Hardwarová akcelerace AES-GCM pro protokol SSH

Karel Koranda

ikoranda@fit.vutbr.cz

Fakulta informačních technologií
Vysoké učení technické v Brně
Brno, Česká Republika

Abstract

This paper describes the design of hardware acceleration unit of SSH protocol used for securing network traffic. The unit is to be used as a part of FPGA design on embedded device, thus the design of the unit follows the principles of HW/SW codesign and is supposed to cooperate with modification of existing library implementing mechanisms of SSH protocol. The proposed architecture aims on acceleration of performance heavy computation, namely integrity assurance and encryption over processed data. To complete the goal, encryption algorithm Advanced Encryption Standard (AES) in Galois Counter Mode of operation was chosen, which provides data confidentiality, integrity and authentication. The implemented acceleration unit reaches maximum throughput of 2,4 Gbps at frequency of 100 MHz, though optimalizations are still possible.

Abstrakt

Tento článek popisuje návrh hardwarové akcelerační jednotky pro protokol SSH, který slouží pro zabezpečení přenosu síťových dat. Jednotka bude součástí čipu FPGA na vestavěném zařízení, takže návrh jednotky je založen na principech techniky HW/SW codesign. Jednotka je připravena pro spolupráci s modifikací existující softwarové knihovny implementující mechanismy protokolu SSH. Navržená architektura si bere za cíl urychlení výpočetně náročných operací protokolu SSH, konkrétně o zajištění integrity a šifrování zpracovávaných dat. Akcelerační jedntoka implementuje šifrovací algoritmus Advanced Encryption Standard (AES) v režimu činnosti blokových šifer Galois Counter Mode (GCM), který zajišťuje důvěrnost, integritu a autentizaci dat. Implementovaná akcelerační jednotka dosahuje na frekvenci 100 MHz propustnosti až 2,4 Gb/s, existuje však potenciál pro další optimalizace jednotky.

Klíčová slova: SSH, AES-GCM, hardwarová akcelerace.

1 Úvod

Jedním z trendů dnešní doby je neustále se zvyšující potřeba připojení uživatelů do sítě Internet, přičemž současně dochází k nárůstu objemu přenášených dat. Přenášená data jsou jsou často citlivá a mohou lákat potenciální útočníky. Organizace, jako je například *Internet Engineering Task Force – IETF*, se snaží navrhovat, standardizovat a aktualizovat prostředky určené pro zabezpečení přenášených dat.

S potřebou přenést vysoké objemy dat se také pojí požadavek na jejich vyšší přenosovu rychlost. Není proto nijak neobvyklým jevem, že právě ve zpracování dat přenášených počítačovou sítí se uplatňují principy HW/SW codesignu, techniky, jejíž cílem je pomocí akceleračních hardwarových jednotek umožnit urychlení a odlehčení výpočtů hlavního procesoru počítače předzpracováním dat. Akcelerační jednotky jsou zvláště výhodné u samostatných vestavěných zařízení, které často nedisponují výkonnými procesory. Pokud je pak vyžadován přenos zabezpečených dat z takových zařízení, například ze sondy provádějící monitorování síťového provozu, může být bez pomocných akceleračních jednotek uspokojení požadavků na rychlost zpracování a přenosu těchto dat velice obtížně dosažitelné.

S vidinou této jednoznačné motivace přichází i tento článek prezentující vytvořenou hardwarovou akcelerační jednotku pro vestavěná zařízení s čipy FPGA provádějící urychlení výpočetně náročných operací zabezpečení přenosů velkoobjemových dat protokolem *Secure Shell – SSH*. Akcelerační jednotka konkrétně implementuje zabezpečení zpráv tohoto protokolu algoritmem *Advanced Encryption Standard – AES* [14] v režimu činnosti blokových šifer *Galois Counter Mode* [3, 12]. Článek je založen na výsledcích diplomové práce [10] autora článku.

2 Existující akcelerační jednotky

V současné době existuje poměrně mnoho komerčních řešení pro akceleraci zabezpečení dat s využitím čipů FPGA. Většinou se jedná o řešení dodávaná v podobě tzv. *Intelectual Property Cores – IP Cores*, která implementují samostatné šifrovací algoritmy a kryptografické hashovací funkce. Příkladem budiž akcelerační jednotky společností IP Cores [8], HiTech Global [7], CAST [2] a mnoho dalších. Existuje také celá řada otevřených a volně dostupných IP Cores, zveřejňovaných například v rámci projektu OpenCores [15]. Co se však týká kompletního řešení pro konkrétní protokoly, například pro protokol SSH, je pole působnosti poměrně volné a otevřené. Autorovi článku je v současné době známo pouze jediné kompletní řešení protokolu SSH pro vestavěné systémy, a to komerční produkt NanoSSH [13]. V akademické sféře byl rovněž prezentován stručný článek o akceleraci protokolu SSH na čipech FPGA [5], dokumentované zrychlení však není nijak zvlášť výrazné.

3 Použitá platforma

Systém je navrhován pro využití v rámci vestavěného zařízení [11] s požadavkem na přenos dat o maximálních rychlostech 1 Gb/s. Hlavním výpočetním prvkem zařízení je čip FPGA, v rámci jehož architektury je instancován nevýkonný softcore procesor MicroBlaze umožňující běh operačního systému Linux pro zajištění softwarových prostředků pro řízení zařízení. Výhledově by kombinace FPGA a procesoru MicroBlaze na cílové platformě měla být nahrazena výkonnější variantou v podobě ARM procesoru s logikou FPGA Xilinx Zynq [19]. Pro cílovou platformu v současné době neexistuje použitelné řešení implementující protokol SSH, které by bylo schopné zajistit zabezpečený přenos dat na požadované rychlosti 1 Gb/s.

4 Zabezpečení přenosu dat

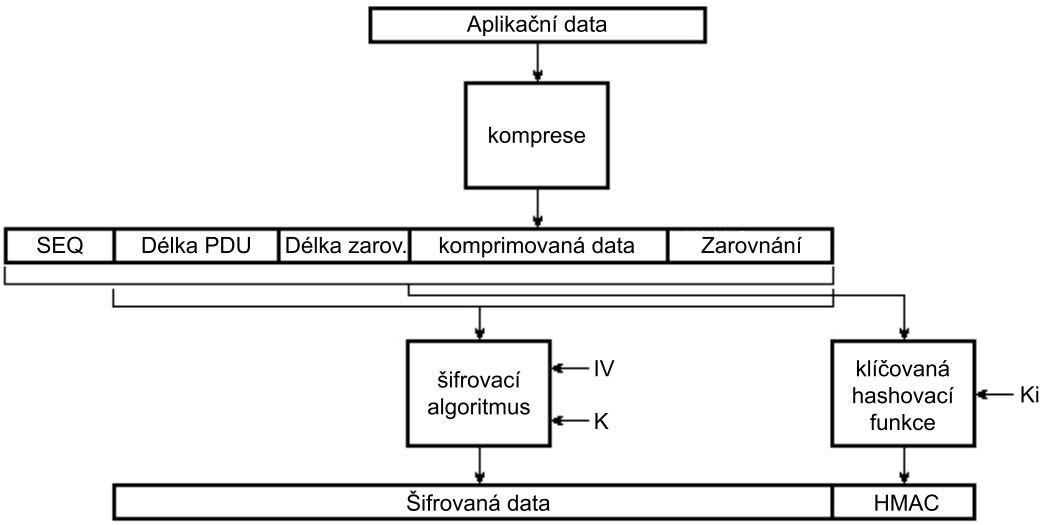
V současné době existuje poměrně mnoho síťových protokolů zaměřujících se na zabezpečení přenášených dat. Mezi nejznámější patří například na síťové vrstvě modelu ISO/OSI standard IPsec (RFC 4301), na transportní vrstvě protokol TLS (RFC 6347), nebo na aplikační vrstvě protokol SSH (RFC 4250). Podrobné srovnání zmíněných protokolů je možné si přečíst v příslušné technické zprávě [9]. V rámci tohoto článku se budeme zaměřovat výhradně na protokol SSH, protože pro něj je cílena navrhovaná akcelerační jednotka.

4.1 Protokol SSH

Protokol SSH je jedním z protokolů zajišťujících zabezpečení dat přenášených počítačovou sítí. Kromě zabezpečení přenosu dat poskytuje také prostředky pro vzdálený přístup, a tím poskytuje, kromě autentizace, důvěrnosti a integrity dat, také možnost autorizace vzdáleného uživatele. Protokol se nachází na aplikační vrstvě síťového modelu ISO/OSI. Je založen na modelu komunikace klient-server. V současné době existují dvě verze protokolu SSH - SSHv1 a SSHv2. Obě dvě verze jsou navzájem nekompatibilní, SSHv1 navíc již dnes není považováno za bezpečné.

Protokol SSHv2 byl navržen tak, aby byl schopen odolat útokům na doménové názvy, zcizení IP adresy, útokům typu Man-in-the-Middle a zcizení probíhajícího spojení. Díky průběžné výměně klíčů rovněž znemožňuje statistické útoky na šifrovací klíč. Softwarové implementace protokolu jsou poměrně rozšířené, např. nástroje OpenSSH jsou standardní součástí operačních systémů na bázi Unixu.

Protokol SSH je navržen modulárně, skládá se z několika podvrstev, které se vyskytují ve formě samostatných protokolů. Těmi jsou SSH Transport Layer Protocol (RFC 4253), SSH Authentication Protocol (RFC 4252), SSH Connection Protocol (RFC 4254). Z hlediska této práce je zajímavá pouze vrstva protokolu SSH Transport Layer Protocol, která slouží pro navázání spojení mezi klientem a serverem a zajišťuje zabezpečení aplikačních dat. Během ustanovení spojení jsou obě strany autentizovány, je provedena dohoda parametrů relace a výměna šifrovacích klíčů založená na mechanismu Diffie-Hellman. Za nejdůležitější z hlediska tvorby akcelerační jednotky je však považován definovaný způsob zabezpečení přenášených dat. Schéma tohoto zabezpečení je znázorněno na obrázku 1.



Obrázek 1: Schéma zabezpečení dat protokolem SSH

Z obrázku 1 je patrné, že kromě zabezpečení umožňuje protokol také kompresi přenášených dat (konkrétně metodou LZ77). Vidíme, že přenášená data jsou opatřena hlavičkou obsahující délku zprávy a délku zarovnání zprávy. Zároveň je každé přenášené zprávě přiřazeno sekvenční číslo určující pořadí přenášených dat v rámci relace. Z těchto údajů je vypočítán integritní součet klíčovanou hashovací funkcí. Zpráva je zašifrována šifrovacím algoritmem dohodnutým při ustanovení spojení.

Jak již bylo řečeno, protokol SSH je možné při ustanovování relace konfigurovat. Tato konfigurace spočívá ve výběru metody šifrování, metody pro výpočet integritního součtu a dohodě na používaných kryptografických klíčích pro danou relaci. Dohodnutí parametrů se účastní jak klient, tak server, a spočívá ve výběru co možná nejsilnějšího bezpečnostního mechanismu, který obě dvě strany podporují. Dle RFC 4344 a 6239 je pro použití v rámci protokolu SSH vymezeno použití kryptografických hashovacích funkcí SHA-1, MD5, od RFC 6668 také SHA-2. Použití SHA-3 zatím není definováno. Pro zajištění důvěrnosti dat je pak vymezeno použití šifrovacích algoritmů RC4, Blowfish, Twofish, 3DES, Serpent, Cast a především šifrovacího algoritmu AES. Široká nabídka použitých metod umožňuje vysokou míru interoperability různých aplikací od různých výrobců.

4.2 Režim činnosti blokových šifer Galois Counter Mode

Režim činnosti blokových šifer Galois Counter Mode, standardizovaný organizací NIST [3, 12], byl navržen tak, aby mohl zajišťovat důvěrnost a integritu dat na vysokých rychlostech v softwarových i hardwarových implementacích. Toto je možné díky novému přístupu oproti režimům CBC, ECB a CTR. Tímto přístupem je takzvané *autentizované šifrování* a *autentizované dešifrování*. Bloková šifra pracující v režimu GCM provádí zajištění důvěrnosti, integrity a autentizace důvěrných dat zároveň a umožňuje tím ze schématu zabezpečení komunikace vynechat kryptografickou hashovací funkci (např. SHA-2), jejíž realizace bývají složitě optimalizovatelné na rychlost. Pro účely náhrady těchto funkcí je navržena tzv. *univerzální hashovací funkce*, která je definovaná nad binárním Galoisovým polem $GF(2^{128})$ s generujícím polynomem

$$x^{128} + x^7 + x^2 + x + 1. \tag{1}$$

Režim GCM rovněž obsahuje mechanismus pro zajištění integrity části dat, která mají být přenášena v otevřené podobě, což je využitelné například u přenosu otevřených hlaviček některých komunikačních protokolů.

Šifrování

Šifrování v režimu GCM vychází z režimu CTR. Princip si vysvětlíme s použitím šifrovacího algoritmu AES v režimu CTR. AES je symetrický šifrovací algoritmus, který byl standardizován americkou organizací NIST [14]. Dnes je považován za jednu z nejbezpečnějších blokových šifer. Algoritmus pracuje s bloky pevné délky o velikosti 128 bitů a dle standardu teoreticky umožňuje použití libovolné délky šifrovacího

klíče. Běžně používané délky klíčů 128, 192 a 256 bitů. Algoritmus je založen na principu Feistelovy šifry, při šifrování, resp. dešifrování bloku využívá sadu aritmetických a logických operací, přičemž činnost probíhá v několika kolech. Počet kol závisí na délce použitého klíče.

Algoritmus AES v režimu CTR funguje jako generátor pseudonáhodného řetězce, kterým je s využitím logické operace exkluzivní součet šifrován vstupní otevřený text. Vztahy pro šifrování a dešifrování můžeme vyjádřit jako

$$C_i = P_i \oplus AES_K(IV + i), \qquad \text{pro } i \in 0, \ldots, n-1 \tag{2}$$

$$P_i = C_i \oplus AES_K(IV + i), \qquad \text{pro } i \in 0, \ldots, n-1, \tag{3}$$

kde K označuje šifrovací klíč, P_i blok otevřených dat, C_i blok šifrovaných dat, IV inicializační vektor, jehož hodnota je inkrementována o pořadové číslo právě šifrovaného bloku i.

Hashování

Hashování dat prováděné v režimu GCM je vystaveno na matematických základech, konkrétně na operaci násobení polynomů v Galoisově poli $GF(2^{128})$. Hashované datové bloky v binární reprezentaci vyjadřují právě polynomy, podobně jako je tomu u výpočtu CRC32 a podobných integritních součtů[1]. O operaci násobení v Galoisových polích budeme v souvislosti s režimem GCM dále hovořit jako o násobení bloků a označovat ji budeme symbolem $\bullet$. Tuto operaci je možné matematicky velice efektivně optimalizovat na rychlost zpracování, jak ji implementovat jen s využitím kombinační logiky představila společnost Intel [4, 6].

Princip výpočtu univerzální hashovací funkce, budeme ji označovat GHASH, je podobný principu, na kterém je vystaven režim blokových šifer CBC, šifrovací algoritmus je ale nahrazen operací násobení bloků $\bullet$ hashovacím klíčem H. Vstupem univerzální hashovací funkce je zpráva X, jejíž délka je násobkem 128 bitů. Jednotlivé 128 bitové segmenty zprávy budeme označovat X_i. Výpočet integritního součtu probíhá v iteracích, kde průběžné výsledky jednotlivých iterací získáme pomocí vztahů

$$Y_0 = 0 \tag{4}$$

$$Y_i = (Y_{i-1} \oplus X_i) \bullet H \tag{5}$$

Výsledkem je 128 bitový blok Y_m, kde m je počet 128 bitových bloků X_i.

Celkové schéma režimu GCM

Celkové schéma výpočtu zabezpečení dat režimem blokových šifer GCM (autentizované šifrování[2]) je vyjádřeno algoritmem 1. Vstupem algoritmu jsou dva druhy přenášených dat - data, která mají být šifrovaná, a data, která mají být přenášena pouze v otevřené podobě, ale má být zajištěna jejich integrita. Výstupem je pak šifrovaná část dat, část dat v otevřené podobě a vypočtený integritní součet označovaný jako *autentizační tag*.

Algoritmus 1 Princip činnosti režimu GCM

1: Blokový šifrovací algoritmus $E_K(x)$ s velikostí bloku 128 bitů, šifrovací klíč K
2: Inicializační vektor IV, data v otevřené podobě P, autentizovaná data v otevřené podobě A
3: $H \leftarrow E_K(0^{128})$ ▷ Výpočet hashovacího klíče
4: $J_0 = IV \parallel 0^{31} \parallel 1$ ▷ Počáteční obsah čítače
5: $C_i = P_i \oplus E_K(J_0 + i)$, pro $i \in \{0, \ldots, n-1\}$, n počet bloků P ▷ Šifrování
6: $u \leftarrow$ počet bitů zarovnání posledního neúplného autentizovaného bloku
7: $v \leftarrow$ počet bitů zarovnání posledního neúplného šifrovaného bloku
8: $S \leftarrow GHASH_H(A \parallel {}^{u} \parallel C \parallel 0^{v} \parallel$ délka A $\parallel$ délka C)
9: $T \leftarrow S \oplus E_K(J_0)$ ▷ Výpočet autentizačního tagu
10: **return** (A, C, T)

[1] Je poměrně důležité zdůraznit tento fakt, že univerzální hashovací funkce se díky tomuto principu více podobá právě výpočtu kontrolních součtů typu CRC32 a v žádném případě se nejedná o kryptogracikou hashovací funkci. Proto nesmí být využita mimo režim GCM jako samostatná hashovací funkce. Zdůvodnění, proč je funkce bezpečná v rámci režimu GCM je možné nalézt ve standardu [3].

[2] Autentizované dešifrování probíhá analogicky, na závěr výpočtu je pouze potřeba porovnat vypočtený integritní součet dešifrované zprávy s přijatým integritním součtem.

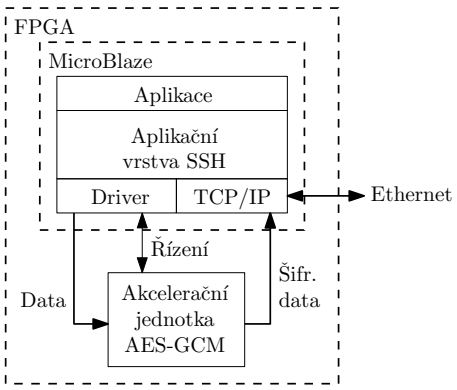
Režim GCM v rámci protokolu SSH

V rámci protokolu SSH je režim GCM definován pro použití pouze s jedním blokovým šifrovacím algoritmem a tím je algoritmus AES (RFC 5647). Díky autentizovanému šifrování je mírně upraven formát zprávy transportní vrstvy protokolu SSH, 32 bitové pole hlavičky obsahující délku zprávy je v režimu GCM přenášeno v otevřené podobě.

5 Akcelerační jednotka

V této části článku bude představen koncept umístění hardwarové akcelerační jednotky z pohledu cílové platformy jako celku, dále bude představeno komunikační rozhraní jednotky a její architektura. Na obrázku 2 je možné shlédnout jedno z možných zapojení akcelerační jednotky. Procesor MicroBlaze, instancovaný v rámci čipu FPGA, zde reprezentuje softwarovou část platformy. Software je realizován odlehčenou verzí OS Linux pro vestavěná zařízení [17]. Akcelerační jednotka je konfigurovatelná a řízená ze softwaru. Součinnost softwarové a hardwarové vrstvy probíhá následovně:

1. Aplikační vrstva nad softwarovou knihovnou SSH provede ustanovení relace mezi vestavěným zařízením a vzdálenou aplikací.
2. Po ustanovení relace se provede konfigurace hardwarové akcelerační jednotky potřebná pro zahájení zabezpečení přenášených dat, konkrétně nahrání platných kryptografických parametrů. Po konfiguraci může být činnost akcelerační jednotky započata.
3. Aplikačí vrstva během existence relace zajišťuje průběžnou výměnu klíčů relace dle specifikace protokolu SSH a dynamicky pozastavuje, rekonfiguruje a znovuspouští činnost akcelerační jednotky.



Obrázek 2: Zjednodušený návrh hardwarové akcelerace protokolu SSH

5.1 Rozhraní jednotky

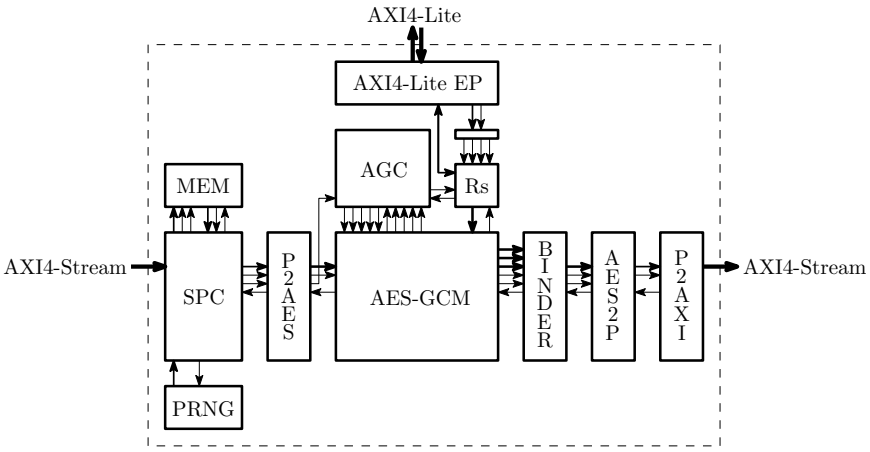
Použité rozhraní akcelerační jednotky bylo vybráno tak, aby bylo co možná nejvíce univerzální a použitelné v rámci čipů FPGA mimo vestavěné zařízení, pro které je jednotka portována. Pro tyto účely byly zvolené sběrnice AXI4-Lite [1] a AXI4-Stream [18]. Jednotka samozřejmě disponuje portem hodinového signálu (v rámci cílového zařízení se počítá s frekvencí 100 MHz) a portem synchronního resetu umožňujícím vynulování a opětovnou inicializaci stavových informací akcelerační jednotky.

Sběrnice AXI4-Lite slouží jako konfigurační rozhraní jednotky ze softwaru. Šířka datového kanálu této sběrnice je pevně stanovena na 32 bitů, sběrnice neumožňuje „burst“ transakce a není tedy možné přistupovat přes jednu adresu k registrům větší datové šířky než 32 bitů. Sběrnice ovšem obsahuje nezávislé kanály pro čtení a zápis. Pomocí AXI4-Lite se do jednotky nahrávají šifrovací klíč, inicializační vektor šifrovacího algoritmu, příkazy pro ovládání činnosti jednotky, rovněž je přes toto rozhraní možné vyčíst platný stav jednotky do softwaru.

Rozhraní AXI4-Stream slouží jako vstupní a výstupní datové rozhraní. Na vstup jednotky po tomto rozhraní přichází předzpracovaná data, která mají být zabezpečena. Rozhraní má rovněž datovou šířku 32 bitů.

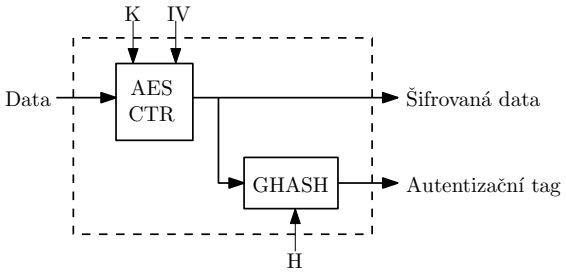
5.2 Architektura akcelerační jednotky

Architektura navržené akcelerační jednotky je přehledně znázorněna na obrázku 3. Data, jež mají být zabezpečena pro přenos protokolem SSH, do jednotky vstupují po datové sběrnici AXI4-Stream, a jsou předzpracované v komponentě *SPC*. Tato komponenta provádí uvedení dat do formátu zprávy transportní vrstvy protokolu SSH. Protože protokol SSH vyžaduje, aby v hlavičce zprávy byl údaj o délce zprávy, je potřeba průběžně ukládat data připravované zprávy do vyrovnávací paměti *MEM*, protože není dopředu známá jejich délka. Pro zarovnání dat náhodnou výplní je součástí akcelerační jednotky také jednoduchý generátor pseudonáhodných čísel *PRNG*.



Obrázek 3: Schéma top level architektury akcelerační jednotky

Zarovnaná data opatřená příslušnou hlavičkou jsou připravena k procesu zabezpečení. Komponenta *P2AES* provede převod 32 bitového vstupního rozhraní na 128 bitové rozhraní komponenty *AES-GCM*, ve které probíhá samotné zabezpečení zprávy algoritmem AES v představeném režimu GCM. Výrazně zjednodušenou strukturu jednotky *AES-GCM* je možné shlédnout na obrázku 4. Implementace je založena na IP Core implementujícím AES [16]. Proces zabezpečení je řízen konečným automatem, který je na obrázku 3 reprezentován blokem *AGC*. Šifrovací klíč a inicializační vektor je uložen v reigstrech *Rs*, kam jsou náhrány přes konfigurační sběrnici AXI4-Lite. Pro snažší komunikaci se sběrnicí je dostupná endpoint komponenta *AXI4-Lite EP* s příslušným adresovým dekodérem.



Obrázek 4: Zjednodušené schéma jednotky AES-GCM

Výstupem komponenty AES-GCM jsou šifrovaná a autentizovaná data a autentizační tag. U těchto položek je zapotřebí zajistit jejich správné pořadí v přenášené zprávě (autentizační tag musí být umístěn na konci zprávy). K tomuto slouží komponenta *BINDER*. Pro převod na 32 bitovou výstupní sběrnici AXI4-Stream jsou součástí akcelerační jednotky komponenty *AES2P* a *P2AXI*.

6 Dosažené výsledky

V této části článku bude čtenář seznámen s výsledky dosaženými při syntéze akcelerační jednotky, a to z pohledu maximální dosažené frekvence a spotřebovaných a dostupných zdrojů čipu FPGA. Dále je diskutována maximální datová propustnost jednotky dosažitelná v ideálních podmínkách simulačního prostředí. Podrobnější dosažené výsledky lze najít v textu diplomové práce [10].

6.1 Výsledky syntézy

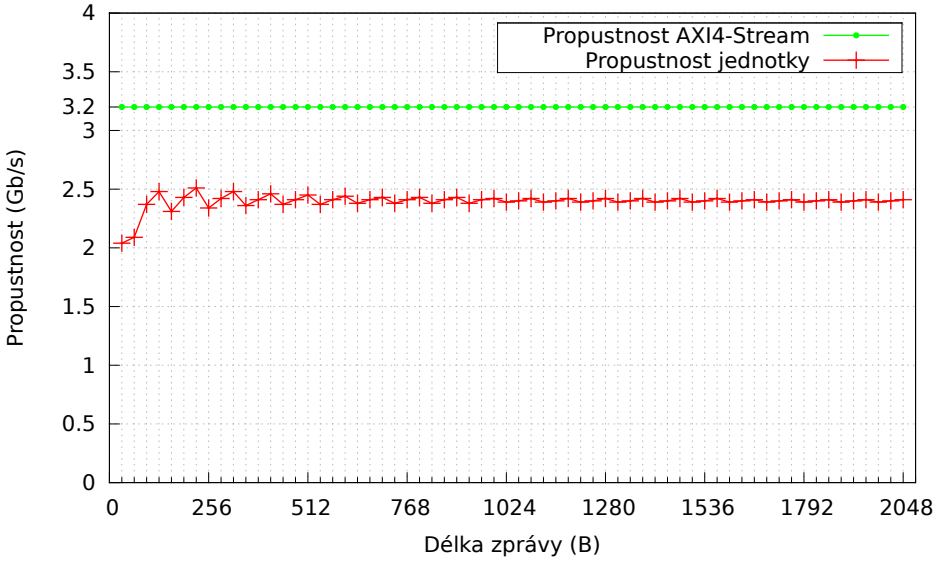
V tabulce 1 je možné shlédnout přehled vybraných spotřebovaných zdrojů[3] na cílovém čipu a srovnat je s dostupnými zdroji[4]. Z tabulky je zřetelné, že na čipu zůstává stále dostatečné množství zdrojů pro další akcelerační jednotky. Je patrné, že značná část zdrojů je spotřebována na použitý AES IP Core. Jednotku se podařilo syntetizovat na maximální frekvenci 121 MHz. Úzkým hrdlem jednotky, které brání dosažení vyšších frekvencí, jsou kritické cesty v kombinační logice výpočtu univerzální hashovací funkce.

Zdroj	Počet využitých	AES IP Core	Počet dostupných
Slice Registers	11 710 (6 %)	8 002 (4 %)	51 %
Slice LUTs	18 884 (20 %)	10 782 (11 %)	81 %
BlockRAMs	3 (1 %)	2 (1 %)	60 %

Tabulka 1: Využité zdroje na cílovém čipu FPGA

6.2 Propustnost jednotky

V rámci simulace implementované akcelerační jednotky bylo provedeno také měření její propustnosti v závislosti na různých délkách zpráv. Měření bylo provedeno od zpráv délky 32 do 2048 bajtů, délka zprávy byla vždy násobkem 32 bajtů. Výsledek je vynesen do grafu na obrázku 5. Pro porovnání je ve stejném grafu uvedena také maximální teoretická propustnost sběrnice AXI4-Stream při frekvenci designu 100 MHz, která je vstupním a výstupním datovým rozhraním akcelerační jednotky.



Obrázek 5: Graf závislosti propustnosti jednotky na délce zprávy

Z obrázku je patrné, že jednotka dosahuje maximální propustnosti pohybující se kolem 2,4 Gb/s. Fakt, že jednotka nedostahuje maximální propustnosti sběrnice, je zapříčiněn drobným zpožděním v rámci jednotky SPC tím, že provádí připojení hlavičky k datům a příslušné zarovnání na bloky, které jsou násobkem 16 bajtů. Další zpoždění je způsobeno připojením 128 bitového integritního součtu vypočteného komponentou AES-GCM k vytvořené zprávě. Je nutné zmínit, že jádro jednotky reprezentující algoritmus AES-GCM je teoreticky schopné pracovat na maximální propustnosti až 12,8 Gb/s při frekvenci 100 MHz, protože pracuje se 128 bitovými bloky – úzké hrdlo z tohoto pohledu představuje použitá 32 bitová sběrnice. Toto dává prostor pro případné optimalizace pro dosažení vyšší propustnosti jednotky.

7 Závěr

Požadavek na přenosy vyšších objemů dat počítačovou sítí a důraz na urychlování zpracování přenášených dat se promítá i do oblasti bezpečnosti, především díky vysoké výpočetní náročnosti kryptografických algoritmů, kterými je zabezpečení dat prováděno. Zabezpečení dat přenášených z vestavěných zařízení,

[3] Uvedené výsledky jsou výstupem při syntéze nástrojem Xilinx ISE Design Suide verze 14.1.

[4] Dostupné zdroje vychází z největšího známého designu pro čip FPGA v rámci cílového vestavěného zařízení.

které nedisponují výkonnými procesory, může být poměrně náročné, pokud není zajištěna příslušná hardwarová akcelerace. Tento článek prezentuje koncept akcelerační jednotky pro čipy FPGA umožňující urychlení zabezpečení dat protokolem SSH, konkrétně šifrovacím algoritmem AES v režimu činnosti GCM. Výsledná jednotka je schopna pracovat na frekvenci 100 MHz a dosahovat maximální propustnosti až 2,4 Gb/s, přičemž existuje velký prostor pro její další optimalizaci s potenciálem zvýšit rychlost zpracování až na 12,8 Gb/s při zmíněné frekvenci.

Poděkování

Tento článek byl vypracován za částečné podpory Evropského fondu regionálního rozvoje (EFRR) v rámci projektu Centra Excelence IT4Innovations (CZ.1.05/1.1.00/02.0070) a projektu Pokročilé bezpečné, spolehlivé a adaptivní IT (FIT-S-11-1).

Literatura

[1] ARM Ltd.: AMBA AXI Protocol Specification, 2010.

[2] CAST Inc.: Semiconductor IP Cores for ASICs and FPGAs, 2013. Dostupné na: `<http://www.cast-inc.com/ip-cores/>`.

[3] Dworkin, M.: Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC, National Institute of Standards and Technology, USA, 2007. NIST Special Publication 800-38D.

[4] Gopal, V., Ozturk, E., Feghali, W. et al.: Optimized Galois-Counter-Mode Implementation on Intel Architecture Processors, 2010.

[5] Gonzalez, I., Gomez-Arribas, F., Lopez-Buedo, S.: Hardware Accelerated SSH on Self-Reconfigurable Systems, in *Proceedings 2005 IEEE International Conference on Field Programmable Technology*, Singapore, 2005.

[6] Gueron, S. a Kounavis, M.: Intel Carry-Less Multiplication Instruction and its Usage for Computing the GCM Mode, 2010.

[7] HiTech Global, LLC: Verilog/VHDL IP Cores for ASIC/SOC and FPGA, 2012. Dostupné na: `<http://www.hitechglobal.com/ipcores/>`.

[8] IP Cores, Inc.: Security and DSP IP Cores for ASIC and FPGA Applications, 2008. Dostupné na: `<http://www.ipcores.com>`.

[9] Kajan, M., Koranda, K. a Polčák, L.: Spolehlivá a zabezpečená komunikace v rámci systému pro zákonné odposlechy, Brno, ČR, 2012.

[10] Koranda, K.: Akcelerace šifrování přenosu síťových dat, Brno, ČR, 2013. Diplomová práce.

[11] Kořenek, J., Korček, P., Košař, V. et al.: A New Embedded Platform for Rapid Development of Network Applications, in *ACM/IEEE Symposium on Architectures for Networking and Communications Systems 2012*, Austin, USA, 2012.

[12] McGrew, D. a Viega, J.: The Galois/Counter Mode of Operation (GCM), 2005.

[13] Mocana Corporation: NanoSSH, 2013. Dostupné na: `<http://www.mocana.com/for-device-manufacturers/nanossh/>`.

[14] National Institute of Standards and Technology: Announcing the Advanced Encryption Standard (AES), USA, 2001. Federal Information Processing Standards Publication 197.

[15] OpenCores: Homepage, 2013. Dostupné na: `<http://www.opencores.org>`.

[16] OpenCores: Pipelined AES, 2010. Dostupné na: `<http://www.opencores.org/project,aes_pipe>`.

[17] The Buildroot developers: Buildroot: Making Embedded Linux Easy, 1999-2013. Dostupné na: `<http://buildroot.uclibc.org>`.

[18] Xilinx, Inc.: AXI Reference Guide, 2012.

[19] Xilinx, Inc.: Zynq-7000 All Programmable SoC: Technical Reference Manual, 2013.