



AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
WYDZIAŁ INFORMATYKI, ELEKTRONIKI I TELEKOMUNIKACJI

KATEDRA TELEKOMUNIKACJI

Praca dyplomowa inżynierska

*Opracowanie biblioteki programistycznej do bezpiecznego
uwierzytelniania urządzeń AVR.*

Development of libraries for authentication of AVR devices.

Autor:	<i>Kacper Żuk</i>
Kierunek studiów:	<i>Teleinformatyka</i>
Opiekun pracy:	<i>dr inż. Jarosław Bułat</i>

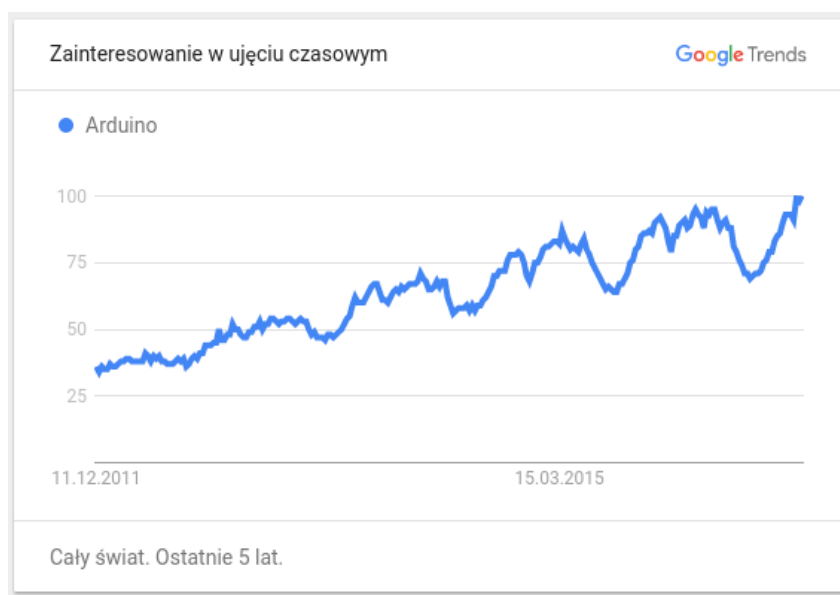
Kraków, 2016

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystycznego wykonania albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.): „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej «sądem koleżeńskim».”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Spis treści

1. Wstęp	5
1.1. Cele pracy	5
1.2. Zawartość pracy	6
2. Charakterystyka platformy sprzętowej	7
3. Metody uwierzytelniania	9
3.1. Kryptografia asymetryczna	9
3.2. Kryptografia symetryczna	10
4. Implementacja	11
4.1. Generowanie współdzielonego klucza	11
4.2. Szyfrowanie wiadomości	11
4.3. Uwierzytelnienie wiadomości	11
4.4. Protokół komunikacji	11
5. Walidacja	13
6. Podsumowanie	15

1. Wstęp



Rys. 1.1. Relatywna liczba wyszukiwań frazy „Arduino” w ostatnich pięciu latach.

Źródło: Google Trends

AVR to rodzina mikroprocesorów opracowana i rozwijana przez firmę Atmel. Oparta o nią jest m. in. platforma Arduino, która – jak przedstawiono na Rys. 1.1 – z roku na rok zyskuje popularność. Platforma Arduino zaprojektowana została z myślą o osobach, które niekoniecznie posiadają formalne wykształcenie inżynierskie [1]. Jest ona też często używana do prototypowania urządzeń, wpisujących się w koncepcję *Internetu Rzeczy* (ang. *Internet of Things*, *IoT*).

Urządzenia wbudowane podłączone do Internetu są szczególnie narażone na ataki. W 2016 roku podatne urządzenia wbudowane zostały wykorzystane do przeprowadzenia masowych ataków typu DDoS [2].

1.1. Cele pracy

Istotne jest więc dostarczenie narzędzi, które pozwalają nie tylko na szybkie prototypowanie, ale które pozwolą także zachować odpowiedni poziom bezpieczeństwa. Należy pamiętać przede wszystkim o tym, że urządzenia *IoT* są tworzone także przez ludzi bez formalnego wykształcenia inżynierskiego.

W niniejszej pracy przedstawiono protokół bezpiecznej komunikacji oraz bibliotekę programistyczną na urządzenia AVR zaprojektowane z myślą o prostocie obsługi. Wybrane zostały zestawy algorytmów, które zapewniają niezbędny poziom bezpieczeństwa. Ich złożoność została ukryta za prostym interfejsem programistycznym (*ang. API*), który nie pozwala na wprowadzenie błędów zmniejszających bezpieczeństwo. Zaproponowane rozwiązanie zapewnia poufność, autentyczność oraz integralność przesyłanych danych.

1.2. Zawartość pracy

W rozdziale 2 scharakteryzowana jest platforma sprzętowa AVR, ze szczególnym uwzględnieniem jej ograniczeń. Następnie w rozdziale 3 przedstawione zostały różne metody uwierzytelniania i uzasadniony został wybór konkretnych rozwiązań. Implementacja została szczegółowo opisana w rozdziale 4. Całość rozwiązania została zwalidowana poprzez porównanie z implementacją na inną platformę, co opisano w rozdziale 5. W rozdziale 6 podsumowano całe rozwiązanie oraz przedstawiono jego ograniczenia i słabe strony.

Całość kodu źródłowego dostępna jest w serwisie GitHub¹.

¹<https://github.com/kacperzuk/seconn>

2. Charakterystyka platformy sprzętowej

Mikropocesyory Atmel AVR są w większości 8-bitowe i na takich skupia się praca. Rodzina AVR jest szeroka, od ATtiny4 z 32 bajtami SRAM (*ang. Static Random Access Memory*) [3] do ATxmega384C3 z 32 kilobajtami SRAM [4]. W pracy wykorzystywany był model ATmega32u4 z 2,5 kilobajta SRAM [5].

SRAM jest głównym ograniczeniem, ponieważ 32 bajty nie są wystarczające do przeprowadzania operacji, przy których sam klucz zajmuje 16 lub 32 bajty. Należy też pamiętać, że obsługa bezpiecznego połączenia nie może zajmować całości pamięci. Część należy przeznaczyć na obsługę peryferiów oraz właściwą logikę programu.

Istotnym elementem jest też wielkość domyślnych buforów. *Arduino* w modułach *Serial* oraz *SoftwareSerial* domyślnie używa 16- lub 64-bajtowego (w zależności od ilości dostępnej pamięci) buforu na przychodzące dane¹. Przy wiadomościach dłuższych niż 64 bajty oznacza to, że zbyt długie przetwarzanie jednej wiadomości spowoduje błędne odebranie następnej, jeżeli zostanie ona za szybko wysłana.

FIXME tutaj jeszcze cos o taktowaniu procesora.

¹<https://github.com/arduino/Arduino/blob/master/hardware/arduino/avr/cores/arduino/HardwareSerial.h>

3. Metody uwierzytelniania

W zależności od potrzeb i ograniczeń stosuje się różne metody uwierzytelniania podmiotów w komunikacji. Wyróżnić należy uwierzytelnianie przy pomocy kryptografii asymetrycznej, w której używana jest para matematycznie związanych ze sobą kluczy, oraz uwierzytelnianie przy pomocy kryptografii symetrycznej, w której używany jest jeden, współdzielony, tajny klucz.

Klucze w przypadku kryptografii asymetrycznej muszą posiadać konkretne właściwości. W przypadku RSA bezpieczeństwo polega na trudności w faktoryzowaniu dużych liczb, co wymaga stosowania kluczy co najmniej 2048 bitowych [6]. Klucze w przypadku kryptografii symetrycznej nie muszą mieć konkretnych właściwości poza ich nieprzewidywalnością.

Ważną różnicą jest też wydajność. Kryptografia asymetryczna jest dużo bardziej złożona obliczeniowo od symetrycznej [7]. Jest to szczególnie istotne na ograniczonych sprzętowo systemach wbudowanych. Przewagą kryptografii asymetrycznej jest jednak brak konieczności ustalenia wspólnego klucza przed rozpoczęciem komunikacji, jak ma to miejsce w przypadku kryptografii symetrycznej.

Zalecanym rozwiązaniem jest najpierw ustalenie wspólnego, tajnego klucza przy użyciu kryptografii asymetrycznej, a następnie użycie tego klucza do kryptografii symetrycznej [7].

3.1. Kryptografia asymetryczna

Przy wyborze algorytmu dla potrzeb pracy istotne były:

- jakość implementacji dostępnych na mikroprocesory AVR,
- złożoność obliczeniowa (niższa jest lepsza),
- długość klucza wymagana do zapewnienia niezbędnego poziomu bezpieczeństwa.

Biblioteka *AVR-Crypto-Lib* dostarcza implementację algorytmów RSA oraz DSA¹. Biblioteka *Em-sign* dostarcza implementację RSA, lecz tylko z 64 bitowym kluczem², co nie jest wystarczające dla zapewnienia bezpieczeństwa. Komercyjna biblioteka *LightCrypt-AVR8-ECC* oraz biblioteka *micro-ecc* dostarczają implementację kryptografii opartej o krzywe eliptyczne³. Brak jest na rynku implementacji

¹<https://trac.cryptolib.org/avr-crypto-lib/browser>

²<http://www.emsign.nl/>

³http://industrial.crypto.cmmsigma.eu/lightcrypt_avr8/lc_avr8_ecc.pl.html

innych algorytmów klucza publicznego. Dostępność implementacji ogranicza wybór algorytmu do RSA, DSA oraz krzywych eliptycznych.

Następnym kryterium jest złożoność obliczeniowa. W analizie przeprowadzonej przez pracowników *Sun Microsystems Laboratories* wykazano, że na mikroprocesorach AVR algorytmy oparte o krzywe eliptyczne są o rząd wielkości szybsze od algorytmu RSA [8].

FIXME DSA jest podobne do RSA, więc też nie

FIXME dlaczego taki a nie inny secp <https://www.keylength.com/en/4/>

FIXME ECDH jest fajne

FIXME ale nie daje forward secrecy

3.2. Kryptografia symetryczna

ECBC-MAC OMAC CCM HMAC

4. Implementacja

4.1. Generowanie współdzielonego klucza

4.2. Szyfrowanie wiadomości

4.3. Uwierzytelnienie wiadomości

4.4. Protokół komunikacji

5. Walidacja

Informacja o bibliotece dla Javy i przykładowych implementacjach.

6. Podsumowanie

?

Spis rysunków

1.1	Relatywna liczba wyszukiwań frazy „Arduino” w ostatnich pięciu latach. Źródło: Google Trends	5
-----	--	---

Bibliografia

- [1] Massimo Banzi and Michael Shiloh. *Getting Started with Arduino: The Open Source Electronics Prototyping Platform*. Sebastopol: Maker Media, Inc., 2014.
- [2] Martin McKeay i in. *Q3 2016 State of the Internet Security Report*. Spraw. tech. Akamai Technologies, Inc., 2016.
- [3] *ATtiny4 / ATtiny5 / ATtiny9 / ATtiny10 - Datasheet Summary*. Atmel. 2016. URL: http://www.atmel.com/Images/Atmel-8127-AVR-8-bit-Microcontroller-ATtiny4-ATtiny5-ATtiny9-ATtiny10_Datasheet-Summary.pdf (dostęp dnia 2016-12-06).
- [4] *ATxmega384C3 - Datasheet*. Atmel. 2016. URL: http://www.atmel.com/Images/Atmel-8361-8-and-16-bit-AVR-XMEGA-Microcontrollers-ATxmega384C3_Datasheet.pdf (dostęp dnia 2016-12-06).
- [5] *ATmega16U4/ATmega32U4 - Datasheet*. Atmel. 2016. URL: http://www.atmel.com/Images/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf (dostęp dnia 2016-12-06).
- [6] *Recommendation for Key Management Part 1: General*. National Institute of Standards and Technology, U.S. Department of Commerce. 2016. URL: <http://dx.doi.org/10.6028/NIST.SP.800-57pt1r4> (dostęp dnia 2016-12-07).
- [7] Abdullah Al Hasib i Abul Ahsan Md Mahmudul Haque. „A comparative study of the performance and security issues of AES and RSA cryptography”. W: *Convergence and Hybrid Information Technology, 2008. ICCIT'08. Third International Conference on*. T. 2. IEEE. 2008, s. 505–510.
- [8] Nils Gura i in. „Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs”. W: *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*. Wyed. Marc Joye i Jean-Jacques Quisquater. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, s. 119–132. ISBN: 978-3-540-28632-5. DOI: 10.1007/978-3-540-28632-5_9.