x

[1] V současné době polynomiální a racionální funkce.

hodnotou $x' = f(x)$, která se nachází na x-tém řádku tabulky. Po posledním náhledu je vstupní blok zašifrován klíčem, který byl použit pro generování tabulek.

Průběžné hodnoty x a x' mezi jednotlivými náhledy jsou přístupné útočníkovi. Ze znalosti hodnot x a x' lze dedukovat informace o funkci f. Pro ztížení dedukce je využito *vstupního a výstupního kódování* (dále IOC). Během generování tabulky pro funkci f je zvolena náhodná bijekce G_{IOC} a spočtena její inverze G_{IOC}^{-1}. Na x-tý řádek předpočtené tabulky pro f je namísto hodnoty $f(x)$ umístěna hodnota $G_{IOC}(f(x))$. Výsledkem náhledu tak není přímo $f(x)$, ale $G_{IOC}(f(x))$. Mějme nyní funkci g, která je aplikována na výsledek funkce f a kterou chceme realizovat taktéž předpočtenou tabulkou. Na řádek y-tý řádek tabulky pro funkci g umístíme hodnotu $g(G_{IOC}^{-1}(y))$. Složením získáme platnost následující rovnice: $g(f(x)) = g(G_{IOC}^{-1}(G_{IOC}(f(x)))$. V tomto příkladě je bijekce G výstupním kódováním tabulky pro funkci f a inverze G_{IOC}^{-1} vstupním kódováním tabulky pro funkci g. Tabulka pro f nemá vstupní kódování[2] a tabulka pro g nemá výstupní kódování. Je zřejmé, že IOC lze snadno rozšířit na libovolné množství po sobě jdoucích funkcí.

3.1 Výhody a nevýhody

Šifrování využívající WBACR AES má oproti standardní implementaci několik předností:

1. *Utajení hodnoty použitého klíče* – zpětný zisk hodnoty klíče je výpočetně obtížný i při znalosti vygenerovaných tabulek. Otevřená hodnota průběžných výsledků během série náhledů je přítomna pouze v transformovaném tvaru pomocí IOC.
2. *Oddělitelná šifrovací a dešifrovací funkčnost* – vygenerované tabulky pro šifrování a dešifrování jsou navzájem nezávislé. Lze zavést analogii asymetrické kryptografie. Šifrovací část tabulek pro klíč K je použita jako „privátní klíč“, dešifrovací část slouží jako „veřejný“ klíč distribuovaný ostatním aplikacím. Proces vytváření digitálního podpisu zprávy je nahrazen časově méně náročnou tvorbou autentizačního kódu (MAC). Podrobnější srovnání s asymetrickou kryptografií a výhod tohoto řešení je uvedeno v 6.2.
3. *Vstupní a výstupní kódování* – vstupním kódováním je myšlena bijektivní transformace G_{IOC} vstupního bloku dat, která je v rámci náhledů do předpočtených tabulek pro první rundu pomocí inverzní transformace G_{IOC}^{-1} (zahrnuté v tabulkách) odstraněna. Transformace G_{IOC} je volena náhodně během generování tabulek. Před začátkem šifrování musí být data přítomna v transformovaném tvaru pomocí G_{IOC}, jinak je výsledek šifrování chybný. Analogicky pro výstupní kódování aplikované v rámci náhledů poslední rundy, které je nutno před použitím výstupních dat inverzní transformací G_{IOC}^{-1} odstranit. Použití vstupního anebo výstupního kódování ztěžuje možnost samostatného použití tabulek extrahovaných z kódu softwarového agenta. Útočník musí získat i předpis použité bijektivní transformace F, což zvyšuje celkovou provázanost kódu. Zároveň je poskytnut základ pro „napojení“ dalších operací ve stylu mobilní kryptografie.

Nevýhodou použití může být naopak větší velikost implementace (cca 600kB předpočtených tabulek pro šifrování i dešifrování daným klíčem) a nemožnost dynamické změny hodnoty klíče po vygenerování tabulek. Je vhodné poznamenat, že předpočtené tabulky mohou být používaný jako „šifrovací stroj“ i bez znalosti obsaženého klíče, proto je vhodné doplňkovými nástroji typu *obfuscation*[3][10] zvýšit obtížnost jejich extrakce.

3.2 Bezpečnost

Bezpečnost šifrování pomocí WBACR AES je vhodné zvážit ve třech základních kategoriích a dále dle konkrétního způsobu použití.

První kategorií je bezpečnost zašifrovaných dat pomocí WBACR AES proti útočníkovi, který nevlastní WBACR AES tabulky ani klíč použitý k jejich generování. V tomto případě je bezpečnost ekvivalentní standardní implementaci algoritmu AES se 128bitovým klíčem a 128bitovým blokem. WBACR AES je pouze variantou implementace algoritmu AES, nikoli novým algoritmem.

Druhou kategorií je situace, kdy jsou WBACR AES tabulky umístěné v kódu softwarového agenta a útočník se je snaží extrahovat. Zde bude bezpečnost závislá na použitém typu *obfuscation* transformací,

[2] Resp. identitní: $\forall x : G_{IOC}(x) = x$

[3] Sémanticky invariantní transformace kódu a dat, která ztěžuje zpětnou rekonstrukci funkčnosti kódu a zisk hodnoty dat.

obecně však lze říci, že vzhledem k velikosti tabulek a omezení operací pouze na náhledy do tabulek, bude *obfuscation* poměrně dobře proveditelná. Extrahované tabulky může útočník použít jako šifrovací stroj bez znalosti šifrovacího klíče. Toto použití lze ztížit zavedením vstupního a výstupního kódování.

Třetí kategorií je případ, kdy se již útočníkovi podařilo tabulky extrahovat a jeho cílem je zisk hodnoty klíče, který byl použit pro jejich generování. V současné době není znám žádný výpočetně proveditelný útok, který by umožnil odstranit interní kódování a vedl tak ke kompromitaci hodnoty uchovávaného klíče. Bližší rozbor bezpečnosti WBACR AES lze nalézt v [2]. Možným ohrožením by mohlo být využití varianty diferenciální chybové analýzy presentované v [13] v případě, že není použito vstupního ani výstupního kódování. Konkrétní aplikace tohoto typu útoku je předmětem dalšího studia.

4 SEcure AUthenticated Transport protokol

Stranu umístěnou v bezpečném prostředí označme A, stranu umístěnou v prostředí pod možnou kontrolou útočníka (softwarový agent) označme B. Navržený protokol SEAUT se skládá ze dvou částí, autentizační a transportní.

4.1 Autentizační část

Autentizační část navrženého protokolu je založena na 3-průchodovém protokolu ISO9798-2 (dále P3-ISO9798-2) pro vzájemnou autentizaci [11]. Je použito následující značení: id_A resp. id_B jednoznačná identifikace strany A resp. B; N_A resp. N_B kryptograficky náhodné číslo generované stranou A resp. B; $E_{KX_Y}(data)$ operace šifrování dat pomocí klíče KX_Y; $V(X)$ hodnota proměnné obsahující uloženou hodnotu X; $H_{K_x}(data)$ operace jednosměrné klíčované hashovací funkce nad daty s použitým klíčem K_x. K_{R_0} iniciální hodnota klíče K_R.

Vyměňované zprávy (autentizační část):
1. $A \leftarrow B : \{id_B, N_B\}$.
2. $A \rightarrow B : \{id_A, E_{KD_{auth}}(N_A, N_B, id_B)\}$.
3. $A \leftarrow B : \{E_{KE_{auth}}(N_B, N_A)\}$.
4. $K_{R_0} = H_{K_1}(N_A|N_B|id_B|V(N_B)|V(id_B))$

Protokol SEAUT používá klíče KE_{auth} a KD_{auth} na místo jednoho klíče protokolu P3-ISO9798-2. Pro šifrování 2. zprávy je použit klíč KD_{auth}, pro 3. zprávu KE_{auth}. Lze snadno nahlédnout, že strana B využívá klíč KE_{auth} pouze pro zašifrování a klíč KD_{auth} pouze pro dešifrování (opačně pro stranu A). Cílem této úpravy je zajistit, aby na straně B nemusela být přítomna funkčnost pro dešifrování klíčem KE_{auth}. Klíč KE_{auth} tak může být na straně B realizován pouze šifrovací částí WBACR AES tabulek (viz 3). Analogicky pro klíč KD_{auth}.

Další modifikací je způsob vyhodnocení korektnosti provedené autentizace na straně B. Kontrola shody keksíku N_B a identifikace id_B vůči očekávaným hodnotám $V(N_B)$ a $V(id_B)$ z 1. a 2. zprávy je prováděna jen nepovinně, neboť ji útočník může snadno modifikovat nebo odstranit. Namísto toho je ze zaslaných i obdržených hodnot pomocí jednosměrné klíčované hashovací funkce H_{K_1} (viz 5.3) vytvořena iniciální hodnota klíče relace K_{R_0} (číselný index u klíčeK_R je zaveden z důvodu pravidelné změny hodnoty K_R, viz. 4.2). Klíč relace K_R bude ustanoven i v případě nekorektní autentizace strany A vůči straně B, bude však odlišný od klíče vzešlého z korektní autentizace. Použití klíčované hashovací funkce (s využitím WBACR AES) pro jeho tvorbu zabraňuje útočníkovi odvodit jeho přímou hodnotu. Vzhledem ke způsobu použití K_R v transportní části pak nekorektní K_R povede chybnému zpracování vyměňovaných zpráv v případě, že útočník použije zprávy zachycené během předchozí (korektní) komunikace mezi A a B.

Vzhledem k dlouhodobému využívání klíčů KE_{auth} a KD_{auth} je vhodné pro ochranu před útoky využívající nenáhodný inicializační vektor pro CBC režim použít techniku ignorování prvního bloku. Před data určená k zašifrování je přidán blok náhodných dat, který je po dešifrování odstraněn.

4.2 Transportní část

Transportní část protokolu zajišťuje důvěrnost a čerstvost vyměňovaných zpráv. K zachování důvěrnosti je využita dvojice šifrovacích klíčů KE_{trans} a KD_{trans}, realizovaných a používaných obdobně jako klíče KE_{auth} a KD_{auth} v autentizační části. Pokud strana B pomocí klíče KD_{trans} korektně dešifruje příchozí zprávu, může si být jista, že pochází od strany A, neboť sama nedisponuje funkčností pro zašifrování

tímto klíčem a útočník ji tak nemůže zneužít pro vytvoření podvržené zprávy. Ze stejného důvodu je chráněn před útočníkem obsah zachycených zpráv určených pro stranu A, neboť B nedisponuje funkčností pro jejich dešifrování. Index i u hodnoty K_{R_i} označuje stav klíče K_R po jeho $i-t$ aktualizaci. $X_{K_R}(data)$ značí operace XOR hodnoty K_R nad daty. V případě, že délka dat přesahuje délku hodnoty K_R, probíhá operace XOR opakovaně po blocích délky K_R.

Vyměňované zprávy (transportní část):
5. A, B vypočtou: $K_{R_i} = H_{K_2}(K_{R_{i-1}})$.
6. $A \leftarrow B : \{idA, idB, X_{K_{R_i}}(E_{KE_{trans}}(data))\}$.
7. A, B vypočtou: $K_{R_{i+1}} = H_{K_2}(K_{R_i})$.
8. $A \rightarrow B : \{idA, idB, X_{K_{R_{i+1}}}(E_{KD_{trans}}(data))\}$.

Problémem tak zůstává zajištění čerstvosti (na straně B). Metody založené na porovnávacích kontrolách různých typů keksíku (náhodné číslo, čas, pořadové číslo) nelze použít z důvodu snadné manipulace kontrolního kódu. Vzhledem k implementaci šifrovacího algoritmu pomocí WBACR AES nelze využít ani pravidelnou změnu šifrovacího klíče. Navržená implementace využívá klíče relace K_R, aktualizovaného pomocí klíčované hashovací funkce H_{K_2} po každé odeslané i přijaté zprávě ($K_{R_{i+1}} = H_{K_2}(K_{R_i})$). Klíč K_R je použit takovým způsobem (operace X_{K_R}), aby ovlivnil každý bit odesílané i přijímané zprávy (viz 5.3). Jeho nekorektní hodnota vzhledem ke zpracovávané zprávě tak vede k jejímu poškození. Při použití vstupního a výstupního kódování pro aktualizaci K_R není na straně B jeho otevřená hodnota použita a je tak ztížena jeho lokalizace nebo modifikace. Vzhledem k použití klíče K_R jako inicializačního vektoru není nutné použít techniku ignorování prvního náhodného bloku použitou v autentizační části.

WBACR AES tabulky pro šifrovací klíče KE_{trans} a KD_{trans} jsou generovány s využitím vstupního a výstupního kódování (narozdíl od KE_{auth} a KD_{auth}). Data získaná dešifrováním zprávy pomocí klíče KD_{trans} tak nejsou ihned přístupná v otevřené podobě čitelné pro útočníka. Odstranění použitého kódování může být provedeno postupně v následujícím kódu nebo může sloužit k napojení další operace ve stylu mobilní kryptografie. Zvyšuje se tím provázanost jednotlivých částí kódu, útočník má ztíženo získání dat v otevřené podobě a extrakci nebo nahrazení WBACR AES tabulek. Volbou různého vstupního a výstupního kódování lze provádět personalizaci softwarového agenta strany B nezávisle na straně A.

Podrobnější rozbor lze nalézt v [14].

5 Stavební prvky SEAUT

Tato část textu se věnuje popisu implementace jednotlivých prvků protokolu SEAUT v kódu softwarového agenta strany B. Na straně A, u které nepředpokládáme přístup útočníka ke kódu a průběhu zpracování, je implementace prováděna běžným způsobem, například použitím standardní implementace algoritmu AES přijímajícím na vstupu data i klíč a podobně. Strana A ani B nemusí mít žádnou apriorní znalost o způsobu implementace zpracování zpráv protokolu SEAUT na opačné straně.

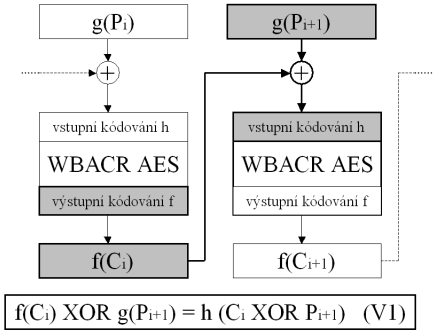
V následujícím textu bude používána zkratka IOC bez indexů pro označení vstupního a výstupního kódování tak, jak bylo zavedeno v 3 ve významu obecného principu použití funkce G_{IOC} . Pokud je použit dolní index IOC_X, označujeme tím konkrétní množinu obsahujících jedno nebo více G_{IOC} kódování, která se používá pro transformaci hodnot konkrétní proměnné (například K_R). Konkrétní proměnná se může nacházet ve stavu transformace různými množinami IOC_X, v daném okamžiku vždy však nejvýše jednou. Například pro K_R je jsou použity množiny IOC_1, IOC_2, IOC_3, IOC_4 a IOC_7. Více než jeden prvek má množina IOC_X v případě, že velikost m použitelného G_{IOC} (zde typicky 8 bitů) je menší než velikost n transformované proměnné (zde typicky 128 bitů). Proměnná je sekvenčně rozdělena na $\lceil n/m \rceil$ částí s různým kódování (zde typicky 16 částí) a každá část je nezávisle transformována. Konkrétní kódování G_{IOC} z množiny X pro $j-tou$ část proměnné je značeno IOC_X^j.

5.1 I/O kódování pro CBC režim

Vstupní a výstupní kódování (dále IOC), jak je popsáno v [2], poskytuje způsob, jak ztížit použití WBACR AES tabulek, pokud dojde k jejich extrakci. Bez dodatečných úprav však IOC nelze použít pro jiný šifrovací režim než ECB. Použití pro režim CBC by vyžadovalo odstranění resp. aplikaci IOC před každou jeho iterací, což by výrazně snížilo rychlost šifrování. Zároveň by byl kód aplikující kódování s využitím dynamické inspekce snadno lokalizovatelný a jeho použití by ztratilo význam. Pro praktickou možnost využití IOC pro režim CBC byla navržena modifikace znázorněná na obrázku 1. Ke dvojici vstupního

h a výstupního kódování f je přidáno datové kódování g. Narozdíl od původního však nejsou jednotlivá kódování nezávislá, ale jsou generována tak, aby byl splněn vztah $f(C_i)\ XOR\ g(P_{i+1}) = h(C_i\ XOR\ P_{i+1})$ (dále V_1). Po aplikaci funkce XOR na výstup předchozí iterace (kódování f) a následujícího vstupu (kódování g) získáme hodnotu $h(C_i\ XOR\ P_{i+1})$. Použití IOC během celého procesu šifrování (resp. dešifrování) je transparentní a nevyžaduje žádnou změnu oproti běžnému režimu CBC. Datové kódování g je aplikováno v libovolném místě před počátkem šifrování a může být součástí předchozí operace navržené ve stylu mobilní kryptografie. Analogicky pro výstupní kódování f. Je vhodné poznamenat, že použití IOC je plně záležitostí strany B a nemá žádný vliv na formát zpráv vyměňovaných mezi A a B. Strana A tedy nemusí mít žádnou znalost o hodnotě použitého IOC a sama ho žádným způsobem nevyužívá.

Interní struktura WBACR AES prakticky umožňuje použít IOC o maximální velikosti 8 bitů. Nelze tedy použít jediné kódování pro celý 128 bitový šifrovaný blok. Operace XOR je ale „blokovatelná“ (výsledek i-tého bitu nezávisí na j-tém bitu), lze tedy použít 16 nezávislých IOC^1 až IOC^{16} pro 1. až 16. bajt bloku.



Obrázek 1: *Vstupní a výstupní kódování použitelné pro šifrovací režim CBC. P_i značí i-tý blok vstupních dat, C_i značí i-tý blok zašifrovaných dat.*

Algoritmus generování funkcí f, g, h splňujících V_1 vychází z následujících úvah:

- Pro splnění vztahu V_1 musí být splněn i méně obecný vztah V_2 vzniklý volbou x = y:
  $f(x)\ XOR\ g(x) = h(x\ XOR\ x) = h(0)$ (V_2)
- Z podmínky bijekce funkce h a V_1 plyne:
  $\forall m,\ n,\ m',\ n' \in DOM : \{m\ XOR\ n = m'\ XOR\ n'\} \Rightarrow$
  $\Rightarrow \{\ h(m\ XOR\ n) = h(m'\ XOR\ n')\} \Rightarrow \{f(m)\ XOR\ g(n) = f(m')\ XOR\ g(n')\}$ (V_3)
- Podmínka bijekce funkce f stanovuje, že:
  $\forall m, n \in DOM : m \neq n \Rightarrow \{f(m) \neq f(n)\}$ (V_4)
- Znalost funkční hodnoty h(s) a vztah V_3 umožňuje stanovit omezující podmínku na rozsah možných hodnot, kterou musejí splňovat funkční hodnoty f(m) a g(n), pokud s = m XOR n:
  $\forall s, m, n \in DOM : (m\ XOR\ n = s) \Rightarrow f(m)\ XOR\ g(n) = h(s)$ (V_5)
- Pomocí operace XOR a operandů z množiny $P = \{x | x = 2^p, p \in \{0, \ldots, k-1\}\}$ lze vyjádřit libovolné číslo z intervalu $< 0, 2k >$. Využitím této vlastnosti, vztahu V_5, volby f(0) a h(0) a volby f(x) pro $\forall x \in P$ lze jednoznačně určit všechny funkční hodnoty funkce f.

Funkce f, g, h jsou bijektivní a pro praktické použití zadány formou tabulky. Definiční obor i obor funkčních hodnot těchto funkcí jsou přirozená čísla z intervalu $DOM = < 0, 2k - 1 >$ (definiční obor), kde k je velikost argumentu použitého kódování (v bitech).

Postup generování:

1. Zvolíme náhodně funkční hodnoty $f(0)$ a $h(0)$. Inicializujeme všechny hodnoty tabulek určující funkce f, g a h hodnotou UNASSIGNED. ($UNASSIGNED \notin DOM$).
2. Využitím V_1, $f(0)$ a $h(0)$ vypočteme $g(0)$.
3. Zvolíme náhodně, ale v souladu s V_4, funkční hodnoty $f(s)$ pro $\forall s : s = 2^p, p \in \{0, \ldots, k\ -1\}$.
4. Pro všechny zvolené funkční hodnoty $f(s)$ z bodu 3 využitím V_1, $f(s)$ a $h(0)$ vypočteme $g(s)$.
5. Pro všechny zvolené funkční hodnoty $f(s)$ z bodu 3 využitím V_1, $f(s)$ a $g(0)$ vypočteme $h(s)$.
6. Využitím vztahu V_5 a již vypočtených funkčních hodnot pro $f(r)$ a $h(s)$ vypočteme funkční hodnotu $g(r\ XOR\ s) \leftarrow f(r)\ XOR\ h(s)$. Z vypočtené hodnoty $g(rXORs)$ a vztahu V_2 vypočteme funkční

hodnoty pro $f(rXORs)$ a $h(rXORs)$. Bod 6 opakovaně iterujeme přes všechny hodnoty $t \in DOM$, dokud $\exists t : f(t) = UNASSIGNED$.

7. Generujeme WBACR AES tabulky s použitím vstupního kódování h a výstupního kódování f.
8. Před šifrováním v režimu CBC aplikujeme na vstupní bloky dat P_i funkci g.
9. Provedeme kroky šifrovacího režimu CBC standardním způsobem.

V případě šifrování dle ECB je počet všech možných různých variant vstupních a výstupních kódování v obou případech $[(2^k)!]$ pro kódování o velikosti k bitů. Navržená metoda generování f, g, h díky nutnosti splnit vztah V_1 umožňuje vytvořit 2^k možných voleb pro $f(0)$, 2^k možných voleb pro $h(0)$, a k možností volby jednoznačně neurčitelných funkčních hodnot f v souladu s V_4. Ostatní funkční hodnoty funkcí f, g a h jsou plně determinovány. Celkové množství různých variant bijektivních kódování f, g a h je rovno $[2^k x 2^k x (2^k - 1) x \ldots x (2^k - k))]$. Pro 8-bitové kódování (k = 8) tedy asi 2^{70} různých variant.

5.2 Klíčovaná hashovací funkce s ukrytím hodnoty klíče

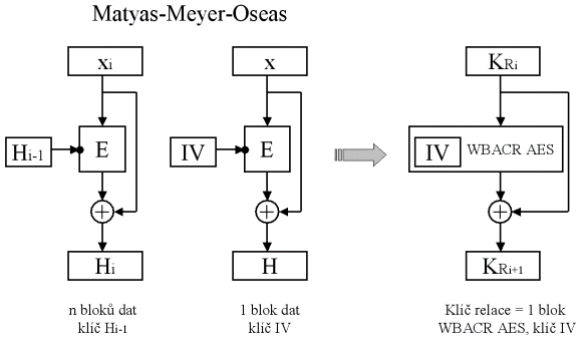
Pro potřeby robustního zajištění čerstvosti vyměňovaných zpráv jsou na aktualizační proces klíče K_R kladeny následující požadavky:

a) Útočník není schopen odvodit předchozí hodnotu(y) klíče relace z aktuální hodnoty.
b) Útočník není schopen odvodit následující hodnoty klíče relace z aktuální hodnoty.
c) Útočník není schopen smysluplně modifikovat hashovací funkci.
d) Útočník není schopen smysluplně modifikovat aktuální hodnotu klíče relace.

Splnění požadavku a) je zajištěno, pokud aktualizační proces využívá kryptograficky jednosměrné hashovací funkce. Splnění požadavku b) je zajištěno, pokud je využita klíčovaná hashovací funkce a hodnotu použitého klíče lze utajit před útočníkem. Splnění vlastnosti c) lze zajistit transformací hashovací funkce pomocí technik mobilní kryptografie nebo *obfuscation*. Požadavek d) lze splnit, pokud je hodnota K_R v průběhu celého procesu přítomna pouze v podobě s aplikovaným IOC kódováním a nedochází k jeho odstranění ani během jejího použití.

Na základě výše uvedených požadavků byla navržena metoda aktualizace hodnoty klíče relace využívající WBACR AES. Metoda je založena na využití schématu Matyas-Meyer-Oseas [12] (dále MMO) pro tvorbu hashovací funkce z blokového šifrovače, jak je znázorněno na obrázku 2. Vzhledem k fixní velikosti klíče relace dochází vždy pouze k jedinému cyklu MMO a hodnota použitého šifrovacího klíče je vždy rovna předem dané hodnotě IV. Tato vlastnost umožňuje použít blokový šifrovač realizovaný formou šifrovací části WBACR AES tabulek.

Navržená metoda zajišťuje vlastnost kryptografické jednocestnosti nejméně na úrovni MMO s jedním cyklem. Použití WBACR AES umožňuje utajit před útočníkem hodnotu klíče IV. Lze použít IOC generované dle 5.1 pro aktualizaci K_R bez nutnosti odstranění IOC.



Obrázek 2: *Klíčovaná hashovací funkce pro aktualizace klíče relace K_R s využitím WBACR AES.*

5.3 Použití a aktualizace K_R

Klíč relace K_R je vytvářen klíčovanou hashovací funkcí (viz 5.2) z očekávaných a obdržených hodnot během autentizační části protokolu SEAUT. Tvorba hashovací funkce z blokového šifrovače realizovaného s využitím WBACR AES zvyšuje bezpečnost K_R, neboť útočník nemůže bez extrakce klíče určit ve vlastní

režii výslednou hodnotu K_R. Hashovací funkce má výstupní kódování odpovídající vstupnímu kódování K_R.
Hodnota K_R je použita dvěma způsoby:

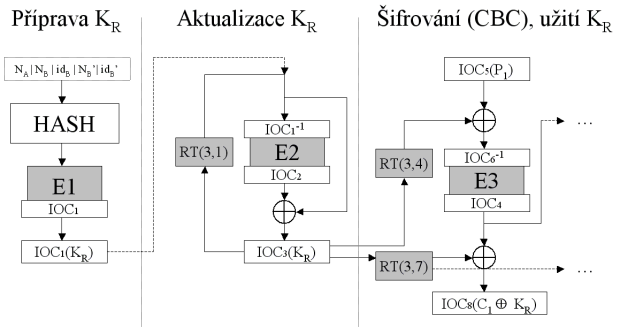
1. *Inicializační vektor pro režim CBC* – cílem je zajistit, aby chybně ustanovený klíč relace K_R vedl k vytváření a zpracování nekorektních zpráv vyměňovaných během transportní části. Pokud útočník modifikuje běh softwarového agenta tak, aby pokračoval i přes nekorektní autentizaci, bude hodnota K_R odlišná a vyměňované zprávy nebudou zpracovány korektně. Vzhledem k dlouhodobému využití klíčů KE_{trans} a KD_{trans} s neměnící se hodnotou poskytuje změna klíče K_R po každé zprávě ochranu proti útokům využívajícím nenáhodný inicializační vektor.
2. *Opakovaná aplikace pomocí operace XOR na zašifrovaná data* – cílem je zajistit, aby zpracování příchozí zprávy vytvořené pomocí jiné hodnoty K_R vedlo k chybnému zpracování celé zprávy, ne pouze prvního bloku jako v případě použití pro inicializační vektor.

Hodnota K_R je aktualizována po každé přijaté nebo odeslané zprávě. Před další aktualizační iterací je třeba změnit kódovanou hodnotu K_R z výstupního na vstupního kódování (viz 5.4). Pokud je IOC pro K_R použito, útočník nemá v paměti přístupnou jeho otevřenou hodnotu, což ztěžuje jeho nalezení a modifikaci na konkrétní hodnotu. Je zároveň ztíženo i odstranění samotné opakované aplikace pomocí funkce XOR na data, neboť bez jeho použití nedochází ke změně kódování z IOC_4 na IOC_8 (viz 3), což v důsledku vede k chybnému zpracování dat bez změny kódování dalším kódem, který očekává data v kódování IOC_8.

5.4 Provázanost IOC

Pro zajištění čerstvosti komunikace je klíčová ochrana hodnoty K_R proti podvržení. Z tohoto důvodu je přítomna pouze v chráněné podobě s aplikovaným IOC. Hodnota K_R je využívána jako argument funkce XOR a její použití i v kódované podobě lze provést při vhodném generování pro IOC základní bloky využívající WBACR AES (5.3, 4.2). Hodnota K_R se tak nikdy neobjeví v paměti softwarového agenta strany B v otevřené podobě. Provázanost korespondujících IOC je zachycena na obrázku 3. Generování IOC dle 5.1 kompatibilního s více jak jednou operací XOR by příliš omezilo celkový počet možných různých kódování a zvýšilo útočníkovu šanci na odhalení předpisu použitého kódování. Proto jsou jednotlivá kódování generována nezávisle a pro změnu kódování dané hodnoty (například klíče K_R) jsou použity předpočtené tabulky. Změnu kódování z IOC_1 na IOC_2 lze provést pomocí předpočtené tabulky pro operaci identity se vstupním kódováním IC_1 a výstupním kódováním OC_2. V případě n-bitového IOC se jedná o tabulku velikosti $2^n * n$ bitů.

Alternativou ke změně kódování z důvodu použitelnosti pro operaci XOR je použití předpočtených tabulek pro tuto operaci s odpovídajícím IOC. V případě dvou n-bitových argumentů se jedná o tabulku velikosti $2^{2n} * n$ bitů. Výhodou tohoto přístupu je možnost nahradit operaci XOR libovolnou jinou „blokovatelnou“ funkcí, která by byla vhodnější pro zajištění čerstvosti zpráv. Strana A musí také používat stejnou funkci pro aplikaci K_R.



Obrázek 3: *Provázanost vstupních a výstupních kódování základních bloků. E1, E2 a E3 označují šifrovací část WBACR AES tabulek pro různé hodnoty klíče. RT(x,y) značí tabulku pro změnu kódování z IOCx na IOCy. P_1 označuje první blok otevřeného textu. C_1 značí první blok zašifrovaného textu.*

6 Diskuse

6.1 Celková odolnost implementace

Vzájemná autentizace komunikujících stran není konečným krokem při komunikaci dvou stran. Ve většině případů dochází po úspěšném provedení autentizace k výměně dat. Protokol SEAUT této vlastnosti využívá a obě části prostřednictvím generování a použití K_R provazuje. Autentizace útočníka vůči některé straně (následkem úspěšného útoku) tak automaticky nevede k možnosti libovolně manipulovat s následnou komunikací. Samotná autentizace tak nemusí být pro útočníka postačující. Pokud nedochází k výměně dat v transportní části, protokol SEAUT poskytuje pouze ochranu hodnotám nesených autentizačních klíčů, neposkytuje však ochranu proti využití WBACR AES tabulek jako šifrovacích strojů[4].

Bezpečnost autentizační části navrženého protokolu není nižší než bezpečnost běžně implementovaného protokolu P3-ISO9798-2. Shodnou volbou hodnoty klíčů KE_{auth} a KD_{auth} získáme zprávy odpovídající protokolu P3-ISO9798-2. Pro stranu A je tedy autentizační část SEAUT stejně bezpečná jako protokol P3-ISO9798-2. Bez extrakce klíčů KE_{auth} a KD_{auth} z WBACR AES tabulek (v současné době není znám výpočetně proveditelný způsob) nezíská útočník autentizační informace strany A.

Data vyměňovaná během transportní části jsou šifrována dlouhodobě sdílenými klíči KE_{trans} a KD_{trans}. Zprávy od A pro B nelze zaměnit za zprávy od B pro A, neboť je použita jiná hodnota šifrovacího klíče. Kód protokolu v transportní části nepracuje s otevřenými hodnotami zpracovávaných dat. Útočník tak má ztížen jejich zisk nebo cílenou modifikaci. Opakovaně zaslaná zpráva od A k B je chybně dešifrována, neboť již došlo k aktualizaci hodnoty klíče K_R. Hodnota K_R není přístupná v žádném okamžiku v otevřené podobě, proto je ztížen její zisk a cílená modifikace. Pro odstranění opakované aplikace hodnoty K_R na data v kódu strany B útočníkem je nutné zároveň upravit následující funkci(e) tak, aby akceptovala vstupní data v jiném IOC kódování (IOC_4 namísto IOC_8).

Extrakce WBACR AES tabulek pro daný klíč vede ve většině případů k úspěšnému útoku (s výjimkou zisku autentizačních informací strany A). Proto je vhodné implementaci kombinovat s *obfuscation* nástroji, které extrakci tabulek ztíží. Návrh implementace protokolu je motivován snahou zvýšit rozsah modifikací kódu útočníkem pro provedení úspěšného útoku, proto je vhodná kombinace s nástroji pro kontrolu integrity kódu [5, 6].

6.2 Srovnání s asymetrickou kryptografií

Pomocí oddělení šifrovací a dešifrovací části WBACR AES tabulek získáváme výhody plynoucí z asymetrické kryptografie. V kontextu tohoto příspěvku je tento postup výhodnější než standardní algoritmy asymetrické kryptografie (RSA, DSA) z těchto důvodů:

- *Vyšší rychlost šifrování* – Asymetrické kryptografické algoritmy jsou řádově pomalejší, než algoritmy symetrické kryptografie. Hybridní šifrování, při kterém je asymetrická kryptografie použita pouze pro distribuci klíče symetrické kryptografie, nelze použít. Útočník je schopen využitím dynamické analýzy získat otevřenou hodnotu sym. klíče v době jeho dešifrování nebo použití. Naproti tomu u WBACR AES není otevřená hodnota klíče nikdy přístupná v operační paměti.
- *Není chráněna hodnota privátního klíče* – Hodnota privátního klíče asymetrické kryptografie je v kódu přítomna v otevřené podobě a je tedy dostupná útočníkovi s využitím statické a dynamické analýzy.
- *Vhodnější předpoklady pro dodatečné provedení obfuscation* – Šifrování pomocí WBACR AES využívá pouze operace náhledu do předpočetných tabulek (datových polí). Více druhů operací u asymetrické kryptografie zvyšuje sémantickou informaci o probíhající funkci, kterou útočník získá pozorováním aktuální instrukce procesoru.
- *Vstupní a výstupní kódování zpracovávaných dat* – Standardní asymetrické algoritmy vyžadují přístup k otevřené hodnotě zpracovávaných dat a nelze využít možnosti propojení okolního kódu s kódem algoritmu (obsahující hodnotou privátního/veřejného klíče). Lze tedy podvrhnout hodnotu dat určených ke zpracování, lze číst otevřenou hodnotu zpracovaných dat a provést nahrazení hodnot klíčů jinými.

[4] Bez znalosti hodnoty obsaženého klíče.

7 Závěr

Příspěvek popsal základní stavební bloky implementace autentizačního a transportního protokolu navrženého s využitím principů mobilní kryptografie, který by měl ztížit útok na autentizaci, důvěrnost a čerstvost i v případě, že se jedna strana nachází ve výpočetním prostředí kontrolovaném útočníkem. Cílem bylo maximálně využít možností poskytovaných implementací šifrovacího algoritmu AES ve stylu mobilní kryptografie pro zvýšení bezpečnosti implementace protokolu.

Kromě základní výhody ukrytí hodnoty použitého klíče je využita oddělitelnost šifrovací a dešifrovací části tabulek pro zavedení „směrovosti“ klíčů tak, aby na straně B nebyla nutná přítomnost zároveň šifrovací i dešifrovací funkčnosti pro daný klíč. Útočník tak není schopen bez extrakce klíče podvrhnout zprávy pocházející od strany A. Důležitá část práce je věnována vhodnému využití možnosti generovat WBACR AES tabulky se vstupním a výstupním kódováním. Je navržen algoritmus generování kódování tak, aby ho bylo možné efektivně používat pro šifrovací režim CBC s možností „napojení“ dalších funkčních bloků implementovaných ve stylu mobilní kryptografie.

Pro zajištění robustní čerstvosti vyměňovaných dat je navrženo generování a použití klíče relace K_R tak, aby útočník nemohl predikovat jeho hodnotu a neměl ji v žádném časovém okamžiku přístupnou v otevřené podobě v paměti. Vhodným provázáním vstupního a výstupního kódování použitého pro různé funkční bloky v rámci protokolu (šifrování, klíčovaná hashovací funkce) je dosaženo stavu, kdy nedochází k odstranění kódování dat přímo v rámci transportní části protokolu a odhalení „pozice“ v kódu tak nevede přímo ke kompromitaci zpracovávaných dat útočníkem.

Literatura

[1] Sander, T., Tschudin, Ch.: Protecting Agents From Malicious Hosts. Springer LNCS 1419, Berlin, 1998, s. 44-60

[2] Chow, S., Eisen, P., Johnson, H., van Oorschot, P.C.: White-Box Cryptography and an AES implementation. Springer LNCS 2595, Berlin, 2003, s. 250-270.

[3] Rosenblatt, B. Trippe, B. Mooney, S.: Digital Rights Management: Bussines and Technology. Indianapolis, Hungry Minds, Inc., listopad 2001, ISBN 0-7645-4889-1.

[4] Hohl, F.: Time Limited Blackbox Security: Protecting Agents From Malicious Hosts. Springer LNCS 1419, Berlin, 1998, s. 92-113.

[5] Horne, B., Matheson, L., Sheehan, C., Tarjan, R.: Dynamic Self-Checking Techniques for Improved Tamper Resistance. Springer LNCS 2320, Berlin, 2001, s. 141-159.

[6] Chang, H. Attalah, M.: Protecting Software Code by Guards. Springer LNCS 2320, Berlin, 2001, s. 160-175.

[7] NIST: Advanced Encryption Standard AES, 2000. Dokument dostupný na URL `http://www.nist.com/aes/` (listopad 2004)

[8] Červeň, P.: Cracking a jak se proti němu bránit. Praha, ComputerPress, 2002, ISBN 80-7226-382-X.

[9] Zemánek, J.: Cracking bez tajemství. Praha, ComputerPress, 2002, ISBN 80-7226-703-5.

[10] Collberg, Ch., Thomborson, C. Low, D.: A Taxonomy Of Obsfuscating Transformations. New Zeland, University Of Aucland, 1997. Dokument dostupný na URL `http://www.cs.arizona.edu/~collberg/Research/Publications/CollbergThomborsonLow97a/A4.pdf` (listopad 2004)

[11] ISO/IEC 9798-2:1999 Information technology – Security techniques – Entity authentication – Part 2: Mechanisms using symmetric encipherment algorithms. Popis protokolu je také dostupný v [12].

[12] Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press 1996/2001. Dostupné také na URL `http://www.cacr.math.uwaterloo.ca/hac/` (listopad 2004).

[13] Dusart, P.,Letourneux, G., Vivolo, O.: Differential Fault Analysis on A.E.S., Springer LNCS 2846, Berlin, 2003, s. 293-306.

[14] Švenda, P.: Digital Rights Managment. Diplomová práce. Fakulta informatiky, Masarykova universita, Brno, 2004. Dokument dostupný na URL `http://www.fi.muni.cz/~xsvenda/securefw.html`. (listopad 2004)