Co-funded by the Erasmus+ Programme of the European Union

Moduł treningowy InCyT

Bezpieczeństwo informacji dla pracownika		
Część 2, Tydzień 3 **Moduł: Wprowadzenie do kryptografii**		
Zajęcia edukacyjne	❑ **Webinar** ❑ **Exercises** ❑ **Workshop** ❑ **Presentation** ❑ **Other**	
Odpowiedzialni za zajęcia	Claudio Fornaro	
Cele aktywności edukacyjnej	1. Kryptografia 2. Kryptoanaliza 3. Szyfr z kluczem jednorazowym 4. Algorytmy symetryczne 5. Algorytmy asymetryczne 6. Funkcje haszujące 7. Podpisy i infrastruktura klucza publicznego 8. Żetony kryptograficzne 9. Mechanizm wyzwanie-odpowiedź	
Umiejętności do zdobycia	• Podstawowa wiedza na temat algorytmów i infrastruktur kryptograficznych • Umiejętność techniczna 2 - TS2	
Czas trwania okresu edukacyjnego	2 tygodnie	
Wymagania wstępne	***Co jest potrzebne?***	
	• Nie jest wymagana żadna wcześniejsza specjalistyczna wiedza	

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Krótka informacja o module

Opis
Kryptografia odnosi się do metodologii ukrywania informacji przed ujawnieniem, zarówno podczas komunikacji, jak i do celów archiwalnych. Jego cele są osiągane dzięki projektowaniu i wdrażaniu protokołów, które uniemożliwiają nieuprawnionej osobie trzeciej odzyskanie informacji. W bezpiecznej komunikacji osoba trzecia nie ma dostępu do przekazywanych informacji, w przypadku archiwizacji osoba trzecia nie może prześledzić treści tego, co zostało zarchiwizowane. W kryptografii strona trzecia jest zawsze złośliwą jednostką, której celem jest odzyskanie informacji, do których nie może mieć dostępu, czy to ze względu na prywatność, tajemnicę handlową czy bezpieczeństwo narodowe. Poufność i integralność danych, uwierzytelnianie zaangażowanych stron oraz niezaprzeczalność tego, co źródło danych przesłało lub przechowało, to podstawowe zasady nowoczesnej kryptografii. Po części wstępnej, w której opisane są podstawowe pojęcia, przedstawiamy rozwój podstawowych technik kryptograficznych, zaczynając od algorytmu „One Time Pad" (Szyfr z kluczem jednorazowym), a następnie scharakteryzowano i porównano dwie rodziny algorytmów kryptograficznych: Algorytmów Symetrycznych i Asymetrycznych (lub Klucza Publicznego). Omówiono również rolę funkcji skrótu, aby wprowadzić podpisy cyfrowe. Szczegółowo przedstawiono opis kluczowej koncepcji Infrastruktury Klucza Publicznego wraz ze szczegółami dotyczącymi standardowych mechanizmów i formatów. Podsumowując, podano pewne informacje na temat tokenów kryptograficznych i mechanizmu Challenge-Response.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Treść materiału edukacyjnego

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Wprowadzenie

Kryptografia to mechanizm, za pomocą którego treść wiadomości jest ukrywana przed innymi podmiotami. Innymi słowy szyfrowanie zajmuje się szyfrowaniem wiadomości, czyli przyjmowaniem zwykłego tekstu, używaniem algorytmu i ogólnie klucza, co sprawia, że treść wiadomości jest niezrozumiała.

Kryptoanaliza bada, jak odszyfrować wiadomość bez znajomości klucza. W ataku kryptoanaliza polega na próbie prześledzenia treści wiadomości bez posiadania klucza. Kryptografowie i kryptoanalitycy to w rzeczywistości ci sami ludzie: którzy piszą algorytmy kryptograficzne, analizują je również, aby znaleźć ich słabości lub zweryfikować ich solidność. Jak zawsze, gdy mamy do czynienia z bezpieczeństwem, pomysł radzenia sobie z problemami ochrony bez wiedzy o tym, jak działają techniki ataków, nie jest zbyt skuteczny.

Informacje w wiadomościach są na ogół nadmiarowe, w sensie formalnym wiadomości na ogół używają znacznie większej liczby bitów niż ta, która jest naprawdę niezbędna do przenoszenia informacji. Redundancja informacji ułatwia kryptoanalizę: redundancja w informacjach oznacza, że wiadomości mogą wykazywać strukturę, która ułatwia śledzenie treści wiadomości, ponieważ nie wszystkie wiadomości są równie prawdopodobne. Jeśli pomyślimy o tekście reprezentowanym normalnymi znakami ASCII w bajcie, wiemy, że kombinacje bitów, które mogą reprezentować znaczące wiadomości, są stosunkowo nieliczne w porównaniu ze wszystkimi możliwymi kombinacjami.

Prawdziwe wiadomości, które można przedstawić za pomocą określonej liczby bajtów, są bardzo nieliczne. Umożliwia to identyfikację prawidłowych wiadomości poprzez odrzucenie tych, które nie zawierają prawdziwych informacji. Wszystko to znacznie pomaga w aktywności kryptoanalityka, więc celem algorytmu szyfrowania jest również ukrycie tej redundancji w celu lepszego ukrycia informacji.

Przykład: Kodowanie sekwencji czterech kolorów: BLUE, GREEN, RED oraz YELLOW.

Mogą być zakodowane:

- Za pomocą 2 bitów (00, 01, 10, 11)
- Jako 4 różne wielkie litery
- Szyfrowanie nazw kolorów

W pierwszym przypadku używana jest minimalna liczba jednostek (bitów), zastępując każdą nazwę koloru

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

odpowiednimi 2 bitami, co ukryje oryginalną sekwencję, ale nie kolejność, ponieważ istnieje zgodność między KOLOR a 2 bitami. W drugim przypadku, ponieważ każda litera używa 8 bitów w kodzie ASCII, 6 bitów jest redundantnych dla każdego koloru → 75% redundancji i osiągamy ten sam stopień ukrycia, co 2 bity. Nadal nie możemy kojarzyć kolorów i wartości.

Rozważmy następujący bardzo prosty szyfr, tak zwany szyfr Cezara, który nie robi nic innego, jak tylko przewija litery alfabetu o określoną liczbę pozycji. Na przykład, jeśli przesuniemy je o 3, wszystkie "A" staną się "D", wszystkie "B" staną się "E" i tak dalej. Jest to bardzo prosty szyfr, faktycznie przypisywany Juliuszowi Cezarowi, ale jeśli nazwy kolorów są szyfrowane tą metodą, powtarzający się wzór ich zakodowanych liter jest łatwy do zidentyfikowania (kod ASCII dla R, po którym następuje kod ASCII E, a następnie kod ASCII D są łatwe do rozpoznania jako jednostka). Tak więc sekwencję można zidentyfikować, a jeśli słowa są również znane w tym przypadku, wystarczy policzyć liczbę liter, aby zidentyfikować oryginalne słowa.

Ten prosty przykład pozwala nam zrozumieć, że kryptoanaliza nie jest analizą niezależną od treści wiadomości: całkowicie losowa wiadomość byłaby znacznie trudniejsza do zidentyfikowania i przeanalizowania niż wiadomość, która ma rozpoznawalną strukturę.

Niektóre rodzaje ataków kryptograficznych i algorytmów polegają na znajomości części tekstu. W niektórych przypadkach znajomość części tekstu umożliwia prześledzenie klucza, a następnie uzyskanie dostępu do reszty tekstu. Jest to bardzo istotne, jeśli chodzi o komunikację sieciową, ponieważ w komunikacji sieciowej protokoły na różnych poziomach zwykle mają pewną ilość informacji, które w każdym przypadku są rozpoznawalne, np. w pakiecie IP wiemy, że nagłówek zawiera adres nadawcy i adres odbiorcy.

Algorytm szyfrowania ukrywa również zbędne informacje, więc zastosowanie algorytmu kompresji do zaszyfrowanej wiadomości nie przynosi większych korzyści, ponieważ zmniejszyłoby współczynnik kompresji. Jeśli chcemy użyć kompresji i szyfrowania, musimy najpierw użyć kompresji, a następnie szyfrowania.

Algorytm “One Time Pad” – Szyfr z kluczem jednorazowym

Algorytm „One Time Pad (OTP)” jest algorytmem, powiedzmy „doskonałym”. Jest to bardzo prosty algorytm szyfrowania, bardzo prosty w implementacji i ma cechę, która czyni go szczególnie interesującym: udowodniono, że bez znajomości klucza nie można wrócić do oryginalnego tekstu. Teraz przyjrzyjmy się jemu, a następnie przedyskutujmy, dlaczego, skoro mamy tak dobry algorytm, musimy znaleźć inne.

EXCLUSIVE OR ⊕ (XOR) jest funkcją logicznągdzie wynik jest True, gdy dwa operandy są różne, działa na wartościach logicznych, ale tłumaczone na cyfry binarne, zwykle True=1 oraz False=0, jego *tablica prawdy* jest pokazana po lewej:

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

$0 \oplus 0 = 0$

$0 \oplus 1 = 1$

$1 \oplus 0 = 1$

$1 \oplus 1 = 0$

Z tej tabeli jasno wynika, że jeśli X ⊕ Y → Z, wtedy Z ⊕ Y → X oraz Z ⊕ X → Y. EXCLUSIVE OR jest bardzo prostą funkcją, a także łatwą do wdrożenia jako obwód elektroniczny. Mechanizm OTP służy do przesyłania wiadomości od dwóch stron. W literaturze naukowej powszechną praktyką jest identyfikowanie aktorów z imionami osobistymi, zwykle nazywa się ich Alice (A) i Bob (B). Opiszemy więc, w jaki sposób Alicja może bezpiecznie przesłać wiadomość o *n*-bajtach (JawnyTekst) do Boba.

Algorytm One Time Pad wymaga następujących faz:

- Generowana jest losowa sekwencja dokładnie *n* bajtów, jest to *One Time Pad* (OTP) lub po prostu *Klucz*;
- OTP jest w jakiś sposób bezpiecznie współdzielony między dwiema stronami (podany z góry, wysłany w już zabezpieczonym kanale itp.).

Gdy obie strony posiadają klucz, mogą go używać do przesyłania wiadomości. W przypadku, gdyby Alicja chciała wysłać wiadomość do Boba:

- Alicja oblicza *bit po bicie* EXCLUSIVE OR jawnego tekstu (*JawnyTekst)* z *Kluczem*, produkując *ZaszyfrowanyTekst*:

Szyfrowanie: JawnyTekst ⊕ Klucz → *ZaszyfrowanyTekst*

- Wynikowy *ZaszyfrowanyTekst* jest transmitowany przez Alicję do Boba poprzez prawdopodobnie niebezpieczny kanał.
- Bob oblicza *bit po bicie* EXCLUSIVE OR z *ZaszyfrowanyTekst* oraz z *Klucza*, odkrywając oryginalny *JawnyTekst*:

Dekrypcja: *ZaszyfrowanyTekst*⊕ Klucz → *JawnyTekst*

Przykład

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Załóżmy, że zarówno Alicja, jak i Bob uzgodnili już losowy OTP(*Klucz*).

Alicja chce wysłać “HELLO” do Boba (w kodzie ASCII, spacje są tutaj dla lepszej czytelności):

	H	**E**	**L**	**L**	**O**
JawnyTekst:	01001000	01000101	01001100	01001100	01001111
			⊕		
Klucz:	10101001	01110010	10110010	10000110	11000110
			=		
ZaszyfrowanyTekst:	11100001	00110111	11111110	11001010	10001001

Bob otrzymuje *ZaszyfrowanyTekst* i stosuje *Klucz*:

ZaszyfrowanyTekst:	11100001	00110111	11111110	11001010	10001001
			⊕		
Klucz:	10101001	01110010	10110010	10000110	11000110
			=		
JawnyTekst:	01001000	01000101	01001100	01001100	01001111
	H	**E**	**L**	**L**	**O**

Tekst „*JawnyTekst*” został odkryty.

Ponieważ *Klucz* jest sekwencją losową, nie ma sposobu, aby powiedzieć, czy pewien fragment sekwencji jest "0" lub "1", ponieważ są one równie prawdopodobne. Bez *Klucza*, biorąc pod uwagę bit *Zaszyfrowanego Tekstu*, nie mamy szans na odgadnięcie, jaki może być oryginalny bit. Dotyczy to pojedynczych bitów, całej wiadomości, każdego fragmentu naszej wiadomości. W przypadku, gdy część *Klucza* jest znana, możliwe jest odzyskanie odpowiedniej części w tekście jawnym (*JawnyTekst*), ale bez reszty *Klucza* nie brakuje informacji o pozostałych bitach tekstu jawnego (*JawnyTekst*). Dlatego wspomniana wcześniej kryptoanaliza nie jest skuteczna. W tym przypadku, aby odkryć pozostałą część oryginalnej wiadomości, potrzebna jest pozostała część *Klucza*. Wszystkie informacje przekazywane przez *Klucz* są potrzebne do odzyskania wszystkich informacji zawartych w tekście.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Problemy z algorytmem “One Time Pad”

Siła mechanizmu OTP polega na tym, że długość *Klucza* jest dokładnie równa długości tekstu oraz na użyciu go tylko jeden raz. Główną wadą jest to, że transmisja OTP wiąże się z tymi samymi problemami co wiadomość.

W niektórych przypadkach problem ten można łatwo rozwiązać, np. można dostarczyć do ambasady teczkę dyplomatyczną z tasiemką zawierającą bardzo długi klucz, a następnie zaszyfrować wiadomości, które będą wymieniane z tą ambasadą, zużywając stopniowo bity klucza. Są to bardzo osobliwe zastosowania, ale dość skuteczne.

Widać wyraźnie, że jeśli weźmiemy pod uwagę komunikację sieciową, to realne zastosowania „One Time Pad” są bardzo ograniczone. Istnieje potrzeba czegoś bardziej elastycznego, czegoś, co pozwala na wymianę klucza, który może być używany przez bardzo długi czas. Rozważmy każdy z tych problemów.

Transmisja offline z wyprzedzeniem

Jest to ogólny problem z kluczami: klucze muszą być w jakiś sposób uzgodnione przez strony. Ta potrzeba uzgodnienia kluczy jest najważniejszym problemem korzystania z kryptografii. Problem z kryptografią to nie tyle algorytmy i kryptoanaliza, co zarządzanie kluczami: czyli upewnienie się, że dostęp do kluczy mają wszyscy i tylko właściwe podmioty, a następnie ich wymiana z wyprzedzeniem i upewnienie się, że wymieniane klucze są poprawne.

Jednorazowe użycie

„One Time Pad” jest używany tylko raz, jak sama nazwa wskazuje jest to ważny aspekt i ograniczenie. Gdybyśmy próbowali użyć tego samego klucza na dwóch wiadomościach, osłabilibyśmy bezpieczeństwo metody.

Generowanie liczb losowych w systemie deterministycznym jest bardzo trudne

Komputery ogólnego przeznaczenia wykorzystują generowanie liczb pseudolosowych, w którym każda liczba jest obliczana z poprzedniej z pewnymi obliczeniami, co daje sekwencję liczb, które powtarzają się po bardzo długim okresie. To nie są prawdziwe liczby losowe, ponieważ komputer jest maszyną deterministyczną. Jednak w kryptografii ta funkcja jest szkodliwa. Jeśli możliwe jest wyprowadzenie kolejnej losowej liczby z poprzedniej, zabezpieczenia oferowane przez algorytmy lub protokoły kryptograficzne stają się znacznie mniej bezpieczne. Pisanie kodu, który implementuje algorytmy kryptograficzne, nie jest trywialne: należy zwrócić uwagę na wiele szczegółów i wszystkie wady, które zostały zidentyfikowane na przestrzeni lat przez programistów kodu, którymi tylko programiści w tej konkretnej dziedzinie nauczyli się zarządzać. Oznacza to, że zawsze złym pomysłem jest opracowywanie i/lub wdrażanie wewnętrznych algorytmów

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

kryptograficznych zamiast korzystania z dobrze przetestowanych, powszechnie używanych i szeroko zweryfikowanych bibliotek.

Inne algorytmy niż OTP

Pozostałe algorytmy używają kluczy o stałej długości. Jeśli używamy klucza *n*-bitowego, to mamy 2^n różnych kluczy, jeśli wartość n jest wystarczająco duża i po prostu zastosujesz atak brute force (czyli wypróbujesz wszystkie możliwe klucze), to znalezienie właściwego wymagałoby tylu lat, że "prawdopodobnie" treść wiadomości nie będzie już interesująca. Na przykład, przy jednym z najszybszych dostępnych superkomputerów, sprawdzenie 256-bitowego klucza wymagałoby 3.67×10^{55} lat. Oczywiście, używanie kluczy o odpowiedniej długości ma pewien dodatkowy koszt obliczeniowy nawet dla tych, którzy legalnie szyfrują i odszyfrowują, ale wpływ jest ograniczony.

Istnieją algorytmy *symetryczne* i *asymetryczne*.

Algorytmy symetryczne

Najbardziej typową i starą rodziną algorytmów kryptograficznych, tych, o których zwykle myślimy, gdy myślimy o szyfrowaniu wiadomości, są ***algorytmy symetryczne***, nazywane tak od symetrii operacji szyfrowania i deszyfrowania. Algorytmy te wykorzystują jeden klucz, który musi być znany zarówno nadawcy, jak i odbiorcy; służy on do zaszyfrowania wiadomości, a następnie do jej odszyfrowania.

Prosty schemat symetryczny mógłby być taki sam jak OTP, ponieważ OTP jest algorytmem symetrycznym, z tą dużą różnicą, że *Klucz* ma stałą długość (np. 256 bitów) i jest w jakiś sposób powtarzany, ewentualnie po jakiejś złożonej zmianie za każdym razem:

- **Szyfrowanie:**
  funkcja_szyfrująca (JawnyTekst, Klucz) → ZaszyfrowanyTekst
- **Deszyfrowanie:**
  funkcja_deszyfrująca(ZaszyfrowanyTekst, Klucz) → JawnyTekst

Mówiąc bardziej ogólnie, nawet jeśli używamy tego samego klucza, funkcja używana do odszyfrowania nie musi być tą samą używaną do szyfrowania. Na przykład, jeśli rozważymy szyfr Cezara, jest oczywiste, że funkcja, która odszyfrowuje wiadomość, przesuwa alfabet w lewo, a nie w prawo, ale deszyfrowanie używa tego samego klucza: liczby 3.

<u>Wybrane najważniejsze algorytmy symetryczne</u>

DES (Data Encryption Standard)

Ten historyczny algorytm, obecnie przestarzały, używa 56-bitowego klucza, który pod koniec lat 70. (kiedy algorytm został opracowany) był absolutnie wystarczający dla wielu zastosowań,

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

ponieważ atak brute force na 56-bitowy klucz, aby uzyskać wiadomość za pomocą zasobów obliczeniowych dostępnych w tym czasie i w kontekstach, w których ten algorytm był używany, był niepraktyczny. 40 lat później algorytm DES nie wytrzymałby nawet kilku minut pracy domowego komputera. Jest to bardzo ważny aspekt: długość klucza jest wybierana na podstawie rzeczywistych dostępnych możliwości obliczeniowych. Klucz, który jest uważany za wystarczająco solidny teraz, może nie być już tak silny za 20 lat. Ogólnie rzecz biorąc, mamy tendencję do zwiększania długości klucza, aby zagwarantować, że istnieje pewien margines, zarówno dlatego, że możliwości obliczeniowe rosną, ale także dlatego, że kryptoanaliza jako dyscyplina robi postępy i może być w stanie znaleźć słabości w algorytmie. To jest dość ciekawy aspekt. Wbrew pozorom, w normalnych warunkach algorytm kryptograficzny nie ulega awarii nagle, tzn. w dniu odkrycia wady w algorytmie natychmiast zmienia się on z bezpiecznego w zupełnie niezabezpieczony i każdy może za chwilę znaleźć tekst jawny. Co się dzieje, jest to, że badacze analizują algorytm i może znaleźć małe słabości, które mogą ułatwić atakowanie go w niektórych przypadkach. Następnie ktoś może znaleźć jakąś optymalizację algorytmu, aby rozwiązać problem. W miarę upływu czasu algorytm jest stopniowo osłabiany z powodu nierozwiązywalnych problemów, aż przestaje być odpowiedni do pewnych zastosowań lub w ogóle. Następuje stopniowa erozja obrony algorytmów kryptograficznych, a to stopniowe osłabienie jest trudniejszym problemem do opanowania, bo oczywiście nie możemy wiedzieć, jaki będzie postęp w przyszłości z punktu widzenia kryptoanalizy.

AES (Advanced Encryption Standard znany również pod swoją oryginalną nazwą Rijndael)

AES został wybrany w oparciu nie tylko o solidność, ale także o efektywność implementacji w typowych architekturach. Nie szukano tajnego algorytmu, ponieważ tajny algorytm nie jest uważany za bardziej odporny niż nietajny: przeciwnie, algorytm miał być analizowany przez jak największą liczbę wartościowych kryptoanalityków, aby upewnić się, że jest wewnętrznie odporny. Algorytm bezpieczny, ale zbyt wolny, zbyt kosztowny pod względem zasobów obliczeniowych, nie nadawałby się do użytku. To ograniczenie zostało postawione jako najtrudniejsze do pokonania przez konkurentów, w porównaniu z ograniczeniem solidności.

Algorytmy asymetryczne (lub klucza publicznego)

Algorytmy asymetryczne nie używają jednego klucza, ale dwóch: jeden klucz służy do szyfrowania, a drugi do odszyfrowania. Dlatego w wymianie wiadomości nadawca i odbiorca nie używają tego samego klucza. Klucze te są generowane i zarządzane w parach: to, co szyfruje klucz (dowolny z pary), może być odszyfrowane tylko przez drugi klucz z pary.

Powód stosowania algorytmów asymetrycznych wynika z problemów, jakie ma kryptografia symetryczna:

- Poufna komunikacja pomiędzy dwoma podmiotami wymaga współdzielenia unikalnego klucza tylko między nimi. Jeśli podmiotów jest n wówczas $n \cdot (n-1)/2$ wymagane są różne

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

klucze, aby umożliwić poufną komunikację między każdą parą stron.

- Zarówno nadawca, jak i odbiorca znają swój wspólny klucz, więc *niezaprzeczalność* (zaprzeczenie jest zaprzeczeniem bycia autorem danej wiadomości) nie jest możliwa.

Pierwszy problem nie dotyczy liczby kluczy, którymi każdy podmiot musi zarządzać, problemem jest ich *wymiana*. Każdy podmiot musi uzgodnić klucz z każdym z pozostałych n-1 podmiotów, co jest w istocie skomplikowaną operacją; czyni to kryptografię symetryczną niepraktyczną dla człowieka.

Niezaprzeczalność polega na tym, że często chcemy się upewnić, że pewna wiadomość została rzeczywiście wysłana przez pewien podmiot. Jest oczywiste, że jeśli podmiot otrzymuje wiadomość od jakiegoś innego podmiotu, a oba podmioty jako jedyne znają klucz użyty do jej zaszyfrowania, to każdy podmiot jest pewien, że drugi podmiot jest autorem wiadomości. I odwrotnie, ponieważ znają tylko klucz, każdy podmiot wie, że tylko drugi podmiot może odczytać wiadomość. Problem pojawia się, gdy trzeba to udowodnić stronie trzeciej, na potrzeby jakiegoś sporu sądowego, ponieważ każdy podmiot jest w stanie stworzyć wiadomość, zaszyfrować ją, a następnie powiedzieć, że druga strona jest autorem. Problem *niezaprzeczalności* nie jest możliwy do rozwiązania przy użyciu kryptografii symetrycznej.

Schemat asymetryczny jest podobny do symetrycznego, ale używane klucze to dwa, które są generowane w parze:

- **Szyfrowanie:** *funkcja_szyfrująca (JawnyTekst*, ***Klucz1***) → *ZaszyfrowanyTekst*
- **Deszyfrowanie:** *funkcja_deszyfrująca* (*ZaszyfrowanyTekst*, ***Klucz2***) → *JawnyTekst*

Nazwy ***Klucz1*** i ***Klucz2*** zostały użyte zamiast *KluczSzyfrujący* i *KluczDeszyfrujący*, ponieważ oba klucze są matematycznie wymienne. Podmiot generujący klucze na ogół zachowuje jeden z nich jako prywatny/tajny (zwany kluczem prywatnym), a drugi upublicznia (zwany kluczem publicznym), stąd nazwa klasy algorytmów, które nazywane są również ***algorytmem klucza publicznego***. Więcej o pojęciu „publiczności” zostanie omówione później.

Funkcja używana do szyfrowania nie musi być taka sama jak ta używana do odszyfrowania. Ważne jest to, że razem stanowią algorytm kryptograficzny, jedna część opiera się na jednym kluczu, a druga na drugim kluczu.

Te dwa klucze mają takie właściwości matematyczne, że odkrycie jednego klucza z drugiego jest dość skomplikowane i wymaga ogromnej ilości czasu. W szczególności, nawet znając *klucz publiczny*, *JawnyTekst* i *ZaszyfrowanyTekst*, rekonstrukcja *klucza prywatnego* wymagałaby nieracjonalnej ilości czasu.

<u>Wykorzystanie Algorytmu Asymetrycznego</u>

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Załóżmy (szczegóły zostaną podane później), że *klucz publiczny* jest publicznie dostępny, a jego właściciel jest jedynym, który zna odpowiadający mu klucz prywatny. Ponieważ klucz publiczny jest dostępny dla każdego, każdy może go użyć do zaszyfrowania wiadomości, ale tylko właściciel odpowiedniego klucza prywatnego będzie mógł ją odszyfrować, co zapewnia **poufność** transmisji wiadomości.

Tak więc przeszliśmy od sytuacji, w której jeden podmiot potrzebował *n* różnych kluczy, aby poufnie komunikować się z *n* innymi podmiotami, do sytuacji, w której tylko jeden klucz (klucz publiczny odbiorcy) jest potrzebny każdemu, aby wysłać poufną wiadomość do podmiotu.

Aby *n* podmiotów mogło się ze sobą komunikować, każdy podmiot musi mieć parę kluczy, więc całkowita liczba kluczy wynosi $2 \cdot n$. Interesujące jest nie tyle to, że klucze zmniejszyły swoją liczbę, ale to, że będąc kluczami publicznymi mogą być przekazywane w znacznie prostszy sposób. Jeśli ktoś inny pozna te klucze publiczne, to nie jest to żaden problem. Dlatego te klucze mogą być naprawdę publikowane, udostępniane wszystkim i każdy może z nich korzystać bez osłabiania ochrony wiadomości.

Jeżeli podmiot jest jedynym posiadaczem klucza prywatnego (tajnego), którego publiczny odpowiednik jest publicznie dostępny, to każdy może zweryfikować (odszyfrować) wiadomości zaszyfrowane kluczem prywatnym podmiotu. Wówczas jeśli możliwe jest użycie klucza publicznego podmiotu do odszyfrowania wiadomości, oznacza to, że tylko podmiot posiadający odpowiadający mu klucz prywatny mógł zaszyfrować wiadomość, zapewnia to **niezaprzeczalność** wiadomości.

Pozostaje jeszcze do rozwiązania problem: zapewnienie, że klucz publiczny jest rzeczywiście związany z uprawnionym podmiotem. W tym przypadku musi być zapewniona metoda certyfikacji **super partes**, którą zapewnia **infrastruktura klucza publicznego** (**PKI**), omówiona później.

Jako uwaga dodatkowa, algorytmy asymetryczne mogą być również używane jako *algorytm challenge-response*. Aby podmiot A mógł ***uwierzytelnić*** się podmiotowi B (udowodnić, że podmiot jest rzeczywiście tym, za kogo się podaje) , A wysyła (w sposób jawny) losową liczbę do B; następnie B szyfruje ją swoim kluczem prywatnym i wysyła z powrotem do A. Jeśli A jest w stanie odszyfrować wiadomość za pomocą klucza publicznego B, A jest pewien, że drugą stroną jest B, ponieważ jest jedynym, który może zaszyfrować coś, co klucz publiczny B może odszyfrować.

Zalety i wady

Całkowita liczba kluczy jest znacznie zmniejszona, co zmniejsza złożoność zagwarantowania prawidłowego zarządzania wszystkimi kluczami (a tajny klucz zna tylko właściciel). Ponieważ klucze publiczne są udostępniane publicznie, nie ma potrzeby zawierania specjalnych umów między różnymi podmiotami. Takie wykorzystanie kluczy publicznych jest na przykład bardzo ważne w handlu elektronicznym, ponieważ handel elektroniczny nie może wymagać istnienia apriorycznego

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

porozumienia sprzedawcy z klientami.

Uzyskanie tego związku między dwoma kluczami i jednocześnie zagwarantowanie, że jeden nie może być obliczony na podstawie drugiego, a to, co jest zaszyfrowane jednym, jest odszyfrowane drugim, powoduje, że algorytmy asymetryczne są znacznie bardziej złożone pod względem problemów matematycznych niż algorytmy symetryczne. Te problemy matematyczne zasadniczo wymagają wolniejszych algorytmów niż algorytmy symetryczne i stosowania dłuższych kluczy, przynajmniej przy obecnie stosowanych algorytmach. Jeśli więc z jednej strony mamy do czynienia z większą elastycznością, to z drugiej strony są one bardziej uciążliwe pod względem zasobów obliczeniowych.

Algorytm RSA *(od inicjałów Ronald **R**ivest, Adi **S**hamir, and Leonard **A**dleman)*
Ten algorytm opiera się na złożoności faktoryzacji liczb pierwszych: jest to złożony/trudny problem, aby znaleźć dwie liczby pierwsze z wyniku. Wiemy również, że nie jest łatwo znaleźć bardzo duże liczby pierwsze, ponieważ nie ma reguły, która pozwala na znalezienie liczb pierwszych innych niż próba czynnika i odkrycie, czy są one pierwsze. Oznacza to, że praca na bardzo dużych liczbach, odkrycie, które są dwie liczby pierwsze, z których pochodzi dana liczba, jest problemem wymagającym znacznych zasobów obliczeniowych. Z drugiej strony, uzasadnione operacje mogą być wykonywane przy ograniczonych zasobach obliczeniowych.

Algorytm Elliptic Cryptography *(ECC - Elliptic Curve Cryptography)*

„Trudnym problemem" w tym przypadku jest obliczenie logarytmu dyskretnego (w skończonym polu). "Skończone pole" oznacza, że wartości całkowite znajdują się w ograniczonym zbiorze (bardzo przybliżona definicja).

Problem ECC jest bardziej złożony niż faktoryzacja pierwiastków, więc otrzymujesz to samo bezpieczeństwo z krótszymi kluczami. Algorytmy ECC są szybsze niż RSA.

<u>Porównanie algorytmów Symetrycznych iu Asymetrycznych</u>
Algorytmy asymetryczne wykorzystują problemy matematyczne inne niż w kryptografii symetrycznej. Oznacza to, że długości kluczy nie są porównywalne z tymi w kryptografii symetrycznej. Problem z szyfrowaniem symetrycznym polega tylko na tym, że klucz jest wystarczająco długi, aby uniemożliwić wypróbowanie wszystkich możliwych liczb, które mogłyby pasować do klucza, więc przy 128-bitowym kluczu istnieją 2^{128} (to jest $3{,}4 \cdot 10^{38}$) możliwe klucze. Dzisiejsze algorytmy symetryczne zazwyczaj wykorzystują klucze 128-, 256- a czasem 512-bitowe.

Jeśli rozważymy na przykład RSA, która wykorzystuje liczby pierwsze, to jasne jest, że nie wszystkie klucze są możliwe, ponieważ są związane z obecnością liczb pierwszych. Nie wszystkie liczby są pierwsze, więc nie wszystkie liczby są możliwymi kluczami. Bardzo upraszczając, aby mieć taką samą odporność algorytmu (a więc taką samą trudność z punktu widzenia obliczeniowego) 128-

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

bitowego algorytmu symetrycznego, algorytm RSA potrzebuje klucza rzędu 1024 bitów. Dzisiejsze algorytmy asymetryczne zazwyczaj używają kluczy 1024, 2048 i 4096 bitowych.

Funkcje haszujące

Funkcje haszujące obliczają krótkie o stałej długości podsumowanie tekstu o dowolnej długości; to podsumowanie nazywane jest **skrótem**. Funkcja haszująca musi być prosta i szybka do obliczenia.

Funkcje haszujące są "funkcjami jednokierunkowymi": ze skrótu można bardzo skomplikowanie i powoli odzyskać *możliwy* tekst, który go generuje, ale ten tekst jest mało użyteczny. Jest to konsekwencja tego, że oryginalny skrót tekstu jest większy niż skrót, więc praktycznie nieskończone ciągi bajtów mogą skutkować określonym skrótem. Nawet jeśli okaże się, że jakiś tekst ma dany skrót, jest niezwykle mało prawdopodobne, że jest to tekst oryginalny lub nawet ma sens. Zbudowanie sensownej wiadomości z danego skrótu jest dość trudne, każda modyfikacja jednego bitu wiadomości, daje zupełnie inny skrót.

Kontrola integralności

W przypadku wiadomości M, jej skrót D jest obliczany za pomocą funkcji skrótu H: D = H(M)

Bezpieczne przesłanie skrótu D pozwala zapewnić, że wiadomość M jest poprawna: odbiorca wiadomości M oblicza na niej skrót i porównuje go z otrzymanym (D); jeśli są one identyczne, wiadomość nie została zmodyfikowana. Istotne jest, aby skrót był przekazywany w sposób bezpieczny, w przeciwnym razie nie ma żadnej ochrony.

Mechanizm ten jest również stosowany jako weryfikacja pobranego oprogramowania zamiast innych mniej wydajnych (ale szybszych) algorytmów znanych jako CRC (Cyclic Redundancy Checks), powszechnych w protokołach sieciowych.

Wybrane własności funkcji haszujących

Poniżej opisano dwa najbardziej znane algorytmy haszujące.

MD5 (Message Digest #5, 1991)

MD5 wymaga niewielkiej mocy obliczeniowej (w porównaniu do większości innych algorytmów haszujących) i wytwarza 128-bitowy hash. Z czasem był on stopniowo osłabiany, aż stał się zupełnie niewiarygodny dla celów bezpieczeństwa (obecnie w ciągu kilku sekund można znaleźć dwie różne wiadomości z tym samym skrótem MD5 - co nazywa się kolizją), mimo to jest wciąż przydatny w zastosowaniach niekryptograficznych, na przykład do wykrywania błędów w transmisji zamiast bardziej powszechnego (i słabego, ale szybszego) CRC-32.

SHA-3 (Secure Hash Algorithm #3, 2015)

SHA to właściwie rodzina bezpiecznych algorytmów haszujących, SHA-3 jest ostatnią wersją i

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

wynikiem konkursu wśród kryptografów. Algorytm może być dostrojony równoważąc bezpieczeństwo z szybkością; produkuje wartości skrótu 224, 256, 334 i 512 bitów. Jest 2-3 razy wolniejszy od MD5, ale jak dotąd nie znaleziono słabych punktów.

Podpisy cyfrowe

Podpisy cyfrowe, zwane w ustawodawstwie niektórych krajów podpisem elektronicznym, pobierają wiadomość, obliczają skrót/hash i szyfrują go (tylko hash) tajnym kluczem algorytmu asymetrycznego. Ten hasz podpisany tajnym kluczem nazywany jest **podpisem**. Podpis cyfrowy nie szyfruje oryginalnej wiadomości, pozwala jedynie zagwarantować, że odpowiadający jej tekst nie został zmodyfikowany, ponieważ w przeciwnym razie zmieniłby się skrót. Gwarantuje również możliwość weryfikacji pochodzenia (**niezaprzeczalność**), gdyż w rzeczywistości został zaszyfrowany kluczem prywatnym algorytmu asymetrycznego. Oznacza to, że każdy może sprawdzić, czy tekst jest nienaruszony i czy rzeczywiście został wygenerowany przez określony podmiot.

Podobne do podpisów cyfrowych są kody uwierzytelniania wiadomości (MAC, Message Authentication Codes). Ich celem jest również zapewnienie integralności i autentyczności wiadomości. Skrót obliczany jest poprzez haszowanie tekstu, który ma być chroniony oraz klucza *algorytmu symetrycznego* (nie ma pewności niezaprzeczalności, więc nadaje się tylko wtedy, gdy ta cecha nie jest potrzebna). Jednym z najbardziej znanych algorytmów jest **HMAC**: Keyed-hash MAC.

Infrastruktura klucza publicznego

Wymagania dotyczące poufności, integralności i autentyczności operacji szyfrowania asymetrycznego i podpisu cyfrowego wymagają ustanowienia relacji zaufania pomiędzy zaangażowanymi stronami. Ponieważ często strony nie miały wcześniej kontaktu, zaufana (przez nie) trzecia strona musi zapewnić potrzebne gwarancje, innymi słowy ta trzecia strona musi zapewnić autentyczność właściciela klucza publicznego.

W celu zapewnienia autentyczności podmiotu, CA podpisuje *certyfikat cyfrowy*, który łączy podmiot z jego kluczem publicznym (jest to koncepcyjnie podobne do dokumentu tożsamości osoby). Podpis stosowany przez CA jest podpisem cyfrowym, a więc jest obliczany przy użyciu klucza prywatnego CA. Oczywiście certyfikat cyfrowy musi być potwierdzony kluczem publicznym CA.

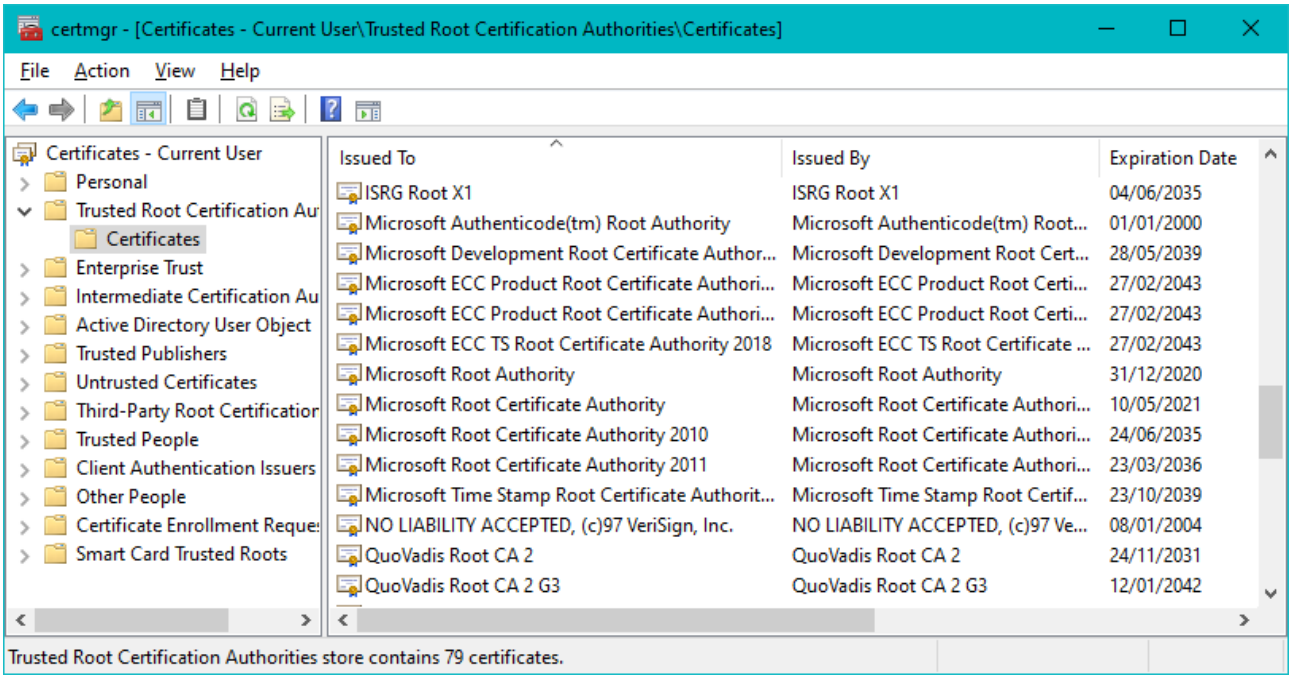
Infrastruktura klucza publicznego (PKI) jest hierarchiczną strukturą urzędów certyfikacji, gdzie każdy urząd podpisuje swój klucz publiczny przez urząd poprzedniego poziomu (który używa do tego celu swojego klucza prywatnego), proces ten przebiega aż do *urzędu głównego* (Root Certificate Authority), który samodzielnie podpisuje certyfikat posiadający jego klucz publiczny przy użyciu swojego klucza prywatnego.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Model hierarchiczny umożliwia budowanie hierarchii o dowolnej głębokości. Na przykład w ramach kraju możemy mieć gminne, ..., regionalne urzędy certyfikacji, aż do krajowego głównego CA.

Certyfikat główny CA musi być upubliczniony w wielu miejscach i na wiele sposobów, aby uniemożliwić zastąpienie go sfałszowanym. Jako przykład można podać, że same systemy operacyjne są dostarczane z zestawem głównych i zaufanych pośrednich CA. Na rysunku 1 przedstawiono fragment listy certyfikatów głównych z systemu operacyjnego MS Windows 10.



Rysunek 1. Menedżer Certyfikatów w Windows 10

Komponenty PKI

Głównym aktorem jest **Urząd Certyfikacji (CA),** zarządza on kluczami kryptograficznymi i certyfikatami PKI. Jest to komponent, który przechowuje prawdziwe klucze i który następnie wykonuje operacje kryptograficzne. Oczywiście w tym sensie jest to również najbardziej krytyczny element urzędu certyfikacji.

Registration Authority (RA) zarządza żądaniami certyfikatów i dostarcza wydane certyfikaty. Zajmuje się dystrybucją tokenów kryptograficznych i oprogramowania (jest front-endem CA). Jest to jakby licznik, do którego może udać się obywatel z żądaniem wydania certyfikatu cyfrowego i z którego można go otrzymać. Może to być fizyczna lada lub strona internetowa, rodzaj związany jest z różnymi poziomami bezpieczeństwa. Ponadto RA na ogół może również pomagać użytkownikowi w bezpiecznym generowaniu pary kluczy i zarządzaniu tymi certyfikatami, dostarczając narzędzia programowe lub sprzętowe, które wspierają wnioskodawcę w generowaniu podpisów, szyfrowaniu dokumentów, weryfikacji podpisów, odszyfrowywaniu

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

informacji.

Serwer certyfikatów to katalog, w którym publikowane są certyfikaty użytkowników i w pewien sposób udostępniane do wyszukiwania. Katalog ten jest zgodny ze standardem X.500 i jest odszukiwany za pomocą protokołu LDAP.

Certifikat cyfrowy

Strukturę i format certyfikatu cyfrowego opisuje standard ITU-T X.509 (jest to standard Międzynarodowego Związku Telekomunikacyjnego oznaczony numerem X.509).

Certyfikat cyfrowy X.509 musi zawierać co najmniej następujące pola:

- **DN** (nazwa wyróżniająca) użytkownik: Gaius Julius Caesar
- **CN** (nazwa wspólna): Julius Caesar
- **O** (organizacja): Rząd
- **OU** (jednostka organizacyjna): Senat
- **C** (kraj): Cesarstwo Rzymskie

Jest to klasyczny wzór certyfikatu spełniającego standard X.500, którego członkiem jest X.509. Certyfikat może mieć inne atrybuty (np. email). Może być adres IP lub adres URL odpowiadający ścieżce katalogu X.500 (np. LDAP - Lightweight Directory Access Protocol):

IT / Cesarstwo Rzymskie

/ Rząd / Julius Caesar / Gaius Julius Caesar /

Certyfikat cyfrowy X.509 musi również zawierać co najmniej następujące pola:

- **Wydawca DN**: nazwa urzędu w formacie X.500 (urząd certyfikacji posiada również swój wyróżnik, który pojawia się w certyfikacie). Oczywiście format, za pomocą którego jest on reprezentowany, jest taki sam jak nazwa wyróżniona podmiotu, którego dotyczy certyfikat.
- **Numer Seryjny**: numer seryjny certyfikatu posiadanego przez urząd (każdy certyfikat jest jednoznacznie identyfikowany przez numer seryjny nadany mu przez urząd. Z drugiej strony, para (***numer seryjny & nazwa wyróżniająca wydawcy***) jednoznacznie identyfikuje certyfikat w całej mnogości certyfikatów, które mogą być wydawane na całym świecie jako unikalna para).

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

- **NotBefore:** data aktywacji
- **NotAfter:** data wygaśnięcia

Są to dwie informacje ściśle ze sobą powiązane. Certyfikat ma zatem datę rozpoczęcia życia i datę ważności, po przekroczeniu której traci ważność. Powód wynika z chęci ochrony kluczy przed nadmiernym użyciem.

- **Public Key:** klucz publiczny podmiotu certyfikowanego (certyfikat zawiera dane osobowe i klucz publiczny)
- **Public Key Algorithm:** algorytm, dla którego klucz publiczny jest odpowiedni
- **Signature Algorithm:** algorytm użyty przez Urząd do podpisania certyfikatu
- **Signature:** podpis złożony na certyfikacie przez urząd certyfikacji (a więc element gwarantujący autentyczność samego certyfikatu)
- **KeyUsage:** cele, do których klucz może być wykorzystywany

Generowanie i unieważnianie certyfikatów

Standard RFC 2314 (z PKCS # 10) określa format wiadomości ***certificateRequest*** i zawiera klucz publiczny oraz dane wnioskodawcy (oba podane RA). Wiadomość z żądaniem wymaga tzw. dowodu posiadania klucza prywatnego: wiadomość ***certificateRequest*** musi być podpisana kluczem prywatnym wnioskodawcy. Organ weryfikując ten podpis merytorycznie sprawdza, czy wnioskodawca rzeczywiście posiada ten klucz (w przeciwnym razie wnioskodawca nie byłby w stanie złożyć tego podpisu).

W dacie odpowiadającej atrybutowi "notAfter" Certyfikat jest uznawany za "wygasły". Wygaśnięcie jest mechanizmem, który służy również zapewnieniu, że te same klucze asymetryczne związane z certyfikatem cyfrowym nie mogą być używane przez zbyt długi okres czasu, co mogłoby narazić je na szereg ataków kryptoanalitycznych.

Wygaśnięcie może odnosić się do:

- **Certyfikatu** (skojarzenie podmiot-klucz)
- **Powiązanych kluczy**, w stosunku do zastosowania, do którego są przeznaczone (klucz może być używany tylko do pewnych celów, a do innych nie)

Wygasły certyfikat nie może być używany przez jego właściciela, ponieważ wygasł jego użycie jest w praktyce zabronione. W rzeczywistości nic nie stoi na przeszkodzie, aby posiadacz certyfikatu wykorzystał go do innych celów, ale wszyscy, którzy się z nim komunikują, w jakiś sposób wykryją, że ten certyfikat wygasł i nie zaakceptują go. Certyfikat jest oczywiście dostępny w celu weryfikacji i odszyfrowania wiadomości wydanych przed wygaśnięciem. W niektórych przypadkach możliwe

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

jest "odnowienie" certyfikatu, co zazwyczaj nie odbywa się automatycznie.

Czasami certyfikat może stać się nieważny przed wygaśnięciem poprzez ***unieważnienie*** (dzieje się to przed wartością atrybutu "notAfter". Powodem może być np. kompromitacja klucza powiązanego z certyfikatem, podmiot nie jest już powiązany z firmą lub zmienioną jednostką, CA, który wydał certyfikat (lub jeden z jego CA wyższego poziomu) został skompromitowany lub inne powody. Unieważniony certyfikat nie może być już używany, z wyjątkiem sytuacji, gdy jest po prostu trzymany "w zawieszeniu". Jest to szczególna forma unieważnienia, w której certyfikat jest tymczasowo umieszczany na liście CRL z określonych powodów (takich jak " możliwe naruszenie klucza", "żądanie unieważnienia przez osobę inną niż właściciel", "tymczasowe nieużywanie certyfikatu" np. na urlopie)" itp. i jego ważność może być później przywrócona. W przypadku "reaktywacji" po zawieszeniu, do listy CRL wstawiany jest nowy wpis ze specjalnym kodem przyczyny: ***removeFromCRL***, ponieważ żaden wpis nie może być nigdy usunięty z listy CRL.

Aby poinformować użytkowników PKI o utracie ważności certyfikatu, jego numer seryjny jest umieszczany na liście CRL (***Certificate Revocation List***). Lista ta jest wydawana okresowo przez wystawiający ją Urząd Certyfikacji i propagowana (aktywnie lub pasywnie) do wszystkich użytkowników, tak aby wszystkie podmioty posiadające taki certyfikat nie mogły go już używać. Lista CRL jest zgodna ze standardem ITU-T X.509 - rfc3280 - rfc4325 co oznacza, że jej format jest ustandaryzowany. Jest ona okresowo podpisywana przez CA i zawiera pole "nextUpdate", które zawiera datę i godzinę, w której nowa lista unieważnień certyfikatów będzie dostępna. Jeżeli data ta należy do przeszłości oznacza to, że nasza lista CRL jest "wygasła", jeżeli data należy do przyszłości wiemy kiedy będziemy musieli zaktualizować naszą listę unieważnionych certyfikatów. Każdy unieważniony certyfikat zawiera powód unieważnienia. Lista CRL jest udostępniana jako usługa poprzez protokół LDAP.

Ponieważ listy CRL mogą stać się dość duże, zwłaszcza jeśli urząd certyfikacji ma setki tysięcy lub nawet miliony użytkowników, można je podzielić i CA udostępnia tylko fragmenty listy unieważnień certyfikatów, na przykład odnoszące się do pewnego okresu czasu. W związku z tym, gdy zachodzi konieczność weryfikacji certyfikatu, obywatel pobiera jedyny interesujący go fragment listy CRL. Zamiast pobierać listę CRL lub jej fragment, można skorzystać z usługi implementującej protokół OCSP (***Online Certificate Status Protocol*** - RFC 2560), aby dowiedzieć się, czy dany numer certyfikatu znajduje się na liście CRL. Odpowiedzi są podpisywane przez CA, aby były godne zaufania.

Łańcuch zaufania

Aby zweryfikować certyfikat, podmiot musi zweryfikować cały łańcuch pośrednich CA, zaczynając od zaufanego, ewentualnie głównego CA. Najpierw sprawdzany certyfikat jest weryfikowany względem CA wystawcy, następnie certyfikat CA jest weryfikowany względem jego CA macierzystego i tak dalej aż do pośredniego zaufanego (przez użytkownika) CA lub samego CA

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

głównego.

Sposób użycia

Właściwie klucze mogą być używane do różnych celów, a certyfikat określa dozwolone zastosowania, wśród których mamy:

- **nonRepudiation (niezaprzeczalność)**: klucz może być wykorzystany do złożenia podpisu o wartości prawnej, który może być następnie przeciwstawiony osobom trzecim jako podpis odręczny
- **dataEncipherment:** klucz może być użyty do zaszyfrowania danych
- **keyAgreement:** klucz może być użyty do bezpiecznej wymiany kluczy

Znakowanie czasem (TimeStamping)

PKI może być również wykorzystane do znakowania czasem dokumentu zgodnie z protokołem ***Time Stamp Protocol*** (RFC 3161), aby udowodnić jego istnienie w jakimś punkcie w przeszłości. CA podpisuje dokument i zawiera znak czasowy. Podpis zapewnia również, że dokument nie został zmieniony po umieszczeniu znacznika czasu. Znacznik czasowy może przedłużyć ważność podpisu elektronicznego poza czas życia certyfikatu cyfrowego związanego z użytymi kluczami, np. poprzez żądanie umieszczenia na umowie nowego znacznika czasowego w każdym mijającym roku. Umieszczanie będzie odbywało się co roku przy użyciu technik kryptograficznych o sile adekwatnej do okresu, w którym stosowany jest sam podpis.

Standardowe mechanizmy i formaty zapisu

Poniżej przedstawiono dwa powszechnie używane standardowe mechanizmy i formaty.

PKCS #7 - Public key cryptography standard #7

PKCS #7 opisuje format wiadomości/plików podpisanych, zaszyfrowanych (za pomocą szyfrowania symetrycznego i asymetrycznego) i możliwych do zweryfikowania za pomocą MAC (dane przetrawione)

S/MIME - Secure Multipurpose Mail Extension

Secure Multipurpose Mail Extension jest rozszerzeniem standardu MIME (RFC 2045, 2046 i 2047 jako rozszerzenie RFC 822) i opisuje szyfrowanie wiadomości pocztowych. Encje PKCS #7 są merytorycznie przetłumaczone na nowe encje MIME, a więc opisują nowe kontenery, w których możemy znaleźć porcje podpisanych lub zaszyfrowanych wiadomości e-mail lub podpisanych lub zaszyfrowanych załączników.

Wymiana i format zapisu

Standard PKCS #12 (*Personal Information Exchange Syntax Standard #12*) definiuje format

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

wymiany i bezpiecznego przechowywania (w pliku) kluczy kryptograficznych i certyfikatów cyfrowych (zaufanego łańcucha certyfikatów). Przechowywanie jest bezpieczne, ponieważ opiera się na szyfrowaniu zarówno kluczy, jak i certyfikatów oraz całego łańcucha certyfikatów, które podmiot wybiera jako zaufane. Jest to format używany przez przeglądarki, klientów poczty elektronicznej i systemy operacyjne, gdy użytkownik eksportuje podpis i klucz szyfrujący oraz certyfikat, np. w celu wykonania kopii bezpieczeństwa lub skopiowania na inne urządzenie.

PKCS #12 szyfruje dane algorytmem symetrycznym, którego klucz pochodzi z hasła podanego przez użytkownika. Gwarantuje integralność treści poprzez HMAC, lub poprzez podpis cyfrowy.

Korzyści wynikające ze stosowania PKCS #12 to:

- **nie ma kosztów wdrożenia**, ponieważ jest to format dostępny w wielu bibliotekach na wielu urządzeniach (od komputerów po telefony komórkowe)
- **są już obsługiwane przez większość systemów operacyjnych** i oprogramowania, oznacza to, że są natychmiast dostępne
- **wykorzystuje "silne" algorytmy symetryczne**
- jest **przenośny na różnych urządzeniach**, można przenieść plik na inne urządzenia lub skopiować go na wszystkie urządzenia, których używamy, aby mieć klucze zawsze dostępne

Są też pewne wady:

- **może zostać usunięty, skopiowany, przekazany,** jest to ważny problem, ponieważ można go ukraść, wymazać, skopiować, aby go zabrać lub po prostu uczynić niedostępnym
- **klucz prywatny musi być wykorzystywany bezpośrednio przez oprogramowanie podpisujące/deszyfrujące**, oznacza to, że oprogramowanie używające deszyfruje ten plik i ma bezpośredni dostęp do klucza, więc jest moment, w którym klucz jest w systemie czysty, niezabezpieczony, choć może to tylko pamięć lotna, która jest najmniej dostępnym obszarem; jednak jest on dostępny w systemie w postaci czystego tekstu w jakiś sposób i jest to bardzo silne ograniczenie tego formatu
- **możliwość przeprowadzenia ataku brute force**, np. w przypadku, gdy ktoś ukradnie plik PKCS #12, może wypróbować wszystkie możliwe hasła w osobnym systemie komputerowym

Tokeny kryptograficzne

Tokeny kryptograficzne to urządzenia sprzętowe, które natywnie implementują algorytmy

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

szyfrujące. Nie pozwalają one na bezpośredni dostęp do materiału kryptograficznego z zewnątrz, są jedyną możliwością dostępu i użycia klucza. Co więcej, polecenia te mogą być w zasadzie wykonywane dopiero po uwierzytelnieniu się użytkownika, a także systemu w jego imieniu, do tokena, na przykład poprzez wprowadzenie kodu PIN. Tokeny mogą mieć niewielki system operacyjny, za pośrednictwem którego użytkownik żąda operacji generowania kluczy, szyfrowania i podpisywania danych. Bezpieczne zniszczenie tokena oznacza często nie tylko jego unieważnienie, ale także brak możliwości dostępu do obszarów pamięci, w których zapisany był klucz, tak aby uniknąć wszelkich form ataku, których nie moglibyśmy sobie dzisiaj wyobrazić. System operacyjny i sprzęt tokena może być certyfikowany według określonych standardów bezpieczeństwa.

Mechanizm "wyzwanie-odpowiedź" (challenge/response)

Mechanizm "wyzwanie-odpowiedź" pozwala dwóm stronom na uwierzytelnienie bez ujawniania użytecznych informacji na temat kanału komunikacyjnego. Idea polega na tym, że jedna ze stron A "wyzywa" drugą stronę B do operowania na losowej liczbie (zwanej **nonce**, ponieważ jest to liczba użyta tylko raz) za pomocą własnego klucza, na przykład za pomocą funkcji jednokierunkowej H. Następnie B zwraca H(nonce) do A. Jeśli B zwraca to, czego oczekuje A, to A jest pewien tożsamości B. W najprostszym przypadku B szyfruje nonce tajnym kluczem znanym A. A, który ma ten sam klucz, z kolei stosuje funkcję do nonce używając tego samego klucza i sprawdza, czy liczba podana mu przez B odpowiada tej, którą obliczył.

Jest to uwierzytelnianie na podstawie hasła, które nie zostało przesłane kanałem, więc nie ma możliwości śledzenia klucza. To co czyni ten mechanizm skutecznym to fakt, że liczba losowa jest inna dla każdej sesji. Ponieważ informacja odczytana na kanale jest inna dla każdej sesji, nie jest ona przydatna do późniejszego uwierzytelniania. Zaletą jest to, że nie jest potrzebna synchronizacja (inaczej niż w przypadku mechanizmu One-Time-Password, gdzie ważna jest znajomość punktu startowego Pada do użycia). Ponieważ jest niełatwy do wykorzystania przez człowieka, stosuje się go zazwyczaj w uwierzytelnianiu typu machine-to-machine. Mechanizm ten występuje w sieciowych mechanizmach uwierzytelniania routerów ADSL do serwera uwierzytelniającego dostawcy.

Pytania:

W oddzielnym pliku.

Załączniki:

Prezentacja Powerpoint w oddzielnym pliku.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

Co-funded by the Erasmus+ Programme of the European Union

Odnośniki:

Każdy temat ma swoją własną stronę w Wikipedii, czasem już zbyt obszerną na potrzeby tego kursu; książki są zbyt specjalistyczne i przydają się dopiero po bardzo głębokim przeszkoleniu na dany temat.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.