Kryptografia

Cele kształcenia

Wraz z rosnącym wykorzystaniem Internetu (World Wide Web) do komunikacji biznesowej i osobistej, zabezpieczanie poufnych informacji, takich jak dane kart kredytowych, kody PIN, numery kont bankowych i prywatne wiadomości, staje się coraz ważniejsze, choć trudniejsze do osiągnięcia. Dzisiejsze organizacje oparte na informacjach szeroko wykorzystują Internet do handlu elektronicznego, badań rynku, obsługi klienta i wielu innych działań. Bezpieczeństwo danych ma kluczowe znaczenie dla prywatności biznesu online i komunikacji. Kryptografia i systemy kryptograficzne („krypto") pomagają w zabezpieczeniu danych przed przechwyceniem i nadużyciem podczas transmisji online. Ten moduł zapewnia kompleksowe zrozumienie różnych kryptosystemów i algorytmów, jednokierunkowych funkcji mieszania, infrastruktury klucza publicznego (PKI) oraz różnych sposobów, w jakie kryptografia może zapewnić prywatność i bezpieczeństwo komunikacji online. Obejmuje również różne narzędzia służące do szyfrowania wrażliwych danych.

Koncepcje kryptograficzne

Kryptografia umożliwia zabezpieczanie transakcji, komunikacji i innych procesów zachodzących w świecie elektronicznym. Ta sekcja dotyczy kryptografii i powiązanych z nią pojęć, co pozwoli ci zrozumieć zaawansowane tematy omówione w dalszej części tego modułu.

Kryptografia

„Kryptografia" pochodzi od greckich słów kryptos, oznaczających „ukryty, ukryty, zawoalowany, tajny lub tajemniczy" oraz graphia, oznaczającego „pisanie"; tak więc kryptografia jest „sztuką tajnego pisania". Kryptografia to praktyka polegająca na ukrywaniu informacji poprzez konwersję zwykłego tekstu (format czytelny) na tekst zaszyfrowany (format nieczytelny) przy użyciu klucza lub schematu szyfrowania. Jest to proces przekształcania danych w zaszyfrowany kod, który jest szyfrowany i przesyłany przez sieć prywatną lub publiczną. Kryptografia chroni poufne dane, takie jak wiadomości e-mail, sesje czatu, transakcje internetowe, dane osobowe, dane firmowe, aplikacje handlu elektronicznego i wiele innych rodzajów komunikacji. Zaszyfrowane wiadomości można czasami odszyfrować za pomocą kryptoanalizy (łamania kodu), mimo że nowoczesne techniki szyfrowania są praktycznie nie do złamania.

Cele kryptografii

Poufność: pewność, że informacje są dostępne tylko dla osób upoważnionych do dostępu do nich.

Integralność: Wiarygodność danych lub zasobów pod względem zapobiegania niewłaściwym i nieautoryzowanym zmianom.

Uwierzytelnianie: pewność, że komunikacja, dokument lub dane są autentyczne.

Niezaprzeczalność: gwarancja, że nadawca wiadomości nie może później zaprzeczyć, że ją wysłał, a odbiorca nie może zaprzeczyć, że ją otrzymał.

Proces kryptograficzny

Tekst jawny (format czytelny) jest szyfrowany za pomocą algorytmów szyfrowania, takich jak RSA, DES i AES, w wyniku czego powstaje tekst zaszyfrowany (format nieczytelny), który po dotarciu do miejsca docelowego jest odszyfrowywany na czytelny tekst jawny.

Rodzaje kryptografii

Kryptografię dzieli się na dwa typy w zależności od liczby kluczy używanych do szyfrowania i deszyfrowania:

Szyfrowanie symetryczne

Szyfrowanie symetryczne wymaga, aby zarówno nadawca, jak i odbiorca wiadomości posiadali ten sam klucz szyfrujący. Nadawca używa klucza do zaszyfrowania tekstu jawnego i wysyła wynikowy tekst zaszyfrowany do odbiorcy, który używa tego samego klucza (używanego do szyfrowania) do odszyfrowania tekstu zaszyfrowanego na zwykły tekst.

Szyfrowanie symetryczne jest również znane jako kryptografia z tajnym kluczem, ponieważ używa tylko jednego tajnego klucza do szyfrowania i odszyfrowywania danych. Ten rodzaj kryptografii działa dobrze, gdy komunikujesz się tylko z kilkoma osobami. Ponieważ nadawca i odbiorca muszą udostępnić klucz przed wysłaniem jakiejkolwiek wiadomości, technika ta ma ograniczone zastosowanie w Internecie, gdzie osoby, które nie miały wcześniejszego kontaktu, często wymagają bezpiecznego środka komunikacji. Rozwiązaniem tego problemu jest szyfrowanie asymetryczne (kryptografia z kluczem publicznym).

Szyfrowanie asymetryczne

Koncepcja szyfrowania asymetrycznego (znana również jako kryptografia z kluczem publicznym) została wprowadzona w celu rozwiązania problemów związanych z zarządzaniem kluczami. Szyfrowanie asymetryczne obejmuje zarówno klucz publiczny, jak i klucz prywatny. Klucz publiczny jest publicznie dostępny, podczas gdy nadawca utrzymuje klucz prywatny w tajemnicy. System z kluczem asymetrycznym to metoda szyfrowania wykorzystująca parę kluczy składającą się z klucza publicznego dostępnego dla każdego i klucza prywatnego posiadanego tylko przez właściciela klucza, co pomaga zapewnić poufność, integralność, uwierzytelnianie i niezaprzeczalność w zarządzaniu danymi.

Szyfrowanie asymetryczne wykorzystuje następującą sekwencję do wysłania wiadomości:

1. Osoba znajduje w katalogu klucz publiczny osoby, z którą chce się skontaktować.

2. Ten klucz publiczny służy do szyfrowania wiadomości, która jest następnie wysyłana do zamierzonego odbiorcy.

3. Odbiorca używa klucza prywatnego do odszyfrowania wiadomości i odczytuje ją.

Nikt oprócz posiadacza klucza prywatnego nie może odszyfrować wiadomości zaszyfrowanej odpowiednim kluczem publicznym. Zwiększa to bezpieczeństwo informacji, ponieważ cała komunikacja obejmuje tylko klucze publiczne; nadawca wiadomości nigdy nie przesyła ani nie udostępnia kluczy prywatnych. Nadawca musi powiązać klucze publiczne z nazwami użytkowników w bezpieczny sposób, aby zapewnić, że osoby podające się za zamierzonego odbiorcę nie przechwycą informacji. Aby zaspokoić potrzebę uwierzytelnienia, można użyć podpisów cyfrowych.

Mocne i słabe strony metod kryptograficznych

Mocne strony:

Szyfrowanie symetryczne:

Szybszy i łatwiejszy do wdrożenia, ponieważ ten sam klucz jest używany do szyfrowania i deszyfrowania

danych

Wymaga mniejszej mocy obliczeniowej

Może być zaimplementowany w aplikacji zintegrowany układ (ASIC).

Zapobiega powszechnym zagrożeniom bezpieczeństwa wiadomości, ponieważ różne tajne klucze są używane do komunikacji z różnymi stronami

Klucz nie jest związany z danymi przesyłanymi łączem; dlatego nawet jeśli dane zostaną przechwycone, nie ma możliwości ich odszyfrowania

Szyfrowanie asymetryczne:

Wygodny w użyciu, ponieważ nie jest wymagana dystrybucja kluczy do szyfrowania wiadomości

Zwiększone bezpieczeństwo, ponieważ nie trzeba nikomu udostępniać ani przekazywać kluczy prywatnych

Zapewnia podpisy cyfrowe, których nie można odrzucić

Słabe strony :

Szyfrowanie symetryczne:

Brak bezpiecznego kanału wymiany tajnego klucza

Trudności w zarządzaniu i zabezpieczaniu zbyt wielu współdzielonych kluczy, które są generowane w celu komunikowania się z różnymi stronami

Nie daje pewności co do pochodzenia i autentyczności wiadomości, ponieważ ten sam klucz jest używany zarówno przez nadawcę, jak i odbiorcę

Podatny na ataki słownikowe i ataki bruteforce

Szyfrowanie asymetryczne:

Powolne przetwarzanie i wymaga dużej mocy obliczeniowej

Możliwe jest powszechne naruszenie bezpieczeństwa wiadomości (tj

atakujący może przeczytać całe wiadomości, jeśli klucz prywatny jest

skompromitowany)

Otrzymanych wiadomości nie można odszyfrować w przypadku utraty klucza prywatnego

Podatny na ataki typu man-in-the-middle i brute-force

Rządowy dostęp do kluczy (GAK)

Rządowy dostęp do kluczy (GAK) odnosi się do ustawowego obowiązku osób i organizacji do ujawniania swoich kluczy kryptograficznych agencjom rządowym. Oznacza to, że firmy produkujące oprogramowanie przekażą kopie wszystkich kluczy (lub przynajmniej tyle kluczy, aby można było złamać resztę) rządowi. Organy ścigania na całym świecie pozyskują i wykorzystują te klucze kryptograficzne do monitorowania podejrzanej komunikacji i gromadzenia dowodów cyberprzestępstw w interesie bezpieczeństwa narodowego. Rząd obiecuje, że będzie trzymał klucze w bezpieczny sposób i używał ich tylko wtedy, gdy sąd wyda na to nakaz. Dla rządu kwestia ta jest

podobna do możliwości podsłuchiwania telefonów. Agencje rządowe często korzystają z depozytu kluczy, aby zapewnić nieprzerwany dostęp do kluczy. Depozyt kluczy to umowa wymiany kluczy, w ramach której podstawowe klucze kryptograficzne są przechowywane u osoby trzeciej w depozycie. Strona trzecia może używać lub zezwalać innym na używanie kluczy szyfrujących w określonych z góry okolicznościach. Stroną trzecią, w odniesieniu do GAK, jest zasadniczo agencja rządowa, która może wykorzystywać klucze szyfrujące do odszyfrowywania dowodów cyfrowych na podstawie upoważnienia lub nakazu sądowego. Rośnie jednak obawa o prywatność i bezpieczeństwo kluczy i informacji kryptograficznych. Agencje rządowe są odpowiedzialne za ochronę tych kluczy. Takie agencje zwykle używają jednego klucza do ochrony innych kluczy, co nie jest dobrym pomysłem, ponieważ ujawnienie jednego klucza może ujawnić inne klucze. Agencje te nie są świadome, jak poufne są informacje chronione kluczami, co utrudnia ocenę wymaganego stopnia ochrony. W przypadkach, gdy zajęte klucze chronią również inne informacje, do których te agencje nie mają prawa dostępu, konsekwencji ujawnienia klucza nie można ustalić, ponieważ agencje rządowe nie są świadome informacji, które chronią klucze. W takich przypadkach właściciel klucza ponosi odpowiedzialność za konsekwencje ujawnienia klucza. Zanim właściciele przekażą swoje klucze agencjom rządowym, muszą mieć pewność, że agencje rządowe będą chronić te klucze zgodnie z wystarczająco silnymi standardami, aby chronić ich interesy.

Algorytmy szyfrowania

Szyfrowanie to proces przekształcania czytelnego tekstu jawnego w nieczytelny tekst zaszyfrowany przy użyciu zestawu złożonych algorytmów, które przekształcają dane w bloki lub strumienie losowych znaków alfanumerycznych. Ta sekcja dotyczy szyfrów i różnych algorytmów szyfrowania, takich jak DES, AES, RC4, RC5, RC6, DSA, RSA, MD5, SHA itp.

Szyfry

W kryptografii szyfr to algorytm (seria dobrze zdefiniowanych kroków) służący do szyfrowania i deszyfrowania. Szyfrowanie to proces przekształcania zwykłego tekstu w szyfr lub kod; proces odwrotny nazywa się rozszyfrowaniem. Wiadomość zaszyfrowana za pomocą szyfru staje się nieczytelna, chyba że jej odbiorca zna tajny klucz wymagany do jej odszyfrowania. Technologie komunikacyjne (np. Internet, telefony komórkowe) opierają się na szyfrach, aby zapewnić zarówno bezpieczeństwo, jak i prywatność. Algorytmy szyfrujące mogą być otwarte (proces algorytmiczny jest w domenie publicznej, podczas gdy klucz jest wybierany przez użytkownika i jest prywatny) lub zamknięte (proces jest opracowany do użytku w określonych dziedzinach, takich jak wojsko i sam algorytm nie jest w domenie publicznej). Ponadto szyfry mogą być bezpłatne do użytku publicznego lub licencjonowane.

Szyfry dzielą się na dwa główne rodzaje: klasyczne i nowoczesne.

Szyfry klasyczne

Szyfry klasyczne to najbardziej podstawowy typ szyfrów, który operuje na literach alfabetu (A-Z). Szyfry te są na ogół realizowane ręcznie lub za pomocą prostych urządzeń mechanicznych. Ponieważ te szyfry są łatwe do rozszyfrowania, generalnie są niewiarygodne.

Rodzaje szyfrów klasycznych

o Szyfr podstawieniowy: Użytkownik zastępuje jednostki tekstu jawnego tekstem zaszyfrowanym zgodnie ze zwykłym systemem. Jednostki mogą być pojedynczymi literami, parami liter lub ich kombinacjami i tak dalej. Odbiorca wykonuje podstawienie odwrotne, aby rozszyfrować tekst.

Przykłady obejmują szyfr Beale'a, szyfr autokey, szyfr Gronsfelda i szyfr Hilla. Na przykład „HELLO WORLD” można zaszyfrować jako „PSTER HGFST” (tj. H=P, E=S itd.).

o Szyfr transpozycyjny: W tym przypadku litery w tekście jawnym są przestawiane zgodnie z regularnym systemem w celu uzyskania tekstu zaszyfrowanego. Na przykład „KRYPTOGRAFIA” po zaszyfrowaniu zmienia się w „AOYCRGPTYRHP”. Przykłady obejmują szyfr ogrodzenia kolejowego, szyfr trasowy i transpozycję Myszkowskiego.

Nowoczesne szyfry

Nowoczesne szyfry są zaprojektowane tak, aby wytrzymać szeroki zakres ataków. Zapewniają poufność wiadomości, integralność i uwierzytelnienie nadawcy. Użytkownik może obliczyć nowoczesny szyfr za pomocą jednokierunkowej funkcji matematycznej, która jest w stanie rozłożyć na czynniki duże liczby pierwsze.

Rodzaje współczesnych szyfrów

o Na podstawie typu użytego klucza

• Algorytmy z kluczem symetrycznym (kryptografia z kluczem prywatnym): Użyj tego samego klucza do szyfrowania i deszyfrowania.

• Algorytmy z kluczem asymetrycznym (kryptografia z kluczem publicznym): Użyj dwóch różnych kluczy do szyfrowania i deszyfrowania.

o Na podstawie typu danych wejściowych

• Szyfr blokowy: algorytmy deterministyczne działające na bloku (grupie bitów) o stałym rozmiarze z niezmienną transformacją określoną przez klucz symetryczny. Większość współczesnych szyfrów to szyfry blokowe. Są szeroko stosowane do szyfrowania danych masowych. Przykłady obejmują DES, AES, IDEA itp. Gdy rozmiar bloku jest mniejszy niż rozmiar używany przez szyfr, stosuje się dopełnienie w celu uzyskania stałego rozmiaru bloku.

• Szyfr strumieniowy: Szyfry z kluczem symetrycznym to cyfry w postaci zwykłego tekstu połączone ze strumieniem klucza (strumień cyfr szyfru pseudolosowego). Tutaj użytkownik stosuje klucz do każdego bitu, po jednym na raz. Przykłady obejmują RC4, SEAL itp.

Standard szyfrowania danych (DES)

DES to standard szyfrowania danych, który wykorzystuje tajny klucz zarówno do szyfrowania, jak i deszyfrowania (kryptosystem symetryczny). DES używa 64-bitowego tajnego klucza, z którego 56 bitów jest generowanych losowo, a pozostałe 8 bitów służy do wykrywania błędów. Wykorzystuje algorytm szyfrowania danych (DEA), tajny szyfr blokowy wykorzystujący 56-bitowy klucz działający na 64-bitowych blokach. DES to archetypowy szyfr blokowy — algorytm, który pobiera ciąg bitów tekstu jawnego o stałej długości i przekształca go w ciąg bitów tekstu zaszyfrowanego o tej samej długości. Konstrukcja DES pozwala użytkownikom na implementację go sprzętowo i używanie go do szyfrowania pojedynczego użytkownika, na przykład do przechowywania plików na dysku twardym w postaci zaszyfrowanej. DES zapewnia 72 biliardy lub więcej możliwych kluczy szyfrujących i wybiera losowy klucz do zaszyfrowania każdej wiadomości. Ze względu na nieodłączną słabość DES w porównaniu z dzisiejszymi technologiami, niektóre organizacje stosują potrójny DES (3DES), w którym trzykrotnie powtarzają ten proces w celu zwiększenia wydajności, dopóki nie będzie ich stać na aktualizację sprzętu do możliwości AES.

Potrójny standard szyfrowania danych (3DES)

W końcu stało się oczywiste, że DES nie będzie już bezpieczny. Rząd federalny USA rozpoczął konkurs w poszukiwaniu zastępczego algorytmu kryptograficznego. Jednak w międzyczasie 3DES został stworzony jako rozwiązanie tymczasowe. Zasadniczo wykonuje DES trzy razy z trzema różnymi kluczami. 3DES używa „pakietu kluczy", który zawiera trzy klucze DES, Kl, K2 i K3. Każdy klucz jest standardowym 56-bitowym kluczem DES. Następnie wykonuje następujący proces:

Szyfruj DES za pomocą Kl, odszyfruj DES za pomocą K2, szyfruj DES za pomocą K3

Istnieją trzy opcje kluczy. W pierwszej opcji wszystkie trzy klucze są niezależne i różne. W drugim wariancie Kl i K3 są identyczne. W trzeciej opcji wszystkie trzy klucze są takie same; dlatego dosłownie stosujesz ten sam algorytm DES trzy razy z tym samym kluczem. Pierwsza opcja jest najbezpieczniejsza, a trzecia najmniej bezpieczna.

Zaawansowany standard szyfrowania (AES)

Advanced Encryption Standard (AES) to specyfikacja Narodowego Instytutu Standardów i Technologii (NIST) dotycząca szyfrowania danych elektronicznych. Pomaga również szyfrować informacje cyfrowe, takie jak dane telekomunikacyjne, finansowe i rządowe. Amerykańskie agencje rządowe używają go do zabezpieczania wrażliwych, ale niesklasyfikowanych materiałów. AES składa się z algorytmu klucza symetrycznego: zarówno szyfrowanie, jak i deszyfrowanie są wykonywane przy użyciu tego samego klucza. Jest to iterowany szyfr blokowy, który działa poprzez wielokrotne powtarzanie zdefiniowanych kroków. Ma 128-bitowy rozmiar bloku, z kluczami o rozmiarach 128, 192 i 256 bitów odpowiednio dla AES-128, AES-192 i AES-256. Konstrukcja AES sprawia, że jego użycie jest wydajne zarówno w oprogramowaniu, jak i sprzęcie. Działa jednocześnie na wielu warstwach sieci.

Pseudokod AES

Początkowo system kopiuje dane wejściowe szyfru do stanu wewnętrznego, a następnie dodaje początkowy okrągły klucz. System przekształca stan, wykonując iterację funkcji okrągłej w kilku cyklach. Liczba cykli może się różnić w zależności od rozmiaru bloku i długości klucza. Po zakończeniu zaokrąglania system kopiuje stan końcowy do wyjścia szyfrowania.

```
Cipher (byte in [4*Nb], byte out [4*Nb], word w[Nb*(Nr+1)])

begin

byte state[4, Nb]

state = in

AddRoundKey (state, w)

for round = 1 step 1 to Nr-1

SubBytes(state)

ShiftRows(state)

MixColumns(state)

AddRoundKey(state, w+round*Nb)

end for

SubBytes(state)
```

ShiftRows(state)

AddRoundKey(state, w+Nr*Nb)

out = state

end

Algorytmy RC4, RC5 i RC6

Algorytmy szyfrowania symetrycznego opracowane przez RSA Security omówiono poniżej.

RC4

RC4 jest szyfrem strumieniowym z kluczem symetrycznym o zmiennym rozmiarze klucza z operacjami zorientowanymi na bajty i opiera się na wykorzystaniu losowej permutacji. Według niektórych analiz okres szyfru może być większy niż 10 100. Każdy bajt wyjściowy wykorzystuje od 8 do 16 operacji systemowych; w ten sposób szyfr może działać szybko, gdy jest używany w oprogramowaniu. RC4 umożliwia bezpieczną komunikację m.in. dla szyfrowania ruchu (co zabezpiecza strony internetowe) oraz dla stron internetowych korzystających z protokołu SSL.

RC5

RC5 to szybki szyfr blokowy z kluczem symetrycznym zaprojektowany przez Ronalda Rivesta dla RSA Data Security (obecnie RSA Security). Algorytm jest sparametryzowanym algorytmem ze zmiennym rozmiarem bloku, zmiennym rozmiarem klucza i zmienną liczbą rund. Rozmiary bloków mogą wynosić 32, 64 lub 128 bitów. Zakres rund może wynosić od 0 do 255, a rozmiar klucza może wynosić od 0 do 2040 bitów. Ta wbudowana zmienność może zapewnić elastyczność na wszystkich poziomach bezpieczeństwa. Procedury używane w RC5 to rozszerzanie klucza, szyfrowanie i deszyfrowanie. w procedurze rozszerzania klucza tajny klucz dostarczony przez użytkownika jest rozszerzany w celu wypełnienia tabeli kluczy (której rozmiar zależy od liczby rund). RC5 używa tabeli kluczy zarówno do szyfrowania, jak i deszyfrowania. Procedura szyfrowania ma trzy podstawowe operacje: dodawanie liczb całkowitych, bitowe XOR i rotację zmiennych. Intensywne wykorzystanie rotacji zależnej od danych oraz połączenie różnych operacji czyni RC5 bezpiecznym algorytmem szyfrowania.

RC6

RC6 to szyfr blokowy z kluczem symetrycznym wywodzący się z RC5. Jest to sparametryzowany algorytm ze zmiennym rozmiarem bloku, rozmiarem klucza i liczbą rund. Dwie cechy, które odróżniają RC6 od RC5, to mnożenie liczb całkowitych (które służy do zwiększenia rozproszenia, osiąganego w mniejszej liczbie rund przy zwiększonej szybkości szyfru) oraz użycie czterech 4-bitowych rejestrów roboczych zamiast dwóch rejestrów 2-bitowych. RC6 wykorzystuje cztery rejestry 4-bitowe zamiast dwóch rejestrów 2-bitowych, ponieważ rozmiar bloku AES wynosi 128 bitów.

Blowfish

Blowfish to rodzaj algorytmu symetrycznego szyfrowania blokowego, który ma zastąpić algorytmy DES lub IDEA. Używa tego samego tajnego klucza do szyfrowania i odszyfrowywania danych. Algorytm ten dzieli dane na bloki o długości 64 bitów i tworzy klucz o długości od 32 do 448 bitów. Ze względu na dużą szybkość i ogólną wydajność blowfish jest używany w oprogramowaniu, od narzędzi do ochrony hasłem po witryny e-commerce do zabezpieczania płatności. Jest to 16-rundowy szyfr Feistela działający na blokach 64-bitowych. Jednak w przeciwieństwie do DES, jego rozmiar klucza waha się od 32 bitów do 448 bitów. Algorytm ten składa się z dwóch części. Pierwsza część obsługuje rozszerzenie klucza. Druga część faktycznie szyfruje dane. Rozszerzenie klucza odbywa się w kilku krokach.

Pierwszym krokiem jest rozbicie oryginalnego klucza na zestaw podkluczy. W szczególności klucz o długości nie większej niż 448 bitów jest podzielony na 4168 bajtów. Jest tablica P i cztery 32-bitowe S-boxy. Tablica P zawiera 18 32-bitowych podkluczy, podczas gdy każdy S-box zawiera 256 wpisów. Rozszerzenie klucza odbywa się w następujący sposób:

1. Pierwszym krokiem jest zainicjowanie P-array i S-boxów.

2. Następnie wykonaj XOR tablicę P z bitami klucza. Na przykład PI XOR (pierwsze 32 bity klucza), P2 XOR (kolejne 32 bity klucza).

3. Użyj powyższej metody do zaszyfrowania ciągu zerowego.

4. To nowe wyjście to teraz PI i P2.

5. Zaszyfruj nowe PI i P2 za pomocą zmodyfikowanych podkluczy.

6. To nowe wyjście to teraz P3 i P4.

7. Powtórz proces 521 razy, aby obliczyć nowe podklucze dla tablicy P i czterech Sboxów.

Funkcja round dzieli 32-bitowe dane wejściowe na cztery 8-bitowe ćwiartki i używa tych ćwiartek jako danych wejściowych do S-boxów. Wyjścia są dodawane modulo 232 i XORowane w celu wytworzenia ostatecznego 32-bitowego wyjścia.

Twofish

Algorytm Twofish był jednym z pięciu finalistów rządu USA, który miał zastąpić DES, ale nie został wybrany. Został zaprojektowany przez Bruce'a Schneiera, Johna Kelseya, Douga Whitinga, Davida Wagnera, Chrisa Halla i Nielsa Fergusona. TwoFish to 128-bitowy szyfr blokowy. Jest to jeden z najbardziej koncepcyjnie prostych algorytmów, który używa jednego klucza zarówno do szyfrowania, jak i deszyfrowania dla dowolnej długości do 256 bitów. To szyfr Feistela. Działa nie tylko szybko w przypadku procesora lub sprzętu, ale jest również elastyczny w przypadku aplikacji sieciowych.

Ponadto umożliwia różne poziomy kompromisu wydajności w zakresie parametrów, takich jak szybkość szyfrowania, liczba bramek sprzętowych, wykorzystanie pamięci itp. Ta technika włączania różnych implementacji poprawia względną wydajność algorytmu. Każdy użytkownik może zoptymalizować wydajność w oparciu o harmonogram kluczy.

Threefish

Threefish został opracowany w 2008 roku i jest częścią algorytmu Skein. Został zgłoszony do konkursu NIST SHA-3 (funkcja skrótu). Jest to duży, modyfikowalny szyfr blokowy z kluczem symetrycznym, w którym rozmiary bloku i klucza są równe, tj. 256, 512 i 1024. Threefish obejmuje tylko trzy operacje, tj. ARX (dodawanie-obrót-XOR), co sprawia, że kodowanie proste, a wszystkie te operacje działają na słowach 64-bitowych. Bloki Threefish 256, 512 i 1024 obejmują odpowiednio 72, 72 i 80 rund obliczeń, aby osiągnąć ostateczny cel bezpieczeństwa. Ten algorytm nie używa S-boxów do zapobiegania atakom synchronizacji pamięci podręcznej.

Serpent

Podobnie jak Blowfish, Serpent jest szyfrem blokowym z kluczem symetrycznym, który był finalistą konkursu AES. Algorytm ten został zaprojektowany przez Rossa Andersona, Eli Bihama i Larsa Knudsena. Używa 128-bitowego symetrycznego szyfru blokowego z kluczami o długości 128, 192 lub 256 bitów. Może być integrowany z oprogramowaniem lub programami sprzętowymi bez żadnych ograniczeń. Serpent obejmuje 32 rundy operacji obliczeniowych, które obejmują operacje

podstawienia i permutacji na czterech 32-bitowych blokach słów przy użyciu 8-zmiennych S-boxów z 4-bitowym wejściem i 4-bitowym wyjściem. Wszystkie S-boxy pracują równolegle 32 razy. Chociaż Serpent jest jednym z najbezpieczniejszych mechanizmów szyfrowania w konkursach AES, badacze wybrali Rijndael zamiast Serpent ze względu na jego umiarkowaną szybkość szyfrowania (ze względu na liczbę używanych rund) i złożoność. Wąż minimalizuje korelację między zakodowanymi obrazami lub tekstami jawnymi w większym stopniu niż Twofish i Rijndael. Dlatego Rijndael jest wyróżniającym się konkurentem AES i jest obecnie używany jako AES.

TEA

Mały algorytm szyfrowania (TEA) został stworzony przez Davida Wheelera i Rogera Needhama i został publicznie zaprezentowany po raz pierwszy w 1994 roku. Jest to prosty algorytm, łatwy do zaimplementowania w kodzie. Jest to szyfr Feistela, który wykorzystuje 64 rundy (zauważ, że jest to sugestia; można go zaimplementować z mniejszą lub większą liczbą rund). Liczba rund powinna być parzysta, ponieważ są one realizowane w parach zwanych cyklami.

TEA używa 128-bitowego klucza działającego na 64-bitowym bloku. Wykorzystuje również stałą zdefiniowaną jako 232/złoty podział. Ta stała jest określana jako delta, aw każdej rundzie używana jest wielokrotność delty. Klucz 128-bitowy jest podzielony na cztery różne 32-bitowe podklucze oznaczone K[0], K[l], K[2] i K[3j. Zamiast używać operacji XOR, TEA używa dodawania i odejmowania, ale z mod 232. Blok jest podzielony na dwie połowy, R i LR. R jest przetwarzany przez funkcję rundy. Funkcja zaokrąglająca bierze połowę R i wykonuje przesunięcie w lewo o 4. Następnie wynik tej operacji jest dodawany do K[0]. Następnie wynik tej operacji jest dodawany do delty (przypomnijmy, że delta jest aktualną wielokrotnością 232/złotego podziału). Wynik tej operacji jest następnie przesuwany w prawo o 5 i dodawany do K[l]. To jest funkcja okrągła. Podobnie jak w przypadku wszystkich szyfrów Feistela, wynik funkcji rundy jest XORowany z L, a L i R są następnie zamieniane na następną rundę.

CAST-128

CAST-128, zwany także CAST5, to szyfr blokowy z kluczem symetrycznym, mający klasyczną 12- lub 16-okrągłą sieć Feistela o rozmiarze bloku 64 bitów. CAST-128 używa klucza o długości od 40 do 128 bitów w przyrostach 8-bitowych. Komponenty CAST-128 obejmują duże 8x32-bitowe S-boxy (SI, S2, S3, S4) oparte na funkcjach wygiętych, modułowym dodawaniu i odejmowaniu, obracaniu zależnym od klucza i operacjach XOR. CAST-128 wykorzystuje klucz maskujący (Kmi) i klucz obrotowy (Kn) do wykonywania swoich funkcji. Funkcja round składa się z trzech naprzemiennych typów do wykonywania operacji dodawania, odejmowania lub operacji XOR na różnych etapach. Używany jako domyślny szyfr w GPG (GNU Privacy Guard) i PGP (Pretty Good Privacy). CAST-256 jest rozszerzeniem CAST-128, które wykorzystuje tę samą procedurę projektowania. CAST-256 ma 128-bitowy rozmiar bloku i wykorzystuje rozmiary kluczy od 128 do 256 bitów. Ponadto wykorzystuje kryptoanalizę zerowej korelacji, która może przerwać 28 rund z czasem = 2246,9 i danymi = 298,8.

Szyfr blokowy GOST

Szyfr blokowy GOST (Government Standard), zwany także Magma, jest szyfrem blokowym z kluczem symetrycznym, mającym 32-rundową sieć Feistela działającą na 64-bitowych blokach z 256-bitowym kluczem. Składa się z S-boxa, który może być utrzymywany w tajemnicy i zawiera około 354 bitów tajnych informacji. GOST jest prostym algorytmem szyfrowania, w którym dodaje się 32-bitowy podklucz modulo 232 funkcji okrągłej i umieszcza w warstwie S-boxów, a operacja obrotu w lewo z przesunięciem jest używana do przesunięcia 11 bitów, zapewniając w ten sposób wyjście funkcji okrągłej. Planowanie klucza w szyfrze blokowym GOST odbywa się poprzez rozbicie 256-bitowego klucza na osiem 32-bitowych podkluczy, gdzie każdy podklucz jest używany cztery razy. W tym

algorytmie słowa kluczowe są używane w kolejności przez pierwsze 24 rundy iw odwrotnej kolejności przez ostatnie 8 rund. Kuznyechik to najnowsze rozszerzenie GOST, które wykorzystuje 128-bitowe bloki.

Camellia

Camellia to szyfr blokowy z kluczem symetrycznym mający albo 18 rund (dla kluczy 128-bitowych), albo 24 rundy (dla kluczy 256-bitowych). Jest to szyfr Feistela o rozmiarze bloku 128 bitów i rozmiarze klucza 128, 192 i 256 bitów. Camellia wykorzystuje cztery 8x8-bitowe S-boxy, które wykonują transformacje afiniczne i operacje logiczne. Funkcja FL warstwy transformacji logicznej lub jej odwrotność jest stosowana co sześć rund. Camellia wykorzystuje kluczową technikę wybielania dla zwiększenia bezpieczeństwa. Camellia jest częścią protokołu Transport Layer Security (TLS), który służy do zapewnienia bezpiecznej komunikacji. Camellia nie może być brutalnie wymuszona nawet przy użyciu najnowszej technologii, chociaż używa mniejszego klucza o długości 128 bitów, dzięki czemu jest bezpiecznym szyfrem. Ponadto Camellia oferuje wysokie bezpieczeństwo, a jej możliwości przetwarzania są równoważne z AES lub Rijndael.

DSA i powiązane schematy podpisów

Digital Signature Algorithm (DSA) to federalny standard przetwarzania informacji dla podpisów cyfrowych. NIST zaproponował DSA do użytku w standardzie Digital Signature Standard (DSS), przyjętym jako FIPS 186. DSA pomaga w generowaniu i weryfikacji podpisów cyfrowych dla wrażliwych i niesklasyfikowanych aplikacji. Tworzy 320-bitowy podpis cyfrowy z zabezpieczeniami 512-1024-bitowymi. Podpis cyfrowy to schemat matematyczny używany do uwierzytelniania wiadomości cyfrowych. Obliczanie podpisu cyfrowego wykorzystuje zestaw reguł (tj. DSA) i zestaw parametrów, dzięki którym użytkownik może zweryfikować tożsamość sygnatariusza i integralność danych.

Procesy zaangażowane w DSA:

Proces generowania podpisu: Klucz prywatny służy do ustalenia, kto go podpisał.

• Proces weryfikacji podpisu: Klucz publiczny służy do weryfikacji autentyczności danego podpisu cyfrowego.

DSA jest kryptosystemem z kluczem publicznym, ponieważ wymaga użycia zarówno kluczy prywatnych, jak i publicznych.

Korzyści z DSA:

Mniejsze ryzyko fałszerstwa w porównaniu z pisemnym podpisem

Szybki i łatwy sposób transakcji biznesowych

Problem fałszywej waluty można znacznie złagodzić

Algorytm DSA:

Każdy podmiot A wykonuje następujące czynności:

1. Wybierz liczbę pierwszą q taką, że $2^{159} < g < 2^{160}$

2. Wybierz t takie, że $0 \leq t \leq 8$, i wybierz liczbę pierwszą p gdzie $2^{511+64}t<p< 2^{512+64t}$, z właściwością, że q dzieli (p-1)

3. Wybierz generator a unikalnej grupy cyklicznej rzędu q w z^*_p, wybierając element $g \in z^*_p$, a następnie obliczając $\alpha = g^{(p-1'/q)} \bmod p$ aż do $\alpha \neq 1$

4. Wybierz losową liczbę całkowitą d taką, że $1 \leq d \leq q-1$

5. Oblicz $y = \alpha d \bmod p$

6. Kluczem publicznym A jest (p, q, α , y); Kluczem prywatnym A jest d.

Aby podpisać wiadomość m, A wykonuje następujące czynności:

1. Wybierz losową tajną liczbę całkowitą k,0<k<q.

2. Oblicz $r=(\alpha^k \bmod p) \bmod q$

3. Oblicz k^{-1} od q

4. Oblicz $s=k^{-1}\{h(m) + dr\} \bmod q$, gdzie h to bezpieczny algorytm mieszania

5. Sygnatura A dla m to para (r, s)

Aby zweryfikować podpis A (r, s) na m, B powinien wykonać następujące czynności:

1. Uzyskaj autentyczny klucz publiczny A (p, q, a, y)

2. Sprawdź, czy 0<r<q i 0<s<q; jeśli nie, odrzuć podpis

3. Oblicz $w = s^{-1} \bmod q$ and h(m)

4. Oblicz $u_1 = w.h(m) \bmod q$ i $u_2 = rw \bmod q$

5. Oblicz $v = (\alpha^{u1} y^{u2} \bmod p) \bmod q$

6. Zaakceptuj podpis wtedy i tylko wtedy, gdy v=r

Rivest Shamir Adleman (RPA)

Ron Rivest, Adi Shamir i Leonard Adleman opracowali RSA, kryptosystem klucza publicznego do szyfrowania i uwierzytelniania w Internecie. RSA wykorzystuje modułową arytmetykę i elementarne teorie liczb do wykonywania obliczeń przy użyciu dwóch dużych liczb pierwszych. System RSA jest szeroko stosowany w różnych produktach, platformach i branżach. Jest to de facto jeden ze standardów szyfrowania. Firmy takie jak Microsoft, Apple, Sun i Novell wbudowują algorytmy RSA w swoje systemy operacyjne. RSA można również znaleźć w telefonach zabezpieczonych sprzętowo, kartach sieciowych Ethernet i kartach inteligentnych.

RSA działa w następujący sposób:

1. Bierze się dwie duże liczby pierwsze (a i b) i określa się ich iloczyn (c = ab, gdzie „c" nazywa się modułem).

2. RSA wybiera liczbę „e", która jest mniejsza niż „c" i względnie pierwsza względem (a-1)(b-1). Zatem e i (a-1)(b-1) nie mają wspólnego czynnika poza 1.

3. Ponadto RSA wybiera liczbę „f" taką, że (ef -1) jest podzielna przez (a-1)(b-1).

4. Wartości „e" i „f" są odpowiednio wykładnikami publicznymi i prywatnymi.

5. Kluczem publicznym jest para (c, e); kluczem prywatnym jest para (c, f).

6. Trudno jest uzyskać klucz prywatny (c,f) z klucza publicznego (c,e). Jeśli jednak ktoś może rozłożyć „c" na „a" i „b", wówczas osoba ta może odszyfrować klucz prywatny (c, f).

Bezpieczeństwo systemu RSA opiera się na założeniu, że taki faktoring jest trudny do przeprowadzenia, dzięki czemu technika kryptograficzna jest bezpieczna.

Zilustrowano przykład, w jaki sposób kryptografia wykorzystuje algorytmy RSA w praktycznej wymianie według następującej kolejności:

Nadawca wiadomości szyfruje ją losowo wybranym kluczem symetrycznym DES. DES (Data Encryption Standard) to stosunkowo niepewny system klucza symetrycznego wykorzystujący szyfrowanie 64-bitowe (56 bitów dla rozmiaru klucza, 8 bitów dla cyklicznej kontroli redundancji) do szyfrowania danych.

Nadawca wyszuka następnie klucz publiczny odbiorcy i użyje go do zaszyfrowania klucza DES przy użyciu systemu RSA.

Nadawca przesyła do odbiorcy cyfrową kopertę RSA, składającą się z wiadomości zaszyfrowanej DES i klucza DES zaszyfrowanego RSA.

Odbiorca odszyfruje klucz DES, a następnie użyje klucza DES do odszyfrowania samej wiadomości.

System ten łączy w sobie dużą szybkość DES z wygodą zarządzania kluczami systemu RSA.

Schemat podpisu RSA

Kryptografia wykorzystuje RSA do szyfrowania klucza publicznego i podpisu cyfrowego (do podpisania wiadomości i jej weryfikacji). Schemat podpisu RSA jest pierwszą techniką używaną do generowania podpisów cyfrowych. Jest to deterministyczny schemat podpisu cyfrowego, który zapewnia odzyskiwanie wiadomości z samego podpisu, co czyni go najbardziej praktyczną i wszechstronną dostępną techniką. RSA obejmuje zarówno klucz publiczny, jak i klucz prywatny. Klucz publiczny, jak sama nazwa wskazuje, może być używany przez każdego do szyfrowania wiadomości. Wiadomości, które użytkownik szyfruje kluczem publicznym, wymagają klucza prywatnego do odszyfrowania. Weźmy pod uwagę, że Jan szyfruje swój dokument M przy użyciu swojego klucza prywatnego SA, tworząc w ten sposób podpis Sjohn(M). John wysyła M wraz z podpisem Sjohn(M) do Alicji. Alice odszyfrowuje dokument przy użyciu klucza publicznego Johna, weryfikując w ten sposób podpis Johna.

Generowanie klucza RSA

Procedura generowania klucza RSA jest wspólna dla wszystkich schematów podpisu opartych na RSA. Aby wygenerować parę kluczy RSA, tj. zarówno klucz publiczny RSA, jak i odpowiadający mu klucz prywatny, każda jednostka A powinna wykonać następujące czynności:

Wygeneruj dowolnie dwie duże, różne liczby pierwsze p i q, każda o mniej więcej tej samej długości bitowej

Oblicz $n = pq$ i $\phi = (p-1)(q-1)$

Wybierz losową liczbę całkowitą $e, 1 < e < \phi$, taką, że $\gcd(e, \phi) = 1$

NWD = największy wspólny dzielnik

Użyj rozszerzonego algorytmu Euklidesa, aby obliczyć unikalną liczbę całkowitą $d, 1 < d < \phi$, taką, że $ed \equiv 1 \pmod{\phi}$

Kluczem publicznym A jest (n, e); Kluczem prywatnym A jest d

Zniszcz p i q na koniec generowania klucza

Algorytm RSA generuje i weryfikuje podpis RSA w następujący sposób:

Jednostka A podpisuje komunikat m∈M. Każdy podmiot B może zweryfikować podpis A i odzyskać wiadomość m z podpisu.

1. Generowanie podpisu

Aby podpisać wiadomość m, podmiot A powinien wykonać następujące czynności:

o Oblicz $\tilde{m} = R(m)$, liczbę całkowitą z zakresu [0, n—1]

o Oblicz $s = \tilde{m}^d \bmod n$

o Formularz podpisu A to s

2. Weryfikacja podpisu

Aby zweryfikować podpis A i odzyskać wiadomość m, B powinien wykonać następujące czynności:

o Uzyskaj autentyczny klucz publiczny A (n, e)

o Oblicz $\tilde{m} = s^e \bmod n$

o Sprawdź, czy $\tilde{m} \in M_R$; jeśli nie, odrzuć podpis

o Odzyskaj $m = R^{-1}(\tilde{m})$

Przykład algorytmu RSA

Matematyka leżąca u podstaw szyfrowania klucza publicznego RSA została opisana poniżej:

1. Znajdź P i Q, dwie duże (np. 1024-bitowe) liczby pierwsze.

2. Wybierz E takie, że E jest większe niż 1, E jest mniejsze niż PQ, a E i (P-1)(Q-1) są względnie pierwsze, co oznacza, że nie mają wspólnych czynników pierwszych. E nie musi być liczbą pierwszą, ale musi być liczbą nieparzystą. (P-1)(K-1) nie może być liczbą pierwszą, ponieważ jest liczbą parzystą.

3. Oblicz D tak, że (DE - 1) jest równo podzielne przez (P-1)(Q-1). Matematycy zapisują to jako DE =1(mod (P-1)(Q-1)) i nazywają D multiplikatywną odwrotnością E. Jest to łatwe do zrobienia — po prostu znajdź liczbę całkowitą X, która powoduje, że D = (X(P -1)(Q-1) +1)/E jako liczbę całkowitą, a następnie użyj tej wartości D.

4. Funkcja szyfrująca to C = (TAE) mod PQ, gdzie C to tekst zaszyfrowany (dodatnia liczba całkowita), T to tekst jawny (dodatnia liczba całkowita), a ^ oznacza potęgowanie. Podczas szyfrowania wiadomości T musi być mniejsze niż moduł PQ.

5. Funkcja deszyfrująca to T = (CAD) mod PQ, gdzie C to tekst zaszyfrowany (dodatnia liczba całkowita), T to tekst jawny (dodatnia liczba całkowita), a ^ oznacza potęgowanie.

Twój klucz publiczny to para (PQ, E). Twój klucz prywatny to cyfra D (nie ujawniaj go nikomu). Produkt PQ to moduł. E jest publicznym wykładnikiem. D jest tajnym wykładnikiem. Możesz swobodnie publikować swój klucz publiczny, ponieważ nie ma znanych łatwych metod obliczania D, P lub tylko Qgiven (PQ, E) (Twój klucz publiczny).

Poniżej podano przykład algorytmu RSA:

P = 61<= pierwsza liczba pierwsza (zniszcz to po obliczeniu E i D)

Q= 53<= druga liczba pierwsza (zniszcz to po obliczeniu E i D)

PQ = 3233<= moduł (podaj to innym)

E = 17<= publiczny wykładnik (podaj to innym)

D = 2753<= prywatny wykładnik (zachowaj to w tajemnicy)

Twój klucz publiczny to (E,PQ)

Twój klucz prywatny to D

Funkcja szyfrująca to:

encrypt(T) = (T^E) mod PQ = (T^17) mod 3233

Funkcja deszyfrująca to:

decrypt(C) = (C^D) mod PQ = (C^2753) mod 3233

Aby zaszyfrować wartość 123 w postaci zwykłego tekstu, wykonaj następujące czynności:

encrypt(123) = (123^17) mod 3233

= 337587917446653715596592958817679803 mod 3233

= 855

Aby odszyfrować wartość tekstu zaszyfrowanego 855, wykonaj następujące czynności:

decrypt (855) = (855*2753) mod 3233

= 123

Jednym ze sposobów obliczenia wartości 855A2753 mod 3233 jest następujący:

Rozważ te uprawnienia 855:

855^1 = 855 (mod 3233)

855^2 = 367 (mod 3233)

855^4 = 367A2 (mod 3233) = 2136 (mod 3233)

855^8 = 2136A2 (mod 3233) = 733 (mod 3233)

855^16 = 733A2 (mod 3233) = 611 (mod 3233)

855^32 = 611A2 (mod 3233) = 1526 (mod 3233)

855^A64 = 1526A2 (mod 3233) = 916 (mod 3233)

855^128 = 916A2 (mod 3233) = 1709 (mod 3233)

855^256 = 1709A2 (mod 3233) = 1282 (mod 3233)

855^A512 = 1282A2 (mod 3233) = 1160 (mod 3233)

855^A1024 = 1160A2 (mod. 3233) = 672 (mod. 3233)

855^2048 = 672A2 (mod 3233) = 2197 (mod 3233)

Biorąc pod uwagę powyższe, wiemy, co następuje:

855^2753 (mod 3233)

= 855^(1+ 64 + 128 + 512 + 2048) (mod 3233)

= 855^1* 855^64 * 855^128 * 855^512 * 855^2048 (model 3233)

= 855 * 916 * 1709 * 1160 * 2197 (mod. 3233)

= 794 * 1709 * 1160 * 2197 (mod 3233)

= 2319 * 1160 * 2197 (mod 3233)

= 184 * 2197 (mod 3233)

= 123 (mod 3233)

= 123

Diffie-Hellman

Jest to protokół kryptograficzny, który umożliwia dwóm stronom ustanowienie wspólnego klucza przez niezabezpieczony kanał. Został opracowany i opublikowany przez Whitfielda Diffie i Martina Heilmana w 1976 roku. W rzeczywistości został niezależnie opracowany kilka lat wcześniej przez Malcolma J. Williamsona z Brytyjskiej Służbie Wywiadowczej, ale wówczas była tajna.

Algorytm Diffiego-Hellmana

System ma dwa parametry zwane p i g

Parametr p jest liczbą pierwszą

Parametr g (zwykle nazywany generatorem) jest liczbą całkowitą mniejszą od p i ma następującą właściwość: dla każdej liczby n między 1 a p-1 (oba włącznie) istnieje taka potęga k g, że n = gkmod p

Wiele podręczników kryptografii wykorzystuje fikcyjne postacie „Alicja” i „Bob” do zilustrowania kryptografii; zrobimy to samo tutaj:

Alicja generuje losową wartość prywatną a, a Bob generuje losową wartość prywatną b.

Zarówno a, jak i b są losowane ze zbioru liczb całkowitych

Wyprowadzają swoje wartości publiczne za pomocą parametrów p i g oraz ich wartości prywatnych. Wartość publiczna Alicji to g^amod p, a wartość publiczna Boba to g^bmod p.

Wymieniają swoje wartości publiczne

Alicja oblicza $g^{ab} = (g^b)^a$ mod p, a Bob oblicza $g^{ba} = (g^a)^b$ mod p

Ponieważ $g^{ab} = g^{ba}$ = k, Alicja i Bob mają teraz wspólny tajny klucz k

Algorytm Diffiego-Hellmana nie zapewnia uwierzytelniania wymiany kluczy i jest podatny na wiele ataków kryptograficznych. Niemniej jednak jest podstawą wielu mechanizmów uwierzytelniania; na przykład zapewnia poufność przekazywania w efemerycznych trybach protokołu TLS w zależności od specyfikacji szyfru.

YAK

YAK to oparty na kluczu publicznym protokół Authenticated Key Exchange (AKE). Uwierzytelnianie YAK opiera się na parach kluczy publicznych i potrzebuje PKI do dystrybucji autentycznych kluczy publicznych. YAK to wariant dwuprzebiegowego protokołu Hashed Menezes-Qu-Vanstone (HMQV) wykorzystujący dowody zerowej wiedzy (ZKP) w celu udowodnienia znajomości efemerycznych tajnych kluczy obu stron. W protokole YAK brakuje wspólnej kontroli klucza i doskonałych atrybutów poufności przekazywania. Implementacja protokołu YAK między dwiema stronami Alice i Bob jest opisana w następujący sposób:

1. Alicja wybiera liczbę losową x taką, że $x \in_R [0, q - 1]$, oblicza $X = g^x$ i generuje ZKP z x, oznaczone przez KP{x}. Alicja wysyła X i KP{x} do Boba.

2. Bob wybiera losową liczbę y taką, że $y \in_R [0, q - 1]$, oblicza $Y = g^v$ i generuje ZKP z y, oznaczone przez KP{y}. Bob wysyła Y i KP{y} do Alicji.

3. Alicja weryfikuje odebrany KP{x} i po weryfikacji oblicza klucz sesji jako $k = H ((Y.PK_B)^{x + a})$, gdzie H jest funkcją mieszającą.

4. Bob weryfikuje otrzymane KP{y} i po weryfikacji oblicza klucz sesyjny jako $k = H ((X.PK_A)^{v + b})$.

5. Uwierzytelniają się nawzajem i obaj uzyskują ten sam klucz sesji $k = H (g^{(x + a)(y + b)})$

Protokół YAK może osiągnąć następujące cele:

• Bezpieczeństwo klucza prywatnego

• Pełna poufność przekazywana

• Bezpieczeństwo klucza sesji

Funkcje skrótu wiadomości (jednokierunkowy skrót).

Funkcje skrótu obliczają unikalną reprezentację łańcucha bitów o stałym rozmiarze, zwaną skrótem wiadomości, dowolnego dowolnego bloku informacji. Funkcje skrótu wiadomości destylują informacje zawarte w pliku (małym lub dużym) do pojedynczej liczby o stałej długości, zazwyczaj od 128 do 256 bitów. Jeśli dowolny bit wejścia funkcji zostanie zmieniony, każdy bit wyjściowy ma 50% szans na zmianę. Biorąc pod uwagę plik wejściowy i odpowiadający mu skrót wiadomości, znalezienie innego pliku z taką samą wartością skrótu wiadomości powinno być prawie niemożliwe, ponieważ posiadanie dwóch plików z tą samą wartością skrótu wiadomości jest obliczeniowo niewykonalne. Funkcje skrótu wiadomości są również nazywane jednokierunkowymi funkcjami mieszającymi, ponieważ generują wartości które są prawie niemożliwe do odwrócenia, odporne na ataki, w większości unikalne i szeroko rozpowszechnione. Same algorytmy skrótu wiadomości nie uczestniczą w operacjach szyfrowania i deszyfrowania. Umożliwiają tworzenie podpisów cyfrowych i kodów uwierzytelniających wiadomości (MAC), a także wyprowadzanie kluczy szyfrujących z haseł. Główną rolą kryptograficznej funkcji skrótu jest zapewnienie integralności w zarządzaniu dokumentami. Kryptograficzne funkcje skrótu są integralną częścią podpisów cyfrowych. Są stosunkowo szybsze niż algorytmy podpisu cyfrowego; stąd ich cechą charakterystyczną jest obliczanie sygnatury wartości skrótu dokumentu, która jest mniejsza niż dokument. Ponadto streszczenia pomagają ukryć treść lub źródło dokumentu.

Szeroko stosowane funkcje skrótu wiadomości obejmują następujące algorytmy:

MD5

SHA

Uwaga: Skróty wiadomości są również nazywane jednokierunkowymi funkcjami mieszającymi, ponieważ nie można ich odwrócić.

Funkcja skrótu wiadomości: MD5 i MD6

MD2, MD4, MD5 i MD6 to algorytmy skrótu wiadomości używane w aplikacjach do podpisu cyfrowego w celu bezpiecznego skompresowania dokumentu, zanim system podpisze go kluczem prywatnym. Algorytmy mogą mieć zmienną długość, ale wynikowy skrót wiadomości ma zawsze rozmiar 128 bitów. Struktury wszystkich trzech algorytmów (MD2, MD4 i MD5) wydają się podobne, chociaż konstrukcja MD2 różni się znacznie od konstrukcji MD4 i MD5. MD2 obsługuje maszyny 8-bitowe, podczas gdy MD4 i MD5 obsługują maszyny 32-bitowe. Algorytm wypełnia wiadomość dodatkowymi bitami, aby liczba bitów była podzielna przez 512. Dodatkowe bity mogą zawierać 64-bitowy komunikat binarny. Ataki na wersje MD4 stają się coraz bardziej skuteczne. Badania wykazały, w jaki sposób osoba atakująca przeprowadza ataki kolizyjne na pełną wersję MD4 w ciągu minuty na typowym komputerze PC. MD5 jest nieco bezpieczniejszy, ale wolniejszy niż MD4. Jednak zarówno rozmiar skrótu wiadomości, jak i wymagania dotyczące wypełnienia pozostają takie same. MD5 to szeroko stosowana kryptograficzna funkcja skrótu, która pobiera wiadomość o dowolnej długości jako dane wejściowe i wysyła 128-bitowy (16-bajtowy) odcisk palca lub skrót wiadomości wejściowej. MD5 może być używany w wielu różnych aplikacjach kryptograficznych i jest przydatny w aplikacjach podpisów cyfrowych, sprawdzaniu integralności plików i przechowywaniu haseł. Jednak MD5 nie jest odporny na kolizje; dlatego lepiej jest używać najnowszych algorytmów, takich jak MD6, SHA-2 i SHA-3. MD6 wykorzystuje strukturę podobną do drzewa Merkle, aby umożliwić równoległe obliczanie skrótów na dużą skalę dla bardzo długich danych wejściowych. Jest odporny na ataki kryptoanalizy różnicowej. Aby obliczyć efektywność funkcji skrótu, sprawdź wynik wygenerowany, gdy algorytm losuje dowolny komunikat wejściowy.

Poniżej przedstawiono przykłady minimalnie różnych skrótów wiadomości:

echo "There is CHF1500 in the blue bo" | md5sum

e41a323bdf20eadafd3f0e4f72055d36

echo "There is CHF1500 in the blue box" | md5sum

7a0da864a41fd0200ae0ae97afd3279d

echo "There is CHF1500 in the blue box." | md5sum

2dblff7a70245309e9f2165c6c34999d

Nawet minimalnie różne teksty dają radykalnie różne kody MD5.

Funkcja skrótu wiadomości: Bezpieczny algorytm mieszania (SHA)

NIST opracował algorytm Secure Hash Algorithm (SHA), określony w standardzie Secure Hash Standard (SHS) i opublikowany jako federalny standard przetwarzania informacji (FIPS PUB 180). Generuje kryptograficznie bezpieczny jednokierunkowy skrót. Rivest opracował SHA, który jest podobny do rodziny funkcji skrótu algorytmu skrótu wiadomości. Jest nieco wolniejszy niż MD5, ale jego większy skrót wiadomości sprawia, że jest bardziej bezpieczny przed kolizjami typu brute-force i atakami inwersyjnymi. Szyfrowanie SHA to seria pięciu różnych funkcji kryptograficznych, które obecnie mają trzy generacje: SHA-1, SHA-2 i SHA-3.

SHA-0: Retronim zastosowany do oryginalnej wersji 160-bitowej funkcji skrótu opublikowanej w 1993 roku pod nazwą SHA, która została wycofana z handlu z powodu nieujawnionej w niej „istotnej wady". Został on zastąpiony nieco poprawioną wersją, a mianowicie SHA-l.

SHA-1: Jest to 160-bitowa funkcja skrótu, która przypomina poprzedni algorytm MD5 opracowany przez Rona Rivesta. Tworzy 160-bitowy skrót z wiadomości o maksymalnej długości (264 - 1) bitów. Został zaprojektowany przez Agencję Bezpieczeństwa Narodowego (NSA) jako część algorytmu podpisu cyfrowego (DSA). Jest najczęściej używany w protokołach bezpieczeństwa, takich jak PGP, TLS, SSH i SSL. Od 2010 r. SHA-lis nie jest już zatwierdzony do użytku kryptograficznego ze względu na jego słabości kryptograficzne.

SHA-2: SHA2 to rodzina dwóch podobnych funkcji skrótu o różnych rozmiarach bloków, a mianowicie SHA-256, która używa słów 32-bitowych, oraz SHA-512, która używa słów 64-bitowych. Skrócone wersje każdego standardu to SHA-224 i SHA-384.

SHA-3: SHA-3 wykorzystuje konstrukcję gąbki, w której bloki komunikatów są poddawane operacji XOR w początkowych bitach stanu, który następnie algorytm odwracalnie przetwarza. Obsługuje te same długości skrótu co SHA-2, ale znacznie różni się swoją wewnętrzną strukturą od reszty rodziny SHA.

RIPEMD-160

RACE Integrity Primitives Evaluation Message Digest (RIPEMD) to 160-bitowy algorytm mieszający opracowany przez Hansa Dobbertina, Antoona Bosselaersa i Barta Preneela. Istnieją 128-, 256- i 320-bitowe wersje tego algorytmu, zwane odpowiednio RIPEMD-128, RIPEMD-256 i RiPEMD-320. Algorytmy te zastępują oryginalny RIPEMD, w którym wykryto problem z kolizją. Nie przestrzegają żadnych standardowych zasad ani wytycznych dotyczących bezpieczeństwa. RIPEMD-160 to bezpieczniejsza wersja algorytmu RIPEMED. W tym algorytmie funkcja kompresji składa się z 80 etapów, tj. 5 bloków, z których każdy wykonuje się 16 razy. Ten proces powtarza się dwukrotnie, łącząc wyniki na dole za pomocą dodawania modulo 32.

HMAC

Kod uwierzytelniania wiadomości oparty na skrótach (HMAC) to rodzaj kodu uwierzytelniania wiadomości (MAC), który używa klucza kryptograficznego wraz z funkcją skrótu kryptograficznego. Jest szeroko stosowany do weryfikacji integralności danych i uwierzytelniania wiadomości. Algorytm ten zawiera wbudowaną funkcję skrótu, taką jak SHA-1 lub MD5. Siła HMAC zależy od wbudowanej funkcji skrótu, rozmiaru klucza i rozmiaru wyjścia skrótu. HMAC obejmuje dwa etapy obliczania skrótu. Klucz wejściowy jest przetwarzany w celu wytworzenia dwóch kluczy, a mianowicie klucza wewnętrznego i klucza zewnętrznego. Pierwszy etap algorytmu wprowadza wewnętrzny klucz i komunikat w celu utworzenia wewnętrznego skrótu. Drugi etap algorytmu wprowadza dane wyjściowe z pierwszego etapu i klucza zewnętrznego oraz generuje końcowy kod HMAC. Ponieważ HMAC dwukrotnie wykonuje podstawową funkcję skrótu, zapewnia ochronę przed atakami o różnej długości. Rozmiar klucza i dane wyjściowe zależą od wbudowanej funkcji skrótu; np. 128 lub 160 bitów odpowiednio w przypadku MD5 lub SHA-1.

CHAP

Protokół uwierzytelniania Challenge-Handshake (CHAP) to mechanizm uwierzytelniania używany przez serwery protokołu Point-to-Point Protocol (PPP) do uwierzytelniania lub sprawdzania tożsamości zdalnych klientów lub hostów sieciowych. Jest bezpieczniejszy i skuteczniejszy w porównaniu z procedurą uwierzytelniania hasła (PAP), ponieważ regularnie weryfikuje tożsamość klienta za pomocą trójstronnego uzgadniania i zapewnia ochronę przed atakami powtórkowymi.

EAP

Protokół Extensible Authentication Protocol (EAP) to protokół uwierzytelniania, który został pierwotnie zaprojektowany dla połączeń punkt-punkt. Jest używany jako alternatywa dla protokołów uwierzytelniania CHAP i PAP, ponieważ jest bezpieczniejszy i obsługuje różne mechanizmy uwierzytelniania, takie jak hasła, inteligentne tokeny, hasła jednorazowe (OTP), bezpieczny dowód osobisty, certyfikaty cyfrowe i klucz publiczny mechanizmy szyfrowania. Po wybraniu mechanizmu uwierzytelniania EAP nawiązywana jest sesja i następuje wymiana komunikatów pomiędzy klientem a serwerem uwierzytelniającym. Sesja składa się z żądań i odpowiedzi dotyczących informacji uwierzytelniających. Długość i szczegóły sesji uwierzytelniania są określane przez zastosowany mechanizm uwierzytelniania EAP.

GOST — funkcja skrótu

Ten algorytm mieszania został początkowo zdefiniowany w rosyjskiej normie krajowej GOST R 34.11-94 „Technologia informacyjna — bezpieczeństwo informacji kryptograficznych — funkcja skrótu". Tworzy wyjście o stałej długości 256 bitów. Wiadomość wejściowa jest dzielona na fragmenty 256-bitowych bloków. Jeśli blok ma mniej niż 256 bitów, wiadomość jest uzupełniana przez dodanie do niej tylu zer, ile potrzeba, aby długość wiadomości wynosiła 256 bitów. Pozostałe bity są wypełnione 256-bitową całkowitą sumą arytmetyczną wszystkich wcześniej zahaszowanych bloków. Następnie tworzona jest 256-bitowa liczba całkowita reprezentująca długość oryginalnej wiadomości w bitach.

Inne techniki szyfrowania

Kryptografia krzywych eliptycznych (ECC)

ECC to nowoczesna kryptografia klucza publicznego opracowana w celu uniknięcia większego użycia klucza kryptograficznego. Asymetryczny system kryptograficzny opiera się na teorii liczb i matematycznych krzywych eliptycznych (strukturze algebraicznej) w celu generowania krótkich, szybkich i solidnych kluczy kryptograficznych. RSA jest powszechnym algorytmem klucza publicznego, ale jego rozmiar klucza jest duży. Szybkość szyfrowania zawsze zależy od rozmiaru klucza: mniejsza długość klucza umożliwia szybsze szyfrowanie. Aby zminimalizować rozmiar klucza, zaproponowano kryptografię krzywych eliptycznych jako zamiennik algorytmu RSA. Operacyjne rozmiary kluczy obu algorytmów, aby osiągnąć podobne cele, są wymienione poniżej:

Rozmiar klucza ECC : Rozmiar klucza RSA

160 : 1024

224 : 2048

256 : 3072

384 : 7680

512 : 15360

Podczas gdy RSA używa klucza o rozmiarze 1024 do szyfrowania danych, ECC zapewnia równe bezpieczeństwo przy stosunkowo mniejszym kluczu o rozmiarze 160. W przypadku obliczeń wysokiego poziomu RSA używa klucza o rozmiarze 7680 do implementacji bezpieczeństwa, podczas gdy ECC może zapewnić ten sam poziom bezpieczeństwa z kluczem o rozmiarze 384.

Kryptografia kwantowa

Ponieważ świat w coraz większym stopniu przyjmuje udostępnianie informacji online, systemy kryptograficzne są świadkami gwałtownego wzrostu ataków bezpieczeństwa. Ponieważ szyfrowanie

matematyczne wykorzystuje cyfry binarne (0 i 1), można je łatwo podsłuchiwać lub manipulować przy użyciu różnych technik. Stąd kryptografia kwantowa została wprowadzona w celu ochrony danych przed kradzieżą Midway (np. atakami MITM). Ta kryptografia jest przetwarzana w oparciu o mechanikę kwantową, taką jak kwantowa dystrybucja klucza (QKD), przy użyciu fotonów zamiast matematyki jako części szyfrowania. w kryptografii kwantowej dane są szyfrowane przez sekwencję fotonów, które mają cechę wirowania podczas podróży z jednego końca na drugi. Fotony te zmieniają swoje kształty podczas przechodzenia przez filtry: pionowe, poziome, ukośne i odwrotne. W tym przypadku obroty pionowe i ukośne oznaczają „jedynki", podczas gdy obroty poziome i ukośne oznaczają „zera".

o Poziomo (-): 0

o Pionowo (|):1

ukośnik odwrotny (/):1

o Ukośnik (\): 0

Atakujący mogą podsłuchiwać, ale nie mogą manipulować danymi, ponieważ fotony są przesyłane przez dowolne filtry. Aby złamać ten mechanizm, atakujący muszą znać dokładny kształt fotonów; jeśli nie uda im się wybrać odpowiedniej transmisji, polaryzacja fotonu jest zniekształcona, a odbiornik wykrywa błąd wskazujący na podsłuch.

Szyfrowanie homomorficzne

Szyfrowanie homomorficzne różni się od konwencjonalnych mechanizmów szyfrowania, w których operacje matematyczne są wykonywane w celu zaszyfrowania tekstu jawnego. Szyfrowanie homograficzne pozwala użytkownikom zabezpieczyć i pozostawić swoje dane w zaszyfrowanym formacie, nawet podczas ich przetwarzania lub manipulacji. W tej technice szyfrowanie i deszyfrowanie jest wykonywane przez tego samego posiadacza klucza. Mechanizm homomorficzny umożliwia użytkownikowi/nadawcy zaszyfrowanie poufnych danych i przekazanie ich do przedsiębiorstwa za pośrednictwem usług w chmurze w celu przetworzenia danych. Czym szyfrowanie homomorficzne różni się od innych mechanizmów szyfrowania:

W szyfrowaniu kluczem prywatnym:

o Tylko właściciele kluczy mogą generować i odszyfrowywać teksty zaszyfrowane przy użyciu podobnych kluczy.

W szyfrowaniu kluczem publicznym:

o Tylko posiadacz klucza publicznego generuje tekst zaszyfrowany, a posiadacz klucza tajnego odszyfrowuje tekst zaszyfrowany.

W szyfrowaniu homomorficznym:

o Posiadacz klucza może wygenerować tekst zaszyfrowany i każdy może zmienić tekst zaszyfrowany, ale ponownie tylko posiadacz klucza może odszyfrować dane.

Powodem korzystania z tej kryptografii jest to, że niezaufana jednostka może manipulować danymi. Dzięki temu mechanizmowi nadawca może samodzielnie szyfrować i deszyfrować dane, co pozwala każdemu na wykonywanie operacji matematycznych na zaszyfrowanym tekście z uwzględnieniem zasad stosowanych przez nadawcę.

Szyfrowanie sprzętowe

Szyfrowanie sprzętowe to technika wykorzystująca sprzęt komputerowy do wspomagania lub zastępowania oprogramowania podczas przeprowadzania procesu szyfrowania danych. Urządzenia oferujące techniki szyfrowania można uznać za sprzętowe urządzenia szyfrujące. Podczas implementacji szyfrowania sprzętowego obciążenie związane z techniką kryptograficzną jest przenoszone na procesory sprzętowe, uwalniając zasoby systemowe do wykonywania innych funkcji. Urządzenia te mogą również przechowywać klucze szyfrowania i inne poufne informacje w zabezpieczonych obszarach pamięci RAM lub innych nieulotnych urządzeniach pamięci masowej, takich jak pamięć flash. Sprzętowe urządzenia szyfrujące redukują zestawy instrukcji, w których można wykonać tylko autoryzowany kod. Urządzenia te nie obsługują oprogramowania firm trzecich, co uniemożliwia wykonanie jakichkolwiek złośliwych programów. Szyfrowanie sprzętowe ma wiele zalet w porównaniu z szyfrowaniem programowym, ponieważ umożliwia szybkie przetwarzanie algorytmu. Zapewnia odporne na manipulacje przechowywanie kluczy i pozwala uniknąć nieautoryzowanego kodu. Niektóre sprzętowe urządzenia szyfrujące to bezprzewodowe punkty dostępowe, Nitrokey, terminale kart kredytowych i sieciowe szyfratory masowe.

Rodzaje sprzętowych urządzeń szyfrujących

Moduł TPM

Trusted Platform Module (TPM) to kryptoprocesor lub układ scalony obecny na płycie głównej. Może bezpiecznie przechowywać klucze szyfrujące i wykonywać wiele operacji kryptograficznych. Moduł TPM oferuje różne funkcje, takie jak uwierzytelnianie integralności platformy, zapewnianie pełnych możliwości szyfrowania dysku, przechowywanie haseł i zapewnianie ochrony licencji oprogramowania.

HSM

Sprzętowy moduł bezpieczeństwa (HSM) to dodatkowe zewnętrzne urządzenie zabezpieczające, które jest używane w systemie do przetwarzania kryptograficznego i może służyć do zarządzania, generowania i bezpiecznego przechowywania kluczy kryptograficznych. HSM oferuje ulepszone obliczenia szyfrowania, które są przydatne w przypadku kluczy symetrycznych dłuższych niż 256 bitów. Wysokowydajne urządzenia HSM są podłączone do sieci za pomocą protokołu TCP/IP. Niektóre urządzenia HSM obejmują Thales Luna Network HSM, nSheild HSM, Utimaco HSM i Cryptosec Dekaton PCI.

Szyfrowanie USB

Szyfrowanie USB to dodatkowa funkcja urządzeń pamięci masowej USB, która oferuje wbudowane usługi szyfrowania. Zaszyfrowane urządzenia USB wymagają systemu poświadczeń na urządzeniu lub poświadczeń programowych lub sprzętowych z komputera. Szyfrowanie USB zapewnia ochronę przed dystrybucją złośliwego oprogramowania przez USB i pomaga zapobiegać utracie i wyciekowi danych. Niektóre sprzętowe urządzenia szyfrowane przez USB to Encrypted USB, Kingston Ironkey D300S i diskAshur Pro.

Szyfrowanie dysku twardego

Szyfrowanie dysku twardego to technologia, dzięki której dane przechowywane na sprzęcie mogą być szyfrowane przy użyciu szerokiej gamy opcji szyfrowania. Urządzenia szyfrujące dysk twardy nie mogą korzystać z klawiatury urządzenia ani czytnika linii papilarnych; zamiast tego potrzebują modułu TPM lub modułu HSM. Urządzenia te można zainstalować jako dysk wewnętrzny w komputerze. Niektóre urządzenia do szyfrowania dysków twardych obejmują 256-bitowe szyfrowanie sprzętowe AES klasy wojskowej oraz szyfrowanie dysków twardych DiskCypher AES Sata.

Kryptografia postkwantowa

Kryptografia postkwantowa jest również znana jako kryptografia kwantowo-odporna i kwantowo-odporna, ponieważ jest zaawansowanym algorytmem kryptograficznym (głównie opartym na kluczu publicznym) przeznaczonym do ochrony systemów bezpieczeństwa przed atakami inicjowanymi zarówno na komputery konwencjonalne, jak i kwantowe. Może również działać w połączeniu z podstawowymi protokołami komunikacyjnymi i sieciami operacyjnymi. Co więcej, kryptografia postkwantowa może służyć jako samodzielny algorytm szyfrowania, który zastępuje obecnie podatne na ataki systemy kryptograficzne zgodne ze standardowymi zasadami bezpieczeństwa. Kryptografia postkwantowa ma na celu zapewnienie bezpiecznej komunikacji o szerokim zasięgu, bezpiecznego przetwarzania tajnego klucza, podpisów opartych na kluczu publicznym i szyfrowania opartego na kluczu publicznym dla zaawansowanych działań, takich jak bezpieczne głosowanie elektroniczne. Kryptografia obejmuje kilka tanich, bezpiecznych systemów i innych systemów, które są zwykle wykorzystywane do komunikacji online. Kryptografia postkwantowa ma na celu przygotowanie się na erę komputerów kwantowych poprzez aktualizację określonych algorytmów i standardów.

Lekka kryptografia

Głównym wyzwaniem w obecnych technikach kryptograficznych jest ich wykorzystanie w urządzeniach o niskim poborze mocy. Naukowcy próbują opracować zwarty algorytm, który byłby bezpieczny pod względem kwantowym i który mógłby skutecznie działać na urządzeniach o niskim poborze mocy. Większość obecnych algorytmów kryptograficznych nadaje się do serwerów i komputerów stacjonarnych, ale lekkie algorytmy kryptograficzne są przeznaczone do zastosowań o niskiej złożoności, takich jak tagi RFID, aplikacje oparte na czujnikach i inne aplikacje oparte na loT. Głównym celem rozwoju lekkiej kryptografii jest zużywanie mniejszej ilości energii i zasobów bez narażania bezpieczeństwa.

Tryby działania szyfru

Tryby działania szyfru, znane również jako tryby działania szyfru blokowego, są używane do szyfrowania stałego bloku tekstu jawnego przy użyciu tajnego klucza, aw niektórych trybach wektora inicjującego. Te tryby działania mogą zapewnić poufność i autentyczność danych. Klient i serwer bezpiecznie wymieniają zaszyfrowany klucz symetryczny, aby ułatwić szyfrowanie i deszyfrowanie. Poniżej omówiono cztery tryby działania szyfru blokowego, które wyjaśniają, jak działają szyfrowanie po stronie źródła i deszyfrowanie po stronie docelowej.

Tryb elektronicznej książki kodów (ECB).

Tryb EBC to prosty proces szyfrowania i deszyfrowania, który wymaga zwykłego tekstu, tajnego klucza i algorytmu szyfrowania blokowego. Tekst jawny jest podzielony na bloki o ustalonej długości, która jest równa rozmiarowi tajnego klucza. W pierwszym etapie szyfrowanie rozpoczyna się od pobrania pierwszego bloku tekstu jawnego, a tajny klucz jest traktowany jako dane wejściowe do algorytmu szyfrowania blokowego; wyjściem jest pierwszy blok tekstu zaszyfrowanego. Proces jest powtarzany dla wszystkich bloków tekstu jawnego. Po stronie docelowej deszyfrowanie odbywa się w taki sam sposób, jak generowanie pierwszego bloku tekstu zaszyfrowanego. Tajny klucz jest traktowany jako dane wejściowe do algorytmu deszyfrowania szyfru blokowego, który wyprowadza pierwszy blok zwykłego tekstu. Ten proces jest powtarzany dla wszystkich bloków tekstu zaszyfrowanego. Jednak ten tryb ma wadę: jeśli równo podzielone bloki tekstu jawnego zawierają te same dane, wyjściowe bloki szyfrujące również zawierają ten sam tekst zaszyfrowany, dając analitykom szansę przewidzenia tekstu jawnego.

Tryb łączenia bloków szyfrów (CBC).

Tryb CBC to ulepszenie w stosunku do EBC, które naprawia większość luk w zabezpieczeniach EBC. W trybie CBC proces szyfrowania wymaga wektora inicjującego oraz tajnego klucza. Najpierw tekst jawny jest dzielony na bloki o tym samym rozmiarze. Pierwszy blok to XOR z wektorem inicjującym (IV), a wynikowy jest wysyłany jako dane wejściowe do algorytmu szyfrowania blokowego wraz z tajnym kluczem. Dane wyjściowe to pierwszy blok tekstu zaszyfrowanego. Ten blok szyfrowania jest używany do wykonywania operacji XOR z następnym blokiem tekstu jawnego; proces łańcuchowy trwa do ostatniego bloku tekstu jawnego. Po stronie docelowej pierwszy blok tekstu zaszyfrowanego i tajnego klucza jest wysyłany do algorytmu deszyfrowania szyfru blokowego, a wynikiem jest XOR z tym samym IV. Dane wyjściowe to pierwszy blok tekstu jawnego. Dla następnych bloków szyfrujących, zamiast IV, wprowadzany jest poprzednio używany blok szyfrujący w celu wykonania XOR; ten proces jest kontynuowany dla pozostałych bloków szyfrujących. Jednak ten tryb ma również problem: jeśli jeden wygenerowany blok tekstu zaszyfrowanego zawiera błąd, jest on propagowany do kolejnych bloków szyfrowania.

Tryb sprzężenia zwrotnego szyfrowania (CFB).

w trybie CFB wcześniej wygenerowany tekst zaszyfrowany jest używany jako informacja zwrotna dla algorytmu szyfrowania w celu zaszyfrowania następnego bloku tekstu jawnego do tekstu zaszyfrowanego. Najpierw wektor inicjujący (IV) jest przechowywany w rejestrze przesuwnym i wysyłany do algorytmu szyfrującego wraz z tajnym kluczem. Z wyniku tego szyfrowania wybierane są pierwsze S bity i wykonywana jest operacja XOR z blokiem tekstu jawnego o rozmiarze S. Wynikowym wyjściem jest blok tekstu zaszyfrowanego. Dla następnego bloku szyfrowania poprzedni blok szyfrowania jest podawany jako wejście do rejestru przesuwnego; przesuwa S bitów w lewo, a proces jest kontynuowany do końca tekstu jawnego. Po stronie docelowej proces deszyfrowania jest taki sam, aż do operacji XOR. Operacja XOR jest wykonywana dla pierwszych S bitów z wyniku algorytmu szyfrowania i pierwszego bloku szyfru, a wyjściem jest pierwszy blok tekstu jawnego. W przypadku kolejnych bloków poprzednio używany blok szyfrowania jest traktowany jako wejście do rejestru przesuwnego, a proces trwa do ostatniego bloku szyfru. Zaletą tego trybu jest to, że utrudnia kryptoanalizę, ponieważ wiąże się z pewną utratą danych z powodu użycia rejestrów przesuwnych.

Tryb licznika

Tryb licznika to tryb działania szyfru blokowego, który wykorzystuje wartość licznika w procesie szyfrowania i deszyfrowania. Wartość licznika jest inicjowana i wysyłana jako dane wejściowe do algorytmu szyfrowania szyfru blokowego tajnym kluczem, a wynik poddawany jest operacji XOR z blokiem tekstu jawnego. Wynikiem jest blok tekstu zaszyfrowanego. Ten proces jest wykonywany sekwencyjnie w celu zaszyfrowania wszystkich pozostałych bloków tekstu jawnego. Po stronie docelowej ten tryb używa tych samych wartości liczników i tajnych kluczy. Ten sam algorytm szyfrowania jest używany do szyfrowania wartości licznika i tajnego klucza, wyniku poddawany jest operacji XOR z otrzymanym blokiem szyfrogramu, a wyjście zawiera tekst jawny. Tryb licznika eliminuje problem propagacji błędów, ponieważ nie wykorzystuje wygenerowanego wcześniej szyfrogramu do szyfrowania lub deszyfrowania. Tryb licznika wymaga zsynchronizowanych wartości licznika zarówno po stronie źródłowej, jak i docelowej.

Tryby uwierzytelnionego szyfrowania

Tryby działania uwierzytelnionego szyfrowania (AE) zapewniają integralność i poufność przesyłanej wiadomości. W dowolnym trybie szyfrowania szyfrowanie/odszyfrowywanie jest możliwe tylko za pomocą współdzielonego tajnego klucza, co zapobiega atakom typu man-in-the-middle (MUM).

Atakujący może jednak wykonać wybrany atak zaszyfrowanym tekstem, aby złamać schemat szyfrowania. Schemat AE rozwiązuje problem wybranych ataków szyfrujących. W trybach AE tekst zaszyfrowany jest łączony z kodem uwierzytelniającym wiadomość (MAC). Dlatego wybranie części tekstu zaszyfrowanego nie jest możliwe, a schemat AE odrzuca niewłaściwe teksty zaszyfrowane podczas deszyfrowania.

Uwierzytelnione szyfrowanie za pomocą kodu uwierzytelniania wiadomości (MAC)

MAC to wartość uzyskana przez haszowanie wiadomości w postaci zwykłego tekstu przy użyciu wspólnego tajnego klucza. Zapewnia integralność wiadomości, a odbiorca może zweryfikować wiadomość za pomocą dołączonej do niej wartości skrótu. Poniżej przedstawiono trzy różne sposoby korzystania z komputera MAC podczas szyfrowania wiadomości.

Szyfruj, a następnie MAC (EtM)

W tym podejściu tekst jawny jest najpierw szyfrowany przy użyciu tajnego klucza. Dla otrzymanego zaszyfrowanego tekstu generowana jest wartość skrótu zwana kodem uwierzytelniania wiadomości (MAC). MAC jest dołączany do zaszyfrowanego tekstu i przesyłany. To podejście zapewnia większe bezpieczeństwo przesyłanej wiadomości niż inne podejścia AE.

Szyfruj i MAC (E&M)

W podejściu E&M najpierw generowany jest adres MAC dla tekstu jawnego, po czym tekst jawny jest szyfrowany przy użyciu tajnego klucza. Na koniec zarówno tekst zaszyfrowany, jak i adres MAC są łączone i przesyłane.

MAC-potem-szyfruj (MtE)

W podejściu MtE MAC jest najpierw generowany dla tekstu jawnego za pomocą funkcji mieszającej, a MAC jest łączony z tekstem jawnym. Kombinacja tekstu jawnego i adresu MAC jest szyfrowana tajnym kluczem w celu utworzenia tekstu zaszyfrowanego. Tekst zaszyfrowany zawiera zaszyfrowany adres MAC.

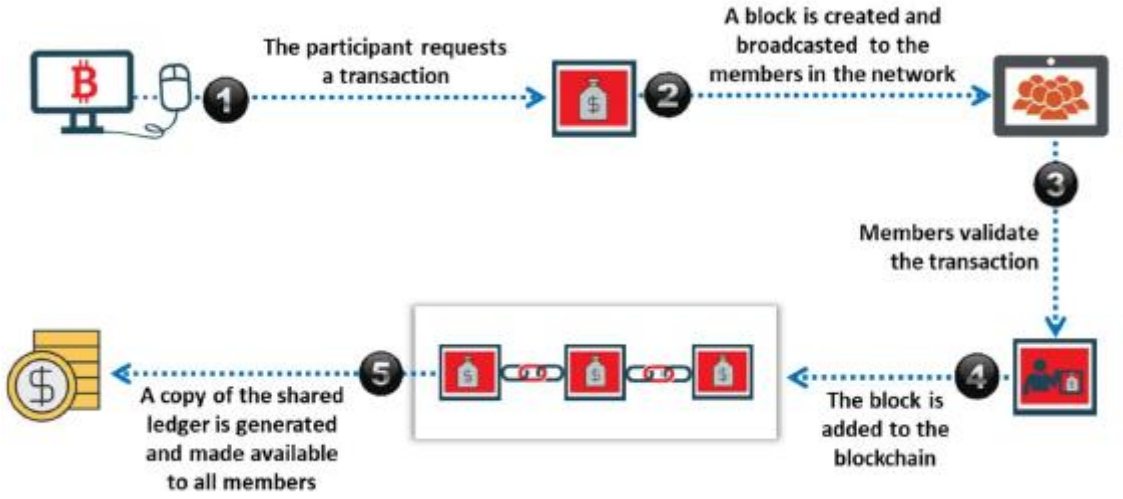
Uwierzytelnione szyfrowanie z powiązanymi danymi (AEAD)

AEAD to kolejne podejście stosowane w celu zapewnienia integralności i autentyczności wiadomości zawierającej zarówno zaszyfrowane, jak i niezaszyfrowane dane. To podejście dodaje dodatkowe dane do tekstu zaszyfrowanego w określonych miejscach, aby udaremnić wybrane ataki szyfrogramem. Nagłówek wiadomości jest niezaszyfrowany, aby odbiorca mógł zweryfikować źródło wiadomości, a ładunek jest szyfrowany w celu zapewnienia poufności.

Zastosowania Kryptografii - Blockchain

Blockchain to rodzaj technologii rozproszonej księgi rachunkowej (DLT), która służy do bezpiecznego rejestrowania i przechowywania historii transakcji w postaci bloków. Dane zapisane w blockchainach są odporne na niechciane modyfikacje, a przejrzystość kont jest utrzymywana dzięki technikom kryptograficznym. W przypadku wielu transakcji tworzonych jest wiele bloków, które są ze sobą połączone kryptograficznie, tworząc „łańcuch bloków”. Ten łańcuch rekordów lub bloków jest znany jako księgi rachunkowe, które są udostępniane w sieci, aby inni uczestnicy byli świadomi wszystkich szczegółów transakcji i liczby bitów posiadanych przez każdego członka. Członkowie sieci uwierzytelniają bloki przy użyciu swoich wartości skrótu, a skróty są dalej weryfikowane przez górników kryptograficznych przy użyciu złożonych algorytmów kryptograficznych, po czym bloki są zatwierdzane do przyłączenia się do mechanizmu łańcucha bloków. Łańcuchy bloków są generalnie

implementowane przy użyciu dwóch mechanizmów: funkcji skrótu (głównie SHA-256) i algorytmów klucza asymetrycznego. Proces sprawdzania poprawności bloków jest znany jako „dowód pracy", za który wynagradzani są górnicy kryptowaluty, a proces dodawania bloków do łańcucha bloków po wykonaniu „dowodu pracy" jest określany jako „kopanie kryptowalut". Każdy blok w łańcuchu bloków składa się z trzech elementów: danych (szczegóły transakcji), hash i hash poprzedniego bloku. Za każdym razem, gdy tworzony jest nowy blok przy użyciu nowej wartości skrótu, wartość skrótu jest współdzielona z następnym blokiem. Pierwszy blok w łańcuchu blokowym określany jako geneza i jest reprezentowany przez Os. Gdy blok zweryfikuje hash swojego poprzedniego bloku, może dołączyć do łańcucha bloków. Jeśli blok zostanie sfałszowany, następny blok w łańcuchu unieważnia go, ponieważ nie pasuje do poprzedniej wartości skrótu w bieżącym bloku. Jednak łańcuch bloków nie jest całkowicie chroniony przez zwykłe generowanie skrótów i porównywanie ich z innymi blokami. Atakujący mogą generować prawidłowe skróty dla każdego bloku przy użyciu wielu technik kryptograficznych. Mechanizm „dowodu pracy" jest stosowany, jak opisano powyżej, w celu ograniczenia takiego ryzyka. Bezpieczeństwo łańcucha bloków zapewnia zarówno efektywne wykorzystanie hashów, jak i „proof of work" przez górników (w księgach publicznych). Poniższy rysunek ilustruje proces tworzenia łańcucha bloków.



Na rysunku blok jest tworzony przez osobę zaangażowaną w transakcję. Ten blok jest udostępniany wszystkim członkom sieci. Każdy członek sprawdza poprawność bloku za pomocą swoich skrótów, po czym blok jest dodawany do łańcucha bloków. W ten sposób każdy członek ma szczegóły nowej transakcji. Blockchainy można tworzyć w czterech wariantach, z których każdy służy do innego celu.

Księga publiczna lub publiczny łańcuch bloków: ten typ łańcucha bloków nie ma centralnego organu ani administracji do zarządzania blokami lub księgami. Jest to zdecentralizowana i pozbawiona uprawnień sieć, w której każdy może dołączać, tworzyć i udostępniać bloki. Po zweryfikowaniu danych w łańcuchu blokowym jest on zabezpieczony przed modyfikacjami lub zmianami. Niektóre przykłady publicznych łańcuchów bloków obejmują Bitcoin i Ethereum. Każdy członek tego łańcucha blokowego może uzyskać dostęp do kopii innych ksiąg bez żadnych uprawnień. Poniżej przedstawiono kluczowe aspekty ksiąg publicznych.

o Każdy w sieci może uczestniczyć w walidacji,

o Po utworzeniu bloku nie można go modyfikować ani modyfikować,

o Księgi publiczne mogą być stosowane w różnych sektorach, takich jak edukacja i opieka zdrowotna,

o Księgi publiczne są odpowiednie dla usług B2C.

Prywatna księga rachunkowa lub prywatny łańcuch bloków: W tego typu łańcuchu bloków nadzorca lub organ centralny decyduje, kto może dołączyć i uczestniczyć w sieci łańcucha bloków. W księdze prywatnej tylko członkowie zaangażowani w transakcję będą mieli wiedzę o odpowiednich księgach. Niektóre przykłady prywatnych łańcuchów bloków to Hyperledger i Ripple (XRP). Poniżej przedstawiono kluczowe aspekty ksiąg prywatnych.

o Administrator zapewnia uczestnikom określony poziom dostępu,

o Organizacje mogą dodawać i usuwać uczestników na żądanie,

o Prywatne księgi rachunkowe mogą być stosowane w sektorach takich jak obronność i bankowość,

o Księgi prywatne są odpowiednie dla usług B2B.

Federacyjny łańcuch blokowy lub łańcuch blokowy konsorcjum: Jest to częściowo zdecentralizowany łańcuch blokowy, w którym grupa osób lub organizacji, a nie pojedynczy podmiot, jak w przypadku prywatnych łańcuchów blokowych, tworzy osobne sieci łańcuchów blokowych i zarządza nimi. Kontrola nad łańcuchem blokowym jest zapewniona grupie z góry określonych lub zaufanych węzłów. Uczestnicy konsorcjum typu blockchain pochodzą głównie z organizacji rządowych lub banków centralnych. Ten rodzaj łańcucha bloków jest niezwykle szybki i skalowalny. EWF (energia) i R3 (banki) to przykłady federacyjnych łańcuchów bloków.

Hybrydowy łańcuch bloków: Jest to połączenie zarówno prywatnego, jak i publicznego łańcucha bloków. W hybrydowym łańcuchu bloków publicznie dostępny jest tylko wybrany zestaw rekordów lub danych z łańcucha bloków; pozostałe dane są poufne w sieci prywatnej. Ten typ łańcucha bloków umożliwia organizacjom wybór danych, które chcą uczynić publicznymi i prywatnymi. Jednym z ważnych przykładów hybrydowego łańcucha bloków jest IBM Food Trust.

Narzędzia kryptograficzne

Ta sekcja dotyczy różnych narzędzi kryptograficznych, których można używać do szyfrowania poufnych danych w celu ochrony ich przed nieautoryzowanym dostępem przez osoby inne niż osoba, dla której są one przeznaczone.

Kalkulatory skrótu MD5 i MD6

Kalkulatory skrótu MD5 i MD6, które używają różnych algorytmów skrótu do konwersji tekstu jawnego na jego równoważną wartość skrótu, omówiono poniżej.

Kalkulator MD5

MD5 Calculator to prosta aplikacja, która oblicza hash MD5 danego pliku. Może być używany z dużymi plikami (np. o rozmiarze kilku gigabajtów). Zawiera licznik postępu i pole tekstowe, z którego końcowy skrót MD5 można łatwo skopiować do schowka. Kalkulator MD5 może być użyty do sprawdzenia integralności pliku. pozwala obliczyć wartość skrótu MD5 wybranego pliku. Kliknij plik prawym przyciskiem myszy i wybierz „Kalkulator MD5”; program obliczy hash MD5. Pole MD5 Digest zawiera obliczoną wartość. Aby porównać ten skrót MD5 z innym, można wkleić inną wartość w polu Porównaj z. Oczywiście znak równości ("=") pojawia się między dwiema wartościami, jeśli są one równe; w przeciwnym razie znak mniejszy niż („<”) lub większy niż („>”) powie ci, że wartości są różne.

HashMyFiles

HashMyFiles to narzędzie, które pozwala obliczyć skróty MD5 i SHA1 jednego lub więcej plików w systemie. Pozwala skopiować listę MD5/SHA1hash do schowka lub zapisać ją w pliku text/html/xml.

Możesz uruchomić HashMyFiles z menu kontekstowego Eksploratora Windows i wyświetlić skróty MD5/SHA1 wybranych plików lub folderów.

Niektóre dodatkowe kalkulatory skrótu MD5 i MD6 są następujące:

HashCalc (https://www.slovosoft.com)

MD6 Hash Generator (https://www.browserling.com)

All Hash Generator (https://www.browserling.com)

md5 hash calculator (https://onlinehoshtools.com)

Message Digester (https://www.freeformotter.com)

Kalkulatory skrótu dla telefonów komórkowych

Poniżej omówiono niektóre kalkulatory skrótu dla urządzeń mobilnych.

hash tools

hash tools to narzędzie do obliczania i sprawdzania skrótu z danego tekstu lub odszyfrowywania skrótu do jego oryginalnego tekstu. W tej aplikacji dostępne funkcje skrótu to MD5, SHA-1, SHA-256, SHA-384 i SHA-512.

Hash Droid

Narzędzie Hash Droid pomaga obliczyć hash z podanego tekstu lub pliku zapisanego na urządzeniu. W tej aplikacji dostępne funkcje skrótu to Adler-32, CRC-32, Haval-128, MD2, MD4, MD5, RIPEMD-128, RIPEMD-160, SHA-1, SHA-256, SHA-384, SHA-512 , Tiger i Whirlpool.

Oto niektóre dodatkowe kalkulatory skrótu MD5 dla urządzeń mobilnych:

Hash Smart Checker - Md5 & Sha256 Calculator (https://ploy.google.com)

Hash Checker (https://ploy.google.com)

Hashr - Hash & Checksum Calculator (https://ploy.google.com)

Hash Calc (https://ploy.google.com)

Hash Generator - Checksum Calculator (https://ploy.google.com)

Narzędzia kryptograficzne

Możesz używać różnych narzędzi kryptograficznych do szyfrowania i odszyfrowywania informacji, plików itp. Narzędzia te implementują różne rodzaje algorytmów szyfrowania.

BCTextEncoder

Narzędzie BCTextEncoder upraszcza kodowanie i dekodowanie danych tekstowych. Kompresuje, szyfruje i konwertuje dane w postaci zwykłego tekstu do formatu tekstowego, który użytkownik może następnie skopiować do schowka lub zapisać jako plik tekstowy. Wykorzystuje metody szyfrowania kluczem publicznym, a także szyfrowanie oparte na hasłach. Ponadto używa silnych i zatwierdzonych algorytmów symetrycznych i klucza publicznego do szyfrowania danych.

Oto niektóre dodatkowe narzędzia kryptograficzne:

AxCrypt (https://www.oxcrypt.net)

Microsoft Cryptography Tools (https://docs.microsoft.com)

Concealer (https://www.belightsoft.com)

SensiGuard (https://www.sensiguord.com)

Challenger (/7ftps://www.encryption-softwore.de)

Narzędzia kryptograficzne dla urządzeń mobilnych

Poniżej omówiono niektóre narzędzia kryptograficzne dla urządzeń mobilnych:

Secret Space Encryptor

Secret Space Encryptor to zintegrowane rozwiązanie do zarządzania hasłami, szyfrowania wiadomości (tekstu) i szyfrowania plików. Chroni wiadomości, notatki i inny tekst przed niepożądanymi czytelnikami. Wykorzystuje algorytmy szyfrowania, takie jak AES (Rijndael) 256bit, RC6 256bit, Serpent 256bit, Blowfish 256bit/448bit, Twofish 256bit i GOST 256bit.

Secure Everything

Secure Everything wykorzystuje szyfrowanie AES do zabezpieczania SMS-ów, filmów, obrazów, plików audio itp. To narzędzie pomaga również w zabezpieczaniu danych karty kredytowej, danych konta bankowego, numeru SSN itp.

Oto niektóre dodatkowe narzędzia kryptograficzne dla urządzeń mobilnych:

Crypto - Tools for Encryption & Cryptography (https://play.google.com)

SSE - File/Text Encryption & Password Vault (https://play.google.com)

Encrypt File Free (https://play.google.com)

EgoSecure Encryption Anywhere (https://play.google.com)

Decrypto (https://play.google.com)

Infrastruktura klucza publicznego (PKI)

Ta sekcja dotyczy infrastruktury klucza publicznego (PKI) i roli każdego składnika infrastruktury PKI, urzędów certyfikacji, takich jak Comodo, IdenTrust, Symantec i GoDaddy, oraz podpisanych certyfikatów (CA) w porównaniu z certyfikatami z podpisem własnym. PKI to architektura bezpieczeństwa opracowana w celu zwiększenia poufności informacji wymienianych w niezabezpieczonym Internecie. Obejmuje sprzęt, oprogramowanie, ludzi, zasady i procedury wymagane do tworzenia, dystrybucji, używania, przechowywania i unieważniania certyfikatów cyfrowych, zarządzania nimi. W kryptografii PKI pomaga powiązać klucze publiczne z odpowiednimi tożsamościami użytkowników za pomocą urzędu certyfikacji (CA).

Składniki PKI

System zarządzania certyfikatami: generuje, dystrybuuje, przechowuje i weryfikuje certyfikaty Certyfikaty cyfrowe: ustala dane uwierzytelniające osoby podczas przeprowadzania transakcji online

Validation Authority (VA): Przechowuje certyfikaty (wraz z ich kluczami publicznymi)

Urząd certyfikacji (CA): wydaje i weryfikuje certyfikaty cyfrowe

Użytkownik końcowy: Żąda certyfikatów, zarządza nimi i ich używa

Urząd rejestracyjny (RA): Działa jako weryfikator CA

PKI to kompleksowy system, który umożliwia korzystanie z usług szyfrowania kluczem publicznym i podpisów cyfrowych w wielu różnych aplikacjach. Uwierzytelnianie PKI zależy od certyfikatów cyfrowych (znanych również jako certyfikaty klucza publicznego), które podpisują i dostarczają urzędy certyfikacji. Certyfikat cyfrowy to podpisane cyfrowo oświadczenie zawierające klucz publiczny i nazwę podmiotu (użytkownika, firmy lub systemu). PKI wykorzystuje kryptografię klucza publicznego, która jest szeroko stosowana w Internecie do szyfrowania wiadomości lub uwierzytelniania nadawców wiadomości. W kryptografii klucza publicznego urząd certyfikacji generuje jednocześnie klucze publiczny i prywatny za pomocą tego samego algorytmu. Klucz prywatny jest w posiadaniu tylko podmiotu (użytkownika, firmy lub systemu) wymienionego w certyfikacie, podczas gdy klucz publiczny jest udostępniany publicznie w katalogu, do którego mają dostęp wszystkie strony. Podmiot zachowuje klucz prywatny w tajemnicy i używa go do odszyfrowania tekstu zaszyfrowanego przez kogoś innego przy użyciu odpowiedniego klucza publicznego (dostępnego w katalogu publicznym). W ten sposób inni szyfrują wiadomości dla użytkownika za pomocą klucza publicznego użytkownika, a użytkownik odszyfrowuje je za pomocą swojego klucza prywatnego.

Etapy procesu PKI są następujące:

1. Podmiot (użytkownik, firma, system) zamierzający w bezpieczny sposób wymieniać informacje występuje do punktu rejestracji (RA) o wydanie certyfikatu.

2. RA otrzymuje żądanie od podmiotu, weryfikuje tożsamość podmiotu i zwraca się do urzędu certyfikacji o wydanie użytkownikowi certyfikatu klucza publicznego.

3. CA wydaje certyfikat klucza publicznego wiążący tożsamość podmiotu z kluczem publicznym podmiotu; następnie zaktualizowane informacje są wysyłane do organu zatwierdzającego (VA).

4. Gdy użytkownik dokonuje transakcji, należycie podpisuje wiadomość cyfrowo za pomocą certyfikatu klucza publicznego i wysyła wiadomość do klienta.

5. Klient weryfikuje autentyczność użytkownika, pytając VA o ważność certyfikatu klucza publicznego użytkownika.

6. VA porównuje certyfikat klucza publicznego użytkownika z aktualnymi informacjami dostarczonymi przez CA i określa wynik (ważny lub nieważny).

Urzędy certyfikacji

Urzędy certyfikacji (CA) to zaufane podmioty, które wydają certyfikaty cyfrowe. Certyfikat cyfrowy poświadcza posiadanie klucza publicznego przez podmiot (użytkownika, firmę lub system) określony w certyfikacie. Pomaga to innym ufać podpisom lub oświadczeniom składanym za pomocą klucza prywatnego, który jest powiązany z certyfikowanym kluczem publicznym.

Poniżej omówiono niektóre popularne urzędy certyfikacji:

Comodo

Comodo oferuje szereg cyfrowych certyfikatów PKI z silnym szyfrowaniem SSL (dostępne 128/256) z Cryptography Gated Server (SGC). Zapewnia standardy poufności, niezawodności systemu i odpowiednich praktyk biznesowych, oceniane przez wykwalifikowanych niezależnych audytów. Oferuje rozwiązania do zarządzania PKI, takie jak Comodo Certificate Manager i Comodo EPKI Manager.

IdenTrust

IdenTrust to zaufana strona trzecia, która świadczy usługi CA dla wielu sektorów, takich jak banki, korporacje, rządy i opieka zdrowotna. Dostarcza rozwiązania, takie jak cyfrowe podpisywanie i pieczętowanie, zgodność z NIST SP 800-171, globalne sieci tożsamości i zarządzany hosting PKI usługi.

DigiCert CertCentral

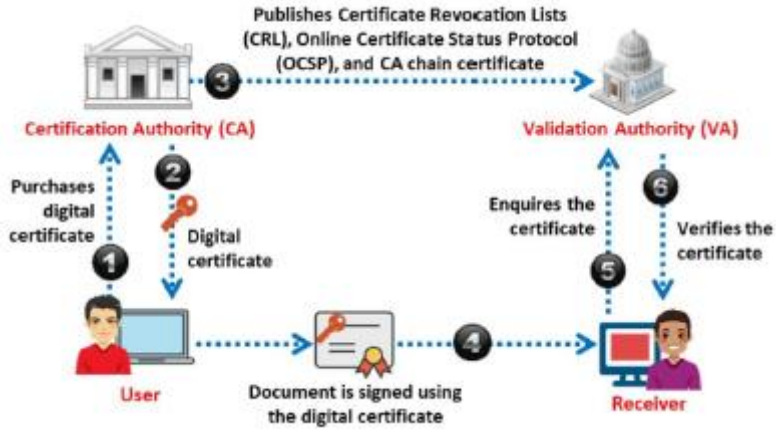
CertCentral upraszcza cały cykl życia, konsolidując zadania wydawania, instalowania, sprawdzania, korygowania i odnawiania certyfikatów TLS/SSL. Zarządza masowym wystawianiem certyfikatów TLS/SSL dla wielu osób i zespołów.

GoDaddy

Certyfikaty GoDaddy SSL oferują pełny zakres certyfikatów zgodnych z wytycznymi CA/Browser Forum. Zapewniają algorytm haszujący SHA-2 i szyfrowanie 2048-bitowe, ochronę nieograniczonej liczby serwerów itp.

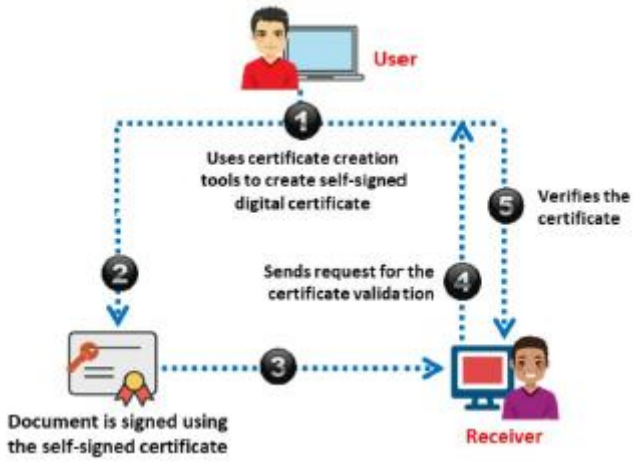
Podpisany certyfikat (CA) a certyfikat z podpisem własnym

Podpisany certyfikat



Jak pokazano na powyższym diagramie, użytkownik otrzymuje certyfikat cyfrowy od godnego zaufania urzędu certyfikacji. Certyfikat cyfrowy zawiera nazwę posiadacza certyfikatu, numer seryjny, daty ważności, kopię klucza publicznego posiadacza certyfikatu oraz podpis cyfrowy urzędu certyfikacji. Użytkownik podpisuje dokument za pomocą certyfikatu cyfrowego i wysyła do odbiorcy. Odbiorca może zweryfikować certyfikat, wysyłając zapytanie do organu zatwierdzającego (VA). VA weryfikuje ważność certyfikatu.

Certyfikat z podpisem własnym



Jak pokazano na powyższym diagramie, użytkownik tworzy certyfikat cyfrowy z podpisem własnym za pomocą narzędzi do tworzenia certyfikatów, takich jak Adobe Acrobat Reader, keytool firmy Java lub pęk kluczy firmy Apple. Certyfikat zawiera nazwę użytkownika, klucz publiczny użytkownika oraz jego podpis cyfrowy. Użytkownik podpisuje dokument za pomocą samopodpisanego certyfikatu i wysyła do odbiorcy. Odbiorca może zweryfikować certyfikat, pytając użytkownika. Użytkownik weryfikuje certyfikat u odbiorcy.

Szyfrowanie wiadomości e-mail

Obecnie większość firm korzysta z poczty e-mail jako podstawowego źródła komunikacji, ponieważ komunikowanie się i udostępnianie informacji jest proste i łatwe. Wiadomości e-mail mogą zawierać poufne informacje o organizacji, takie jak projekty, nadchodzące wiadomości i dane finansowe, które w przypadku uzyskania dostępu przez niewłaściwą osobę mogą spowodować poważne straty dla organizacji. Można chronić wiadomości e-mail zawierające poufne informacje, szyfrując je. Ta sekcja dotyczy mechanizmów bezpieczeństwa poczty e-mail, takich jak podpis cyfrowy, SSL, TLS, zestawy narzędzi kryptograficznych, PGP, GPG i narzędzia do szyfrowania wiadomości e-mail.

Podpis cyfrowy

Podpis cyfrowy wykorzystuje kryptografię asymetryczną do symulacji właściwości bezpieczeństwa podpisu w formie cyfrowej, a nie w formie pisemnej. Podpis cyfrowy to kryptograficzny sposób uwierzytelniania. Kryptografia z kluczem publicznym wykorzystuje szyfrowanie asymetryczne i pomaga użytkownikowi utworzyć podpis cyfrowy. Dwa rodzaje kluczy w kryptografii z kluczem publicznym to klucz prywatny (tylko osoba podpisująca zna ten klucz i używa go do stworzenia podpisu cyfrowego) oraz klucz publiczny (jest powszechnie znany i strona ufająca używa go do weryfikacji podpisu cyfrowego). Funkcja skrótu to algorytm, który pomaga użytkownikowi utworzyć i zweryfikować podpis cyfrowy. Algorytm ten tworzy cyfrową reprezentację, znaną również jako odcisk palca wiadomości. Ten odcisk palca ma wartość skrótu, która jest znacznie mniejsza niż wiadomość, ale jest dla niego unikalna. Jeśli atakujący zmieni wiadomość, funkcja skrótu automatycznie wygeneruje inną wartość skrótu. Aby zweryfikować podpis cyfrowy, potrzebna jest wartość skrótu oryginalnej wiadomości oraz algorytm szyfrowania użyty do stworzenia podpisu cyfrowego. Używając zarówno klucza publicznego, jak i nowego wyniku, weryfikator sprawdza, czy podpis cyfrowy został utworzony z powiązanym kluczem prywatnym i czy nowa wartość skrótu jest taka sama jak oryginalna. Podpis cyfrowy może być dalej chroniony przez zaszyfrowanie podpisanej wiadomości e-mail w celu zachowania poufności.

Bezpieczna warstwa gniazd (SSL)

Protokół Secure Sockets Layer (SSL) to protokół warstwy aplikacji opracowany przez firmę Netscape do zarządzania bezpieczeństwem transmisji wiadomości w Internecie. Służy do zapewnienia bezpiecznego mechanizmu uwierzytelniania między dwiema komunikującymi się aplikacjami, takimi jak klient i serwer. SSL wymaga niezawodnego protokołu transportowego, takiego jak TCP, do transmisji i odbioru danych. Wykorzystuje szyfrowanie asymetryczne RSA (z kluczem publicznym) do szyfrowania danych przesyłanych przez połączenia SSL. Każdy protokół warstwy aplikacji, który jest wyższy niż SSL, taki jak HTTP, FTP i telnet, może tworzyć przezroczystą warstwę na SSL. SSL pełni rolę arbitra między algorytmem szyfrowania a kluczem sesyjnym; weryfikuje również serwer docelowy przed transmisją i odbiorem danych. SSL szyfruje pełne dane protokołu aplikacji w celu zapewnienia bezpieczeństwa.

SSL oferuje również „bezpieczeństwo kanału" z trzema podstawowymi właściwościami:

Kanał prywatny — wszystkie wiadomości są szyfrowane po użyciu prostego uzgadniania w celu zdefiniowania tajnego klucza.

Uwierzytelniony kanał — punkt końcowy konwersacji na serwerze jest zawsze szyfrowany, podczas gdy punkt końcowy klienta jest opcjonalnie uwierzytelniany.

Niezawodny kanał — przesyłanie wiadomości ma kontrolę integralności.

SSL wykorzystuje zarówno asymetryczne, jak i symetryczne mechanizmy uwierzytelniania. Szyfrowanie kluczem publicznym weryfikuje tożsamość serwera, klienta lub obu. Po uwierzytelnieniu klient i serwer mogą tworzyć klucze symetryczne, co pozwala im na szybką komunikację i przesyłanie danych. Sesja SSL jest odpowiedzialna za realizację protokołu uzgadniania SSL w celu uporządkowania stanów serwera i klientów, zapewniając w ten sposób spójność protokołu.

Przepływ protokołu SSL Handshake

Protokół uzgadniania SSL działa na warstwie rekordu SSL. Procesy wykonywane w protokole trójstronnego uzgadniania są następujące:

1. Klient wysyła do serwera wiadomość hello, na którą serwer musi odpowiedzieć wiadomością hello, w przeciwnym razie połączenie zostanie przerwane z powodu wystąpienia błędu krytycznego. Atrybuty ustanowione z powodu powitania serwera i klienta to wersja protokołu, identyfikator sesji, zestaw szyfrów i metoda kompresji.

2. Po nawiązaniu połączenia serwer wysyła klientowi certyfikat w celu uwierzytelnienia. Ponadto serwer może wysłać komunikat wymiany klucza serwera. Podczas uwierzytelniania serwer może poprosić klienta o certyfikat (jeśli jest to odpowiednie dla wybranego zestawu szyfrów).

3. Serwer wysyła komunikat „cześć gotowe", aby poinformować klienta, że faza uzgadniania została zakończona i czeka na odpowiedź klienta.

4. Jeśli klient otrzyma komunikat z żądaniem certyfikatu, musi odpowiedzieć na ten komunikat, wysyłając komunikat o certyfikacie lub alert „brak certyfikatu". Serwer wysyła klientowi komunikat wymiany kluczy. Treść wiadomości zależy od algorytmu klucza publicznego między hello serwera a hello klienta. Jeśli certyfikat wysłany przez klienta ma zdolność podpisywania, certyfikat podpisany cyfrowo weryfikuje wiadomość, a klient ją przesyła.

5. Klient przesyła komunikat o zmienionej specyfikacji szyfru i kopiuje oczekującą specyfikację szyfru do bieżącej specyfikacji szyfru. Klient wysyła komunikat, aby zainicjować zakończenie komunikatu zgodnie z nowym algorytmem, kluczami i kluczami tajnymi.

6. W odpowiedzi serwer odpowiada, wysyłając własny komunikat o zmienionej specyfikacji szyfru, przenosi oczekującą specyfikację szyfru do bieżącej specyfikacji szyfru i inicjuje zakończenie komunikatu zgodnie z nową specyfikacją szyfru. W tym momencie uzgadnianie jest zakończone i serwer rozpoczyna wymianę danych warstwy aplikacji.

Wznowienie poprzedniej sesji lub replikacja istniejącej sesji odbywa się w następujący sposób:

Klient inicjuje komunikację wysyłając wiadomość hello z identyfikatorem sesji, która ma zostać wznowiona.

Jeśli serwer znajdzie dopasowanie, ponownie ustanawia sesję w określonym stanie sesji z tym samym identyfikatorem sesji.

W tym momencie zarówno serwer, jak i klient wymieniają zmienione komunikaty specyfikacji i przechodzą bezpośrednio do gotowych komunikatów.

Po ponownym ustanowieniu serwer i klient wymieniają dane w warstwie aplikacji.

Jeśli identyfikator sesji nie istnieje, serwer tworzy nowy identyfikator sesji. Następnie klient i serwer SSL przeprowadzają pełne uzgadnianie.

Zabezpieczenia warstwy transportowej (TLS)

Protokół Transport Layer Security (TLS) służy do ustanowienia bezpiecznego połączenia między klientem a serwerem oraz zapewnienia prywatności i integralności informacji podczas transmisji. Wykorzystuje klucz symetryczny do szyfrowania masowego, klucz asymetryczny do uwierzytelniania i wymiany kluczy oraz kody uwierzytelniania wiadomości do integralności wiadomości. Wykorzystuje algorytm RSA o sile 1024 i 2048 bitów. Korzystając z TLS, można zmniejszyć zagrożenia bezpieczeństwa, takie jak manipulowanie wiadomościami, fałszowanie wiadomości i przechwytywanie wiadomości. Zaletą TLS jest to, że jest niezależny od protokołu aplikacji. Protokoły wyższego poziomu mogą w przejrzysty sposób leżeć na TLS. TLS składa się z dwóch warstw: protokołu TLS Record i protokołu TLS Handshake.

1. Protokół zapisu TLS

Protokół TLS Record jest protokołem warstwowym. Zapewnia bezpieczne połączenia z metodą szyfrowania, taką jak DES. Zabezpiecza dane aplikacji za pomocą kluczy generowanych podczas uzgadniania oraz weryfikuje ich integralność i pochodzenie. Protokół TLS Record zapewnia bezpieczeństwo połączenia dzięki dwóm podstawowym właściwościom:

o Połączenie jest prywatne: wykorzystuje kryptografię symetryczną do szyfrowania danych (np. DES). Protokół generuje unikalne klucze do szyfrowania symetrycznego dla każdego połączenia, w zależności od tajnego klucza wynegocjowanego przez inny protokół (taki jak protokół TLS Handshake). Można używać protokołu TLS Record Protocol bez szyfrowania.

o Połączenie jest niezawodne: Zapewnia kontrolę integralności wiadomości w czasie transportu wiadomości przy użyciu klucza MAC. Bezpieczne funkcje skrótu (np. SHA, MD5) pomagają wykonywać obliczenia MAC.

Protokół TLS Record wykonuje następujące czynności:

o Fragmentuje dane wychodzące na możliwe do zarządzania bloki i ponownie składa dane przychodzące

o Opcjonalnie kompresuje dane wychodzące i dekompresuje dane przychodzące

o Stosuje MAC do danych wychodzących i używa MAC do weryfikacji danych przychodzących

o Szyfruje dane wychodzące i odszyfrowuje dane przychodzące

Protokół TLS Record Protocol wysyła wychodzące zaszyfrowane dane do warstwy TCP w celu transportu.

2. Protokół uzgadniania TLS

Protokół TLS Handshake umożliwia klientowi i serwerowi wzajemne uwierzytelnienie oraz wybór algorytmu szyfrowania i kluczy kryptograficznych przed wymianą danych przez protokół aplikacji.

Zapewnia bezpieczeństwo połączenia dzięki trzem podstawowym właściwościom:

o Tożsamość peera może być uwierzytelniona za pomocą kryptografii asymetrycznej. Może to być opcjonalne, ale najczęściej jest wymagane dla co najmniej jednego elementu równorzędnego.

o Negocjacja współdzielonego sekretu jest bezpieczna,

o Negocjacje są wiarygodne.

Protokół TLS Handshake działa na bazie protokołu TLS Record Protocol i jest odpowiedzialny za wytwarzanie parametrów kryptograficznych stanu sesji. Na początku komunikacji klient i serwer TLS uzgadniają wersję protokołu, wybierają algorytmy kryptograficzne, opcjonalnie uwierzytelniają się nawzajem i używają asymetrycznych technik kryptograficznych do tworzenia wspólnych tajemnic.

Kroki związane z protokołem TLS Handshake są następujące:

Początkowo klient wysyła do serwera wiadomość „Cześć klienta", której towarzyszy losowa wartość klienta i obsługiwane zestawy szyfrów.

Serwer odpowiada klientowi, wysyłając wiadomość „Server hello" wraz z losową wartością serwera.

Serwer wysyła swój certyfikat do klienta w celu uwierzytelnienia i może zażądać certyfikatu klienta. Serwer wysyła komunikat „Server hello done".

Klient wysyła swój certyfikat do serwera, jeśli jest to wymagane.

Klient generuje losowy sekret przed masteringiem i szyfruje go kluczem publicznym serwera; następnie wysyła zaszyfrowany klucz tajny przed wzorcem do serwera.

Serwer otrzymuje klucz tajny przed wzorcem. Następnie klient i serwer tworzą klucz główny i sesyjny w oparciu o klucz główny i klucz sesyjny.

Klient wysyła do serwera komunikat „Zmień specyfikację szyfru", aby wskazać, że zacznie używać nowych kluczy sesji do mieszania i szyfrowania wiadomości. Klient wysyła również komunikat „Clientfinished".

Serwer otrzymuje od klienta komunikat „Changecipher spec" i przełącza stan zabezpieczeń warstwy zapisu na szyfrowanie symetryczne przy użyciu kluczy sesyjnych. Następnie serwer wysyła do klienta komunikat „Serverfinished".

Teraz klient i serwer mogą wymieniać dane aplikacji przez ustanowiony bezpieczny kanał, a wszystkie wiadomości wymieniane między klientem a serwerem są szyfrowane przy użyciu klucza sesyjnego.

Zestawy narzędzi kryptograficznych

Zestawy narzędzi kryptograficznych obejmują prymitywy kryptograficzne, algorytmy i schematy stosowane w celu zapewnienia bezpieczeństwa różnym aplikacjom. Niektóre zestawy narzędzi kryptograficznych omówiono poniżej:

OpenSSL

OpenSSL to zestaw narzędzi kryptograficznych typu open source, który implementuje protokoły sieciowe SSL i TLS oraz wymagane przez nie standardy kryptograficzne. Jest to narzędzie wiersza poleceń do korzystania z różnych funkcji kryptograficznych krypto-biblioteki OpenSSL z poziomu powłoki. OpenSSL może być używany do tworzenia i zarządzania kluczami prywatnymi, kluczami publicznymi i parametrami; operacje kryptograficzne z kluczem publicznym; tworzenie certyfikatów X.509, CSR i CRL; itp.

Oto niektóre dodatkowe zestawy narzędzi kryptograficznych:

Keyczar (https://github.com)

wolfSSL (https://www.wolfssl.com)

AES Crypto Toolkit (https://www.ni.com)

RELIC (https://code.google.com)

PyCrypto (https://www.pycrypto.org)

Pretty Good Privacy (PGP)

Pretty Good Privacy (PGP) to protokół używany do szyfrowania i odszyfrowywania danych z uwierzytelnianiem i prywatnością kryptograficzną. Jest często używany do kompresji danych, podpisywania cyfrowego, szyfrowania i odszyfrowywania wiadomości, wiadomości e-mail, plików i katalogów oraz do zwiększania prywatności komunikacji e-mail. Algorytm używany do szyfrowania wiadomości to RSA do transportu kluczy i IDEA do szyfrowania wiadomości masowych. PGP używa RSA do obliczania podpisów cyfrowych i MD5 do obliczania skrótów wiadomości. Łączy w sobie najlepsze cechy kryptografii konwencjonalnej (około 1000 razy szybszej niż szyfrowanie z kluczem publicznym) i kryptografii z kluczem publicznym (rozwiązanie problemów z dystrybucją kluczy i transmisją danych), dlatego jest znany jako kryptosystem hybrydowy.

PGP służy do:

Szyfrowanie wiadomości lub pliku przed transmisją, aby tylko odbiorca mógł je odszyfrować i przeczytać

Czytelne podpisywanie wiadomości w postaci zwykłego tekstu w celu zapewnienia autentyczności nadawcy

Szyfrowanie przechowywanych plików komputerowych, aby nikt oprócz osoby, która je zaszyfrowała, nie mógł ich odszyfrować

Usuwanie plików, a nie tylko usuwanie ich z katalogu lub folderu

Kompresja danych do przechowywania lub transmisji

Jak działa PGP?

Szyfrowanie PGP

o Kiedy użytkownik szyfruje dane za pomocą PGP, PGP najpierw kompresuje dane.

o Kompresja danych redukuje wzorce w tekście jawnym, które mogłyby zostać wykorzystane przez większość technik kryptoanalizy do złamania szyfru, znacznie zwiększając w ten sposób odporność na kryptoanalizę.

o Następnie PGP tworzy losowy klucz (GSkAQk49fPD2h), który jest jednorazowym tajnym kluczem.

o PGP używa wygenerowanego losowo klucza do zaszyfrowania tekstu jawnego, w wyniku czego powstaje tekst zaszyfrowany,

o Po zaszyfrowaniu danych losowy klucz jest szyfrowany kluczem publicznym odbiorcy.

o Losowy klucz zaszyfrowany kluczem publicznym (Td7YuEkLg99Qd0) jest wysyłany wraz z tekstem zaszyfrowanym do odbiorcy.

Deszyfrowanie PGP

o Deszyfrowanie działa w odwrotnej kolejności.

o Kopia PGP odbiorcy używa jego klucza prywatnego zamiast klucza publicznego do odzyskania tymczasowego klucza losowego.

o Następnie PGP używa odzyskanego losowego klucza do odszyfrowania konwencjonalnie zaszyfrowanego tekstu.

Uwaga: Każdy etap procesu szyfrowania PGP (haszowanie, kompresja danych, kryptografia z kluczem symetrycznym i kryptografia z kluczem publicznym) wykorzystuje jeden z różnych obsługiwanych algorytmów.

Ochrona prywatności GNU (GPG)

GNU Privacy Guard (GPG) to oprogramowanie zastępujące PGP i bezpłatna implementacja standardu OpenPGP, który służy do szyfrowania i deszyfrowania danych. GPG jest również nazywany oprogramowaniem do szyfrowania hybrydowego, ponieważ wykorzystuje zarówno kryptografię z kluczem symetrycznym, jak i kryptografię z kluczem asymetrycznym w celu zwiększenia szybkości i bezpiecznej wymiany kluczy, co jest osiągane przy użyciu klucza publicznego odbiorcy do szyfrowania klucza sesji. GPG obsługuje również S/MIME i Secure Shell (SSH). Najnowsza wersja GPG obsługuje większość funkcji kryptograficznych, takich jak kryptografia krzywych eliptycznych (ECDSA, ECDH i EdDSA), a także obsługuje bibliotekę kryptograficzną Libgcrypt.

GPG jest używany do:

Właściwe zarządzanie kluczami zarówno prywatnymi, jak i publicznymi

Tworzenie nowych kluczy prywatnych oraz eksportowanie lub importowanie dowolnego klucza, nawet jeśli jest on w jakimś wzmocnionym formacie (np. ASCII)

Przekazanie klucza publicznego do serwera kluczy poprzez podpisanie kodu kluczem GPG posiadającym podpis publiczny

Usuwanie klucza prywatnego z magazynu lokalnego

Szyfrowanie i podpisywanie plików za pomocą kluczy asymetrycznych do szyfrowania dowolnego pliku używanego do poczty e-mail lub FTP

Odszyfrowanie i weryfikacja zaszyfrowanego pliku za pomocą kluczy asymetrycznych

Odłączanie podpisów, gdzie plik podpisu może zostać odłączony od pliku wiadomości

Zarządzanie i budowanie sieci zaufania

Automatyczne zabezpieczanie wiadomości w komunikatorach, takich jak Psi i Fire

Jak działa GPG

Szyfrowanie GPG

o GPG szyfruje wiadomości indywidualnie, używając par kluczy asymetrycznych.

o Użytkownik wysyła surowy plik, a GPG służy do podpisywania pliku przy użyciu klucza prywatnego nadawcy w celu potwierdzenia zawartości pliku w momencie podpisywania.

o Następnie plik jest szyfrowany przy użyciu klucza publicznego odbiorcy. Teraz plik można odszyfrować tylko kluczem prywatnym odbiorcy.

o Po zaszyfrowaniu danych zaszyfrowany plik może być przechowywany lokalnie, dystrybuowany na serwery FTP lub wysyłany do odbiorców poczty elektronicznej.

Deszyfrowanie GPG

o Deszyfrowanie GPG jest procesem odwrotnym do szyfrowania GPG.

o Ponieważ używane są pary kluczy asymetrycznych, GPG szuka klucza prywatnego odbiorcy w celu odszyfrowania pliku.

o Weryfikacja podpisu jest wykonywana automatycznie przez GPG przy użyciu klucza publicznego nadawcy po odszyfrowaniu.

Sieć zaufania (WOT)

Web of trust (WoT) to model zaufania dostępnych systemów PGP, OpenPGP i GnuPG. Jest to idea decentralizacji dystrybucji kluczy wśród użytkowników PGP. W PKI tylko scentralizowana władza, taka jak urząd certyfikacji, podpisuje certyfikaty w sieci, zapewniając autentyczność między kluczem publicznym a jego właścicielem. W WoT wszyscy w sieci są urzędami certyfikacji i mogą podpisywać się w imieniu innych zaufanych podmiotów. WoT to łańcuch sieciowy, w którym poszczególne osoby pośrednio weryfikują nawzajem swoje certyfikaty za pomocą swoich podpisów. Podpisy te weryfikują własność kluczy z różnych poziomów zaufania. Istnieje kilka podobnych poziomów zaufania poprzez bezpośrednie lub pośrednie odniesienia w WoT.

Działanie WOT

W WOT każdy użytkownik PGP w sieci ma krąg kluczy publicznych do szyfrowania danych i przedstawia wielu innych użytkowników, którym ufa. W tym modelu zaufania użytkownik koduje dane kluczem publicznym odbiorcy, który jest odszyfrowywany tylko kluczem prywatnym odbiorcy. Następnie każdy użytkownik w tym modelu cyfrowo podpisuje dane swoimi kluczami prywatnymi; gdy odbiorca sprawdza poprawność klucza publicznego użytkownika, może potwierdzić autentyczność użytkownika. Ten proces zapewni, że dane zostaną otrzymane od ważnego użytkownika bez modyfikacji i że tylko zamierzony użytkownik będzie miał dostęp do informacji, ponieważ tylko on posiada powiązany klucz prywatny.

Szyfrowanie wiadomości e-mail w programie Outlook

Bezpieczne/wielofunkcyjne szyfrowanie rozszerzeń poczty internetowej (S/MIME).

Certyfikacja S/MIME to technika umożliwiająca użytkownikom szyfrowanie wiadomości e-mail. To jest używane do szyfrowania wiadomości e-mail programu Outlook, aby nadawcy i wyznaczeni odbiorcy mieli do nich dostęp bez naruszania integralności wiadomości. Poniżej omówiono kroki szyfrowania wiadomości e-mail przy użyciu szyfrowania S/MIME. Aby wykonać te czynności, do pęku kluczy należy dołączyć certyfikat podpisywania.

Wybierz Plik -> Opcje -> Centrum zaufania -> Ustawienia Centrum zaufania.

Wybierz opcję Zabezpieczenia poczty e-mail w lewym okienku.

W sekcji Zaszyfrowana wiadomość e-mail kliknij opcję Ustawienia obok opcji Ustawienia domyślne.

W wyskakującym oknie Zmień ustawienia zabezpieczeń, w sekcji Certyfikaty i algorytmy wybierz certyfikat S/MIME dla opcji Certyfikat podpisywania i Certyfikat szyfrowania, a następnie kliknij OK.

Szyfrowanie wiadomości Microsoft 365

Usługa Office 365 Message Encryption (OME) umożliwia użytkownikom wysyłanie zaszyfrowanych wiadomości e-mail na dowolny adres e-mail. OME wymaga od odbiorców zalogowania się do konta Microsoft Office 365 lub użycia hasła jednorazowego w celu uwierzytelnienia.

Kroki szyfrowania wiadomości e-mail przy użyciu szyfrowania OME są następujące. W treści wiadomości e-mail wybierz menu Opcje, przejdź do opcji Szyfruj i wybierz szyfrowanie, które obejmuje wymagane ograniczenia, takie jak Tylko szyfrowanie lub Nie przesyłaj dalej.

Wykonaj poniższe czynności, aby zaszyfrować pojedynczą wiadomość:

Kliknij Plik, a następnie Właściwości w treści wiadomości e-mail.

W oknie Właściwości kliknij przycisk Ustawienia zabezpieczeń w sekcji Zabezpieczenia.

W wyskakującym oknie Właściwości zabezpieczeń zaznacz opcję Szyfruj zawartość wiadomości i załączniki i kliknij OK.

Wykonaj poniższe czynności, aby zaszyfrować wszystkie wiadomości wychodzące:

Wybierz Plik -> Opcje -> Centrum zaufania -> Ustawienia Centrum zaufania.

Wybierz opcję Zabezpieczenia poczty e-mail w lewym okienku.

W sekcji Zaszyfrowana wiadomość e-mail zaznacz opcję Szyfruj zawartość i załączniki dla wszystkich wiadomości wychodzących i kliknij OK.

Podpisywanie/szyfrowanie wiadomości e-mail na komputerze Mac

Czynniki zabezpieczające pocztę e-mail w poczcie Apple można ulepszyć, wdrażając funkcje szyfrowania zwane podpisem cyfrowym i szyfrowaniem wiadomości, dostarczane przez dostawców usług pocztowych Apple. Użytkownik może wysłać podpisaną cyfrowo i zaszyfrowaną wiadomość e-mail z urządzenia Apple.

Wysyłanie podpisanych cyfrowo i zaszyfrowanych wiadomości e-mail

Uzyskaj dostęp do poczty Apple na urządzeniu Mac i kliknij Plik -> Nowa wiadomość.

Umieść kursor na polu Od, a następnie wybierz rachunek, na którym znajduje się rachunek osobisty certyfikat w pęku kluczy z wyskakującego menu.

Uwaga: Po włączeniu osobistego certyfikatu w pęku kluczy w wiadomości e-mail pojawi się niebieski znacznik («), wskazujący, że wiadomość wychodząca może być podpisana cyfrowo.

Odbieranie podpisanych cyfrowo i zaszyfrowanych wiadomości e-mail

Odebrana wiadomość e-mail z ikoną podpisu (*) wskazuje, że wiadomość została podpisana cyfrowo przez uprawnionego nadawcę. Odbiorca może uzyskać dostęp do certyfikatu cyfrowego nadawcy, klikając ikonę ^.

Pojawienie się komunikatu ostrzegawczego oznacza, że dane oryginalnej wiadomości e-mail zostały naruszone i że nie można zidentyfikować tożsamości podmiotu uwierzytelniającego.

Odebrany e-mail z ikoną zamkniętej kłódki (fl) wskazuje, że wiadomość jest zaszyfrowana. Odbiorca może uzyskać dostęp do tej wiadomości e-mail tylko po podaniu ważnego klucza prywatnego do odszyfrowania.

Szyfrowanie/odszyfrowywanie wiadomości e-mail za pomocą OpenPGP

Chociaż PGP ma dobry współczynnik bezpieczeństwa, jest podatny na ataki online. Poczta e-mail z aktywnym OpenPGP (hybryda PGP) wdrożona w wielu środowiskach, takich jak Windows, macOS, Android, iOS, Linux i wtyczki do przeglądarek, zapewnia silniejszy współczynnik bezpieczeństwa. Wiadomość e-mail z zainstalowanym aktywnym OpenPGP i skonfigurowanym z kompatybilnymi rozszerzeniami przeglądarki może zwiększyć czynniki bezpieczeństwa w środowisku przeglądarki. Użytkownicy mogą korzystać z narzędzi rozszerzających przeglądarkę, takich jak FlowCrypt, które można skonfigurować wraz z OpenPGP, w celu bezpiecznej komunikacji e-mail.

FlowCrypt (Gmail)

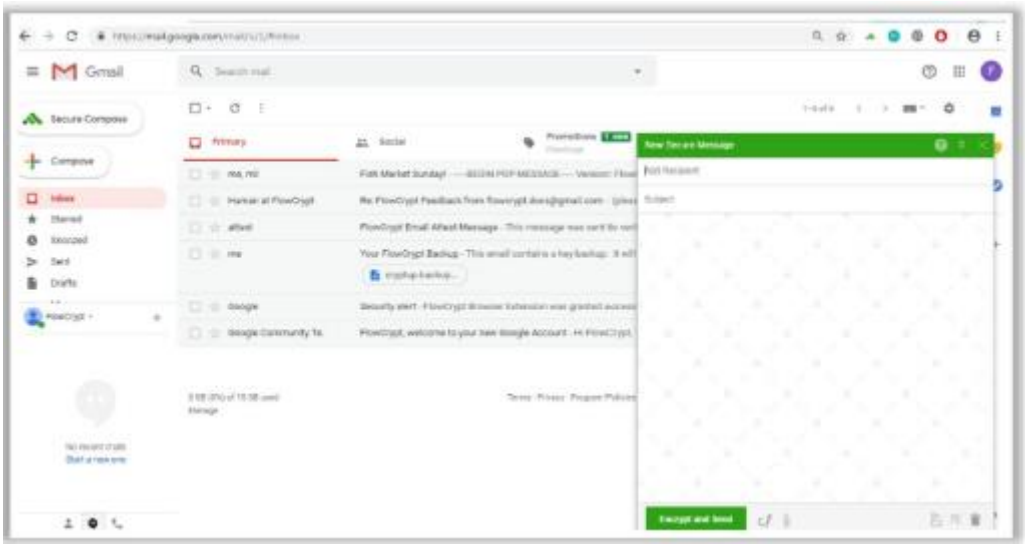
FlowCrypt to kompleksowe oprogramowanie do szyfrowania poczty e-mail skonfigurowane z OpenPGP do zabezpieczania wiadomości e-mail i załączników w poczcie Google (G Suite/Business/Enterprise). Umożliwia szyfrowanie/odszyfrowywanie wychodzących/przychodzących wiadomości e-mail na urządzeniach użytkowników za pomocą prywatnych/publicznych kluczy dostępu do danych. Wykonaj poniższe czynności, aby wysyłać i odbierać zaszyfrowaną pocztę (Gmail) przy użyciu rozszerzenia przeglądarki FlowCrypt w Chrome lub Firefox z OpenPGP:

Otwórz przeglądarkę i włącz Open PGP. Zainstaluj i skonfiguruj rozszerzenie przeglądarki FlowCrypt.

Na końcu nadawcy

Zaloguj się na konto Gmail z https://moil.google.com przy użyciu tej samej przeglądarki.

Wybierz ikonę Bezpieczne tworzenie wiadomości w lewym okienku. Pojawi się nowa skrzynka pocztowa tworzenia bezpiecznej wiadomości, jak pokazano na zrzucie ekranu poniżej:



Wprowadź adres e-mail odbiorcy w polu Dodaj odbiorcę (Nowa zabezpieczona wiadomość).

Uwaga: Jeśli nazwa odbiorcy jest wyświetlana na zielono w skrzynce pocztowej tworzenia wiadomości, odbiorca uzyskuje również dostęp do swojej przeglądarki z zainstalowanym FlowCrypt.

Dodaj temat i treść (dodaj załączniki, jeśli istnieją) dla wiadomości e-mail.

Wyślij wiadomość na adres odbiorcy, klikając Szyfruj i wyślij.

Na końcu odbiorcy

Odbiorca może uzyskać dostęp do wiadomości e-mail, klikając nowo otrzymaną wiadomość e-mail.

Uwaga: W zależności od ustawień użytkownik może wprowadzić klucze PGP do odszyfrowywania wiadomości e-mail.

Narzędzia do szyfrowania wiadomości e-mail

Niektóre ważne narzędzia do szyfrowania wiadomości e-mail używane do zabezpieczania wiadomości e-mail to:

Źródło RMaila:

RMail to narzędzie zabezpieczające pocztę e-mail, które zapewnia otwarte śledzenie, dowód dostarczenia, szyfrowanie wiadomości e-mail, podpisy elektroniczne, funkcję przesyłania dużych plików itp. RMail bezproblemowo współpracuje z istniejącymi platformami poczty e-mail użytkowników, w tym Microsoft Outlook, Gmail itp. Za pomocą tego narzędzia możesz może szyfrować poufne wiadomości e-mail i załączniki w celu zapewnienia bezpieczeństwa lub zgodności z prawem.

Oto niektóre dodatkowe narzędzia do szyfrowania wiadomości e-mail:

Virtru (https://www.virtru.com)

ZixMail (https://getzixmoil.com)

Egress Bezpieczny e-mail i transfer plików (https://www.egress.com)

Proofpoint Email Protection (https://www.proofpoint.com)

Paubox (https://www.paubox.com)

Szyfrowanie dysku

Szyfrowanie dysku szyfruje każdy bit danych przechowywanych na dysku lub woluminie dysku, zapobiegając w ten sposób nielegalnemu dostępowi do przechowywania danych. Ta sekcja dotyczy koncepcji szyfrowania dysku i różnych narzędzi do szyfrowania dysku. Szyfrowanie dysku to technologia, która chroni poufność danych przechowywanych na dysku poprzez przekształcenie go w nieczytelny kod za pomocą oprogramowania lub sprzętu do szyfrowania dysku uniemożliwienie dostępu do niego nieupoważnionym użytkownikom. Szyfrowanie dysku zapewnia poufność i prywatność za pomocą haseł i ukrytych woluminów. Szyfrowanie dysku działa podobnie do szyfrowania wiadomości tekstowych i chroni dane nawet wtedy, gdy system operacyjny nie jest aktywny. Używając programu szyfrującego swój dysk (DVD, pendrive, zewnętrzny dysk twardy, kopia zapasowa), można zabezpieczyć dowolne lub wszystkie informacje na dysku i zapobiec dostaniu się ich w niepowołane ręce. Oprogramowanie do szyfrowania dysków szyfruje informacje na dysku w nieczytelny kod. Dopiero po odszyfrowaniu informacji na dysku można je odczytać i wykorzystać. Szyfrowanie dysku jest przydatne, gdy użytkownik musi fizycznie przesłać poufne informacje. Ponadto szyfrowanie dysku może chronić wymianę informacji w czasie rzeczywistym przed zagrożeniami kompromitującymi. Kiedy użytkownicy wymieniają zaszyfrowane informacje, szanse na ich naruszenie są zminimalizowane. Jedynym sposobem, w jaki osoba atakująca może uzyskać dostęp do informacji, jest odszyfrowanie wiadomości. Ponadto oprogramowanie szyfrujące zainstalowane w systemie użytkownika zapewnia bezpieczeństwo systemu. Zainstaluj oprogramowanie szyfrujące w każdym systemie, który zawiera cenne informacje lub systemy, które są narażone do nieograniczonego transferu danych.

Narzędzia do szyfrowania dysku

Wspólnym celem narzędzi do szyfrowania dysku jest zaszyfrowanie partycji dysku w celu zapewnienia poufności przechowywanych na niej informacji. Poniżej omówiono niektóre narzędzia do szyfrowania dysku.

VeraCrypt

VeraCrypt to oprogramowanie do tworzenia i utrzymywania woluminu szyfrowanego w locie (urządzenie do przechowywania danych). W przypadku szyfrowania w locie dane są automatycznie szyfrowane bezpośrednio przed zapisaniem i odszyfrowywane natychmiast po załadowaniu, bez interwencji użytkownika. Żadne dane przechowywane na zaszyfrowanym woluminie nie mogą zostać odczytane (odszyfrowane) bez użycia poprawnego hasła/plików kluczy) lub poprawnych kluczy szyfrujących. Szyfrowany jest cały system plików (np. nazwy plików, folderów, wolne miejsce, metadane itp.).

Pliki mogą być kopiowane do iz zamontowanego wolumenu VeraCrypt, tak jak są kopiowane na/z dowolnego normalnego dysku (np. poprzez proste operacje przeciągnij i upuść). Pliki są automatycznie odszyfrowywane w locie (w pamięci/RAM), podczas gdy są odczytywane lub kopiowane z zaszyfrowanego wolumenu VeraCrypt. Podobnie, pliki zapisywane lub kopiowane na wolumin VeraCrypt są automatycznie szyfrowane w locie (tuż przed zapisaniem ich na dysku) w pamięci RAM.

Szyfrowanie dysku Rohos

Rohos to narzędzie do szyfrowania dysku, które umożliwia użytkownikom tworzenie ukrytych i zaszyfrowanych partycji na komputerze, dysku flash USB lub usłudze przechowywania w chmurze, takiej jak Dysk Google, OneDrive i Dropbox. Narzędzie wykorzystuje zatwierdzony przez NIST algorytm szyfrowania AES i klucz szyfrujący o długości 256 bitów, co umożliwia automatyczne szyfrowanie.

Szyfrowanie dysków bitlocker

Funkcja BitLocker zapewnia ochronę danych offline i systemu operacyjnego komputera. Pomaga zapewnić, że dane przechowywane na komputerze z systemem Windows® nie zostaną ujawnione, jeśli komputer zostanie zmodyfikowany, gdy zainstalowany system operacyjny jest w trybie offline. Funkcja BitLocker używa mikroczipa zwanego Trusted Platform Module (TPM), aby zapewnić lepszą ochronę danych i zachować integralność komponentów wczesnego rozruchu. Moduł TPM może pomóc w ochronie danych przed kradzieżą lub nieautoryzowanym dostępem, szyfrując cały wolumen systemu WindowsNieco dodatkowego dysku narzędzia szyfrujące są następujące:

FinalCrypt (http://www.finolcrypt.org)

Broadcom Encryption (https://www.broodcom.com)

FileVault (https://support.opple.com)

Gillsoft Full Disk Encryption (https://gilisoft.com)

Checkpoint Full Disk Encryption (https://www.checkpoint.com)

Narzędzia do szyfrowania dysków dla systemu Linux

Cryptsetup

Cryptsetup to narzędzie służące do wygodnego konfigurowania szyfrowania dysku w oparciu o moduł jądra DMCrypt. Obejmuje zwykłe woluminy dm-crypt, woluminy LUKS, pętle AES, TrueCrypt (w tym rozszerzenie VeraCrypt) i formaty BitLocker.

Poniżej przedstawiono kilka dodatkowych narzędzi do szyfrowania dysku dla systemu Linux:

eCryptfs (https://www.ecryptfs.org)

Cryptmount (http://cryptmount.sourceforge.net)

Tomb (https://www.dyne.org)

CryFS (https://www.cryfs.org)

GnuPG (https://gnupg.org)

Narzędzia do szyfrowania dysków dla systemu macOS

FileVault 2

Pełne szyfrowanie dysku FileVault (FileVault 2) wykorzystuje technologię szyfrowania XTS-AES-128 wraz z 256-bitowym kluczem, aby zapobiec nieautoryzowanemu dostępowi do informacji na dysku startowym. FileVault 2 jest dostępny dla wersji macOS Lion lub nowszych. Po włączeniu FileVault 2 użytkownik musi zalogować się na swoje konto. Wkrótce po włączeniu FileVault proces szyfrowania jest automatycznie inicjowany w tle. Pliki są szyfrowane zaraz po utworzeniu, ponieważ są przechowywane na dysku startowym.

Poniżej przedstawiono kilka dodatkowych narzędzi do szyfrowania dysku dla systemu macOS:

VeraCrypt (https://www.veracrypt.fr)

Sophos SafeGuard Encryption (https://www.sophos.com)

BestCrypt Volume Encryption for Mac (https://www.jetico.com)

Broadcom Symantec Endpoint Encryption (https://www.broodcom.com)

Rohos Logon Key for Mac (https://rohos.com)

Kryptoanaliza

Atakujący mogą przeprowadzać różne ataki kryptograficzne, aby obejść zabezpieczenia systemu kryptograficznego, wykorzystując luki w kodzie, szyfrach, protokołach kryptograficznych lub schematach zarządzania kluczami. Ten proces jest znany jako kryptoanaliza. Kryptoanaliza to badanie szyfrów, tekstu zaszyfrowanego lub kryptosystemów z możliwością identyfikacji ich luk, a tym samym wyodrębnienia tekstu jawnego z tekstu zaszyfrowanego, nawet jeśli klucz kryptograficzny lub algorytm użyty do zaszyfrowania tekstu jawnego jest nieznany. Ta sekcja dotyczy różnych ataków kryptograficznych, które osoba atakująca przeprowadza w celu złamania zabezpieczeń systemów kryptograficznych, a także różnych technik i narzędzi do analizy kryptograficznej, które pomagają w naruszeniu bezpieczeństwa kryptograficznego.

Metody kryptoanalizy

Kryptoanaliza liniowa

Kryptoanaliza liniowa opiera się na znalezieniu afinicznych przybliżeń działania szyfru. Jest powszechnie używany w szyfrach blokowych. Ta technika została wymyślona przez Mitsarue Matsui. Jest to atak ze znanym tekstem jawnym i wykorzystuje przybliżenie liniowe do opisania zachowania szyfru blokowego. Mając wystarczającą liczbę par tekstu jawnego i odpowiadającego mu tekstu zaszyfrowanego, można uzyskać bity informacji o kluczu. Oczywiście im więcej par tekstu jawnego i zaszyfrowanego, tym większe są szanse powodzenia. Pamiętaj, że kryptoanaliza to próba złamania kryptografii. Na przykład w przypadku 56-bitowego standardu szyfrowania danych (DES) brutalne wyłamanie klucza może wymagać nawet 256 prób. Kryptoanaliza liniowa wymaga 243 znanych tekstów jawnych. Jest to lepsze niż użycie siły, ale nadal jest niepraktyczne w większości sytuacji. Matematyka może być nieco skomplikowana dla początkujących kryptografów, ale spójrzmy na jej podstawy. Dzięki tej metodzie równanie liniowe wyraża równość dwóch wyrażeń, które składają się z XORowanych zmiennych binarnych. Na przykład następujące równanie XOR sumuje pierwszy i trzeci bit tekstu jawnego, a pierwszy bit tekstu zaszyfrowanego jest równy drugiemu bitowi klucza:

$$P_1 \oplus P_3 \oplus C1 = K_2$$

Możesz użyć tej metody, aby powoli odtworzyć użyty klucz. Po wykonaniu tej czynności dla każdego bitu otrzymasz równanie postaci

$$P_{i1} \oplus P_{i2} \oplus \ldots \oplus C_{j1} \oplus C_{j2} \oplus \ldots = K_{k1} \oplus K_{k2} \oplus \ldots$$

Następnie możesz użyć algorytmu 2 Matsui, używając znanych par tekst jawny-zaszyfrowany, aby odgadnąć wartości bitów klucza biorących udział w przybliżeniu. Dla każdego zestawu wartości bitów klucza po prawej stronie (nazywanego kluczem częściowym) policz, ile razy przybliżenie jest prawdziwe dla wszystkich znanych par tekst jawny-tekst zaszyfrowany; nazwij tę liczbę T. Częściowy klucz, którego T ma największą różnicę bezwzględną od połowy liczby par tekst jawny-tekst zaszyfrowany, jest oznaczony jako najbardziej prawdopodobny zestaw wartości dla tych bitów klucza.

Kryptoanaliza różnicowa

Kryptoanaliza różnicowa jest formą kryptoanalizy mającą zastosowanie do algorytmów z kluczem symetrycznym. Wymyślili go Eli Biham i Adi Shamir. Zasadniczo jest to badanie różnic w danych

wejściowych i ich wpływu na wynikową różnicę w wynikach. Pierwotnie działał tylko z wybranym tekstem jawnym. Może również pracować ze znanym tekstem jawnym i zaszyfrowanym.

Kryptoanaliza integralna

Kryptoanaliza integralna została po raz pierwszy opisana przez Larsa Knudsena. Ten atak jest szczególnie przydatny przeciwko szyfrom blokowym opartym na sieciach substytucyjno-permutacyjnych jako rozszerzenie kryptoanalizy różnicowej. Analiza różnicowa dotyczy par wejść, które różnią się tylko jedną pozycją bitu, przy czym wszystkie inne bity są identyczne. Analiza całkowa dla rozmiaru bloku b utrzymuje stałe b-k bitów i przeprowadza pozostałe k bitów przez wszystkie 2k możliwości. Dla k = 1 jest to po prostu kryptoanaliza różnicowa, ale przy k > 1 jest to nowa technika.

Kryptoanaliza kwantowa

Kryptoanaliza kwantowa to proces łamania algorytmów kryptograficznych przy użyciu komputera kwantowego. Atakujący mogą użyć algorytmu faktoryzacji kwantowej Shora w algorytmach kryptograficznych z kluczem publicznym, takich jak RSA i Elliptic Curve Diffie-Flellman (ECDH), aby znaleźć czynniki dużych liczb w czasie wielomianowym, oraz algorytmu wyszukiwania kwantowego Grovera, aby przyspieszyć wyszukiwanie klucza metodą brute-force dla bloku szyfry (AES) lub funkcje haszujące (SFIA). Aby przeprowadzić kryptoanalizę, osoby atakujące muszą uzyskać zaszyfrowaną zawartość, a proces ten wymaga znacznego czasu i zasobów kwantowych. Poniżej podano wymagane zasoby aby przeprowadzić kryptoanalizę.

o Szerokość obwodu: Określa, ile bitów kwantowych lub kubitów jest wymaganych w kroku czasowym

o Głębokość obwodu: Określa kroki czasowe wymagane dla obwodu

o Liczba bramek: Określa, ile bramek kwantowych jest zaimplementowanych w obwodzie

o Liczba bramek T: Określa, ile bramek T jest zaimplementowanych w obwodzie

o Głębokość T: Określa kroki czasowe wymagane dla bramki T

o MAXDEPTFI: Określa maksymalną głębokość obwodu (np. 240,264 lub 296)

Metody łamania kodów

Można zmierzyć siłę algorytmu szyfrowania za pomocą różnych technik łamania kodu, z których niektóre są następujące:

Brutalna siła

Łamacze kodów lub kryptoanalitycy pracują nad odzyskaniem zwykłego tekstu wiadomości bez wcześniejszej znajomości wymaganego klucza. Mogą najpierw spróbować odzyskać klucz lub mogą zająć się samą wiadomością. Powszechną techniką kryptoanalityczną jest atak siłowy lub wyczerpujące wyszukiwanie, w którym klucze są określane poprzez wypróbowanie każdej możliwej kombinacji znaków. Skuteczność ataku brute-force zależy od konfiguracji sprzętowej. Zastosowanie szybszych procesorów oznacza, że w ciągu sekundy będzie testowanych więcej kluczy. Kryptoanalitycy przeprowadzili udany atak brute-force na metodę szyfrowania DES, który skutecznie sprawił, że DES stał się przestarzały.

Analiza częstotliwości

Analiza częstotliwości to badanie częstotliwości liter lub grup liter w zaszyfrowanym tekście. Analiza częstotliwości liter i słów to kolejna metoda używana do łamania szyfrów. Działa na zasadzie, że w

dowolnym odcinku języka pisanego pewne litery i kombinacje liter występują z różną częstotliwością. Ta technika sprawdza, ile razy dany symbol pojawia się w zaszyfrowanym tekście. Na przykład litera „e" jest powszechną literą w języku angielskim. Jeśli litera „k" pojawia się często w zaszyfrowanym tekście, można rozsądnie wywnioskować, że „k" w zaszyfrowanym języku jest odpowiednikiem „e" w języku angielskim .

Zaszyfrowany kod źródłowy jest bardziej narażony na tego typu ataki, ponieważ słowa takie jak „#define", „struct", „else" i „return" są często powtarzane w kodzie. Zaawansowane kryptosystemy są wymagane do utrzymania bezpieczeństwa wiadomości przed analizą częstotliwości.

Trickery and Deceit

Trickery and Deceit wymagają wysokiego poziomu umiejętności matematycznych i kryptograficznych. Polega na wykorzystaniu technik socjotechnicznych do wydobywania kluczy kryptograficznych.

Przykład: całkiem łatwo jest odszyfrować całą wiadomość, jeśli użytkownik zna część jej treści.

Atakujący może użyć technik inżynierii społecznej, aby oszukać lub przekupić kogoś w celu zaszyfrowania i wysłania znanej wiadomości, którą po przechwyceniu można następnie łatwo odszyfrować przy użyciu standardowych technik analizy kryptograficznej.

Podkładka jednorazowa

Można złamać każdy szyfr, jeśli ma się wystarczającą ilość czasu i zasobów. Istnieje jednak wyjątek zwany jednorazowym padem, którego użytkownicy zakładają, że jest niezniszczalny nawet przy nieskończonych zasobach. Jednorazowa podkładka zawiera najczęściej niepowtarzalny zestaw liter lub cyfr, które system wybiera losowo. Użytkownik zapisuje je na małych karteczkach, a następnie skleja w zeszyt.

Przykład jednorazowego użycia podkładki:

Nadawca szyfruje tylko jeden znak tekstu jawnego za pomocą każdej litery klucza na klawiaturze, a odbiorca odszyfrowuje każdą literę tekstu zaszyfrowanego za pomocą identycznej klawiatury. Gdy list wykorzystuje stronę, on lub ona odrywa ją od bloku i bezpiecznie odrzuca; stąd nazwa onetime pad.

Wada:

Długość klucza jest taka sama jak długość wiadomości, co uniemożliwia szyfrowanie i wysyłanie dużych wiadomości.

Ataki kryptograficzne

Atakujący przeprowadzają ataki kryptograficzne, zakładając, że kryptoanalityk ma dostęp do zaszyfrowanych informacji. Atak kryptograficzny lub kryptoanaliza obejmuje badanie różnych zasad i metod odszyfrowywania tekstu zaszyfrowanego z powrotem do tekstu jawnego bez znajomości klucza.

Różne rodzaje ataków kryptograficznych są następujące:

Atak wyłącznie zaszyfrowanym tekstem

Tylko zaszyfrowany tekst jest mniej skuteczny, ale o wiele bardziej prawdopodobny dla atakującego. Atakujący ma dostęp tylko do zbioru tekstów zaszyfrowanych. Jest to znacznie bardziej prawdopodobne niż znany tekst jawny, ale jest również najtrudniejsze. Atak jest całkowicie udany, jeśli można wydedukować odpowiednie teksty jawne (lub jeszcze lepiej klucz). Możliwość uzyskania jakichkolwiek informacji o podstawowym tekście jawnym jest nadal uważana za sukces. Co więc

atakujący robi z zaszyfrowanymi tekstami, które zgromadził? Możesz analizować je pod kątem wzorców, próbując znaleźć coś, co dałoby ci wskazówkę co do klucza, który został użyty do ich złamania. Często wynikiem tego ataku jest tylko częściowe zerwanie, a nie całkowite zerwanie.

Adaptacyjny atak wybranego tekstu jawnego

W tego typu ataku atakujący ma pełny dostęp do wiadomości w postaci zwykłego tekstu, w tym do jej zaszyfrowania, a także może modyfikować treść wiadomości, wykonując serię interaktywnych zapytań, wybierając kolejne bloki tekstu jawnego na podstawie informacji z poprzedniego zapytania i funkcje szyfrujące. Aby przeprowadzić ten atak, osoba atakująca musi wejść w interakcję z urządzeniem szyfrującym.

Atak z wybranym tekstem jawnym

Wybrany atak w postaci zwykłego tekstu jest wysoce skutecznym rodzajem ataku kryptoanalitycznego. W tym ataku atakujący uzyskuje teksty zaszyfrowane odpowiadające wybranemu przez siebie zestawowi tekstów jawnych. Dzięki temu osoba atakująca może podjąć próbę uzyskania użytego klucza, a tym samym odszyfrować inne wiadomości zaszyfrowane tym kluczem. Zasadniczo, ponieważ atakujący zna tekst jawny i wynikowy tekst zaszyfrowany, zyskuje wiele informacji na temat użytego klucza. Ta technika może być trudna, ale nie jest niemożliwa.

Atak pokrewnymi kluczami

Atak kluczem pokrewnym jest podobny do ataku z wybranym tekstem jawnym, z tą różnicą, że atakujący może uzyskać teksty zaszyfrowane za pomocą dwóch różnych kluczy. W rzeczywistości jest to bardzo przydatny atak, jeśli można uzyskać tekst jawny i pasujący tekst zaszyfrowany. Atak wymaga, aby różne klucze były ze sobą ściśle powiązane, np. w środowisku bezprzewodowym, w którym kolejne klucze mogą pochodzić z poprzednich kluczy. Następnie, chociaż klucze są różne, są blisko. Podobnie jak atak tylko zaszyfrowany, ten typ ataku najprawdopodobniej przyniesie tylko częściową przerwę.

Atak słownikowy

W tym ataku atakujący konstruuje słownik tekstu jawnego wraz z odpowiadającym mu zaszyfrowanym tekstem, który przeanalizował i uzyskał przez określony czas. Po zbudowaniu słownika, jeśli atakujący uzyska tekst zaszyfrowany, użyje już utworzonego słownika, aby znaleźć odpowiedni tekst jawny. Atakujący używają tej techniki do odszyfrowywania kluczy, haseł, haseł i tekstu zaszyfrowanego.

Atak ze znanym tekstem jawnym

W tym ataku jedyną informacją dostępną dla atakującego są niektóre bloki tekstu jawnego wraz z odpowiednim szyfrogramem i algorytmem używanym do szyfrowania i deszyfrowania tekstu. Korzystając z tych informacji, dedukuje się klucz używany do generowania zaszyfrowanego tekstu, aby rozszyfrować inne wiadomości. Atak ten działa na szyfrach blokowych i jest przykładem liniowej kryptoanalizy. Znane bloki tekstu jawnego są generowane przy użyciu serii inteligentnych domysłów i logiki, a nie poprzez dostęp do tekstu jawnego przez kanał.

Atak wybranym tekstem zaszyfrowanym

Atakujący uzyskuje teksty jawne odpowiadające dowolnemu zestawowi tekstów zaszyfrowanych według własnego wyboru. Korzystając z tych informacji, atakujący próbuje odzyskać klucz użyty do zaszyfrowania tekstu jawnego. Aby przeprowadzić ten atak, atakujący musi mieć dostęp do kanału komunikacyjnego między nadawcą a odbiorcą.

Istnieją dwa warianty tego ataku:

o Atak w porze lunchu lub o północy: w tym ataku osoba atakująca może mieć dostęp do systemu tylko przez ograniczony czas lub może uzyskać dostęp tylko do kilku par tekst jawny-tekst zaszyfrowany.

o Atak adaptacyjny z wybranym tekstem zaszyfrowanym: w tym ataku atakujący wybiera serię tekstów zaszyfrowanych, a następnie obserwuje powstałe bloki tekstu jawnego.

Atak gumowego węża

Atakujący wydobywają z osoby tajemnice kryptograficzne (np. hasło do zaszyfrowanego pliku) za pomocą przymusu lub tortur. Ogólnie rzecz biorąc, ludzie pod presją nie mogą zachować bezpieczeństwa i ujawnią tajemnice lub ukryte informacje. Atakujący torturują ofiary, aby ujawnić tajne klucze lub hasła używane do szyfrowania informacji.

Atak wybranym kluczem

W tego typu ataku atakujący nie tylko łamie tekst zaszyfrowany, ale także włamuje się do większego systemu, który jest zależny od tego zaszyfrowanego tekstu. Atakujący zazwyczaj łamie n-bitowy szyfr klucza na 2 operacje n/2. Kiedy atakujący złamie szyfr, uzyskuje dostęp do systemu i może kontrolować cały system, uzyskać dostęp do poufnych danych i przeprowadzać dalsze ataki.

Atak czasowy

Opiera się na wielokrotnym mierzeniu dokładnych czasów wykonania modułowych operacji potęgowania. Atakujący próbuje złamać zaszyfrowany tekst, analizując czas potrzebny do wykonania algorytmu szyfrowania i deszyfrowania dla różnych danych wejściowych. W komputerze czas potrzebny do wykonania operacji logicznej może się różnić w zależności od podanych danych wejściowych. Osoba atakująca próbuje wyodrębnić tekst jawny, podając różne dane wejściowe.

Atak typu Man-in-the-Middle

Ten atak jest przeprowadzany na protokół kryptograficzny. Tutaj atakujący przechwytuje komunikację między klientem a serwerem i negocjuje parametry kryptograficzne. Za pomocą tego ataku osoba atakująca może odszyfrować zaszyfrowaną zawartość i uzyskać poufne informacje, takie jak hasła systemowe. Atakujący może również wstrzyknąć polecenia, które mogą modyfikować przesyłane dane. Atakujący zwykle przeprowadza atak MITM na kryptosystemy z kluczem publicznym, w których przed nawiązaniem komunikacji wymagana jest wymiana kluczy.

Brutalny atak

Złamanie systemów kryptograficznych jest niezwykle trudne, ponieważ nie mają one praktycznych słabości, które można by wykorzystać; nie jest to jednak niemożliwe. Systemy kryptograficzne wykorzystują algorytmy kryptograficzne do szyfrowania wiadomości. Te algorytmy kryptograficzne używają klucza do szyfrowania lub odszyfrowywania wiadomości. W kryptografii klucz ten jest ważnym parametrem określającym transformację tekstu jawnego na tekst zaszyfrowany i odwrotnie. Jeśli jesteś w stanie odgadnąć lub znaleźć klucz użyty do odszyfrowania, możesz odszyfrować wiadomości i przeczytać je zwykłym tekstem. Klucze 128-bitowe są powszechne i uważane za silne. Z punktu widzenia bezpieczeństwa, aby uniknąć odgadnięcia klucza, systemy kryptograficzne używają kluczy generowanych losowo. To sprawia, że poświęcasz sporo wysiłku na odgadnięcie klucza. Flowever, nadal masz możliwość określenia klucza używanego do szyfrowania lub deszyfrowania. Możesz próbować odszyfrować wiadomość przy użyciu wszystkich możliwych kluczy, dopóki nie odkryjesz klucza używanego do szyfrowania. Ta metoda odkrywania klucza nazywana jest atakiem brute-force. Flowever wymaga ogromnej mocy obliczeniowej. Jest to proces wymagający dużej ilości zasobów i czasu. W przypadku dowolnego protokołu bez wad średni czas potrzebny do znalezienia klucza w ataku

brute-force zależy od długości klucza. Jeśli długość klucza jest krótka, znalezienie klucza zajmie mniej czasu; jeśli jest długi, zajmie więcej czasu. Atak brute-force zakończy się sukcesem wtedy i tylko wtedy, gdy atakujący ma wystarczająco dużo czasu na odkrycie klucza. Jednak wymagany czas zależy od długości klucza.

Trudność ataku brute-force zależy od różnych czynników, takich jak

Długość klucza

Liczba możliwych wartości, które może mieć każdy składnik klucza

• Czas potrzebny na próbę każdego klawisza

• Jeśli istnieje jakikolwiek mechanizm, który blokuje atakującego po określonej liczbie nieudanych prób

Na przykład, jeśli system może brutalnie wymusić 56-bitowy klucz DES w ciągu jednej sekundy, to dla 128-bitowego klucza AES zajmie to około 149 bilionów lat. Aby przeprowadzić atak brute-force, atakujący potrzebuje dwukrotnie więcej czasu na każdy dodatkowy bit długości klucza; powodem jest to, że liczba kluczy podwaja się wraz ze wzrostem o jeden bit. Jednak atak brute-force ma większe szanse na osiągnięcie rezultatów.

Power/Cost	40 bits (5 char)	56 bits (7 char)	64 bits (8 char)	128 bits (16 char)
$ 2K (1 PC. Can be achieved by an individual)	1.4 min	73 days	50 years	10^20 years
$ 100K (this can be achieved by a company)	2 sec	35 hours	1 year	10^19 years
$ 1M (Achieved by a huge organization or a state)	0.2 sec	3.5 hours	37 days	10^18 years

Urodzinowy atak

Atak urodzinowy odnosi się do klasy ataków siłowych na hasze kryptograficzne, które sprawiają, że ataki siłowe są łatwiejsze do wykonania. Atak ten opiera się na paradoksie urodzinowym, w którym prawdopodobieństwo, że dwie lub więcej osób w grupie 23 osób ma urodziny tego samego dnia, jest większe niż 0,5.

Urodzinowy paradoks

Na przykład, ile osób jest potrzebnych, aby mieć duże prawdopodobieństwo, że dwie osoby podzielą ten sam dzień urodzin (tj. ten sam dzień i miesiąc, a nie rok). Jest 365 dni w roku i dlatego można by pomyśleć, że co najmniej połowa lub 182 osoby obchodzą te same urodziny, podczas gdy w rzeczywistości są to tylko 23 lata! Podstawowa idea jest następująca: Ile osób musiałbyś mieć w pokoju, aby istniało duże prawdopodobieństwo, że dwie z nich będą miały urodziny tego samego dnia (ten sam dzień i miesiąc, ale nie rok). Oczywiście, jeśli umieścisz w pokoju 367 osób, co najmniej dwie z nich muszą mieć urodziny w tym samym dniu i miesiącu, ponieważ rok ma tylko 365 dni i dodatkowy dzień w przypadku roku przestępnego. Paradoksem nie jest liczba osób, których potrzebujesz, aby zagwarantować dopasowanie, ale liczba osób, których potrzebujesz, aby mieć duże

prawdopodobieństwo. Nawet przy 23 osobach w pokoju istnieje 50% szans, że dwie z nich będą miały urodziny tego samego dnia i miesiąca.

Urodzinowy paradoks: prawdopodobieństwo

Prawdopodobieństwo, że pierwsza osoba nie ma tych samych urodzin co poprzednia osoba, wynosi 100%, ponieważ w zestawie nie ma poprzednich osób. Można to zapisać jako 365/365. Druga osoba ma tylko jedną poprzedzającą osobę, a prawdopodobieństwo, że druga osoba ma urodziny inne niż pierwsza, wynosi 364/365. Trzecia osoba może dzielić urodziny z dwiema poprzedzającymi osobami, więc szanse na wspólne urodziny z którąkolwiek z dwóch poprzedzających osób wynoszą 363/365. Ponieważ każdy z nich jest niezależny, możemy obliczyć prawdopodobieństwo w następujący sposób: 365/365 * 364/365 * 363/365 * 362/365 ... * 342/365 (342 to prawdopodobieństwo, że 23. osoba ma takie same urodziny z osobą poprzedzającą). Kiedy przekonwertujemy je na wartości dziesiętne, otrzymamy (obcięcie do trzeciego miejsca po przecinku) 1* 0,997 * 0,994 * 0,991 * 0,989 * 0,986 * ... 0,936 = 0,49 lub 49%. Jest to prawdopodobieństwo, że 23 osoby nie będą miały wspólnych urodzin; w związku z tym istnieje 51% szans (większe niż parzyste) na to, że dwoje z 23 będzie miało wspólne urodziny. Ideą ataku urodzinowego jest próba znalezienia kolizji dla danego skrótu. Załóżmy teraz, że hash to MD5 z wyjściem 128-bitowym. Musiałbyś wypróbować 2A128 możliwych skrótów, aby zagwarantować kolizję, która jest bardzo dużą liczbą. W zapisie dziesiętnym tak 3.402823669209384634633746074317 7e+38 Teraz, wychodząc z paradoksu urodzin, potrzebujemy 1.174V2A128 lub 21656477542535013597.184 haszy, aby zagwarantować kolizję. Co więcej, jest to nadal bardzo duża liczba, ale o wiele rzędów wielkości mniejsza niż wspomniana powyżej wartość.

Atak Meet-in-the-Middle na schematy podpisów cyfrowych

Atak typu meet-in-the-middle jest najlepszą metodą ataku dla algorytmów kryptograficznych wykorzystujących wiele kluczy do szyfrowania. Atak ten zmniejsza liczbę permutacji siłowych wymaganych do dekodowania tekstu zaszyfrowanego za pomocą więcej niż jednego klucza. Atak typu meet-in-the-middle wykorzystuje kompromis czasoprzestrzenny; jest to również rodzaj ataku urodzinowego, ponieważ wykorzystuje matematykę stojącą za paradoksem urodzinowym, a atak zajmuje mniej czasu niż wyczerpujący atak. Nazywa się to atakiem typu meet-in-the-middle, ponieważ polega na szyfrowaniu z jednego końca i deszyfrowaniu z drugiego końca, tym samym spotykając się „w środku". W ataku typu meet-in-the-middle atakujący używa znanej wiadomości w postaci zwykłego tekstu. Atakujący ma dostęp zarówno do tekstu jawnego, jak i odpowiedniego zaszyfrowanego tekstu. Ten atak jest przeprowadzany przez osoby atakujące w celu fałszowania wiadomości, które wykorzystują wiele schematów szyfrowania. Rozważmy przykład, w którym zwykły tekst to „Jan", a wynikowa wiadomość zaszyfrowana podwójnie DES to „AvBr". Aby odzyskać oba klucze (tj. keyl i key2) używane do szyfrowania, atakujący przeprowadza atak brute-force na keyl, używając wszystkich 256 różnych pojedynczych możliwych kluczy DES do zaszyfrowania tekstu jawnego „John" i zapisuje każdy klucz oraz wynikowy pośredni zaszyfrowany tekst w tabeli. Atakujący brutalnie wymusza klucz2 i odszyfrowuje „AvBr" do 256 razy. Atak jest skuteczny, gdy drugi atak brute-force daje taki sam wynik jak pośredni tekst zaszyfrowany obecny w tablicy szyfrogramu po pierwszym ataku brute-force. Gdy atakujący znajdzie dopasowanie, może określić oba klucze i zakończyć atak. Co najwyżej ten atak wymaga łącznie 256 lub maksymalnie 257 operacji. Dzięki temu atakujący może łatwo uzyskać dostęp do danych w porównaniu z podwójnym DES.

Atak bocznym kanałem

Atak typu side-channel to fizyczny atak przeprowadzany na urządzenie kryptograficzne/kryptosystem w celu zdobycia poufnych informacji. Kryptografia jest na ogół częścią sprzętu lub oprogramowania

działającego na urządzeniach fizycznych, takich jak półprzewodniki (rezystor, tranzystor itd.), które wchodzą w interakcje z różnymi czynnikami środowiskowymi i wpływają na nie w następujący sposób:

Pobór energii

Ujawnia operacje, które mają miejsce i związane z nimi parametry. Ma zastosowanie tylko do sprzętowych systemów kryptograficznych. Analiza zużycia energii jest dwojakiego rodzaju:

o Simple Power Analysis (SPA): Dostarcza informacji dotyczących instrukcji wykonywanej w określonym czasie oraz wartości wejść i wyjść

o Analiza Mocy Różnicowej (DPA): Nie wymaga znajomości szczegółów implementacji algorytmu; wykorzystuje metody statystyczne

Pole elektromagnetyczne

Komponenty komputerowe często generują promieniowanie elektromagnetyczne. Mierząc zmiany pola elektromagnetycznego na powierzchni chipa, osoba atakująca może przewidzieć jego korelację z podstawowymi obliczeniami i danymi oraz może wydedukować cenne informacje na temat tych obliczeń i danych.

Emisja światła

Kuhn odkrył, że średnia jasność rozproszonego odbicia od ściany kineskopu (CRT) jest wystarczająca do zrekonstruowania sygnału wyświetlanego na CRT. W ten sposób osoba atakująca może zebrać wiele informacji, odczytując sygnały emitowane przez optyczne kanały wyjściowe zaufanej platformy obliczeniowej.

Według Loughry'ego i Umphressa można wywnioskować, jakie dane przetwarza komputer na podstawie promieniowania optycznego emitowanego przez diody LED (diody elektroluminescencyjne).

Czas i opóźnienie

Systemy często obliczają algorytmy kryptograficzne bez spójności czasowej dzięki optymalizacji wydajności. Jeśli takie obliczenia obejmują tajne dane, wówczas zmiany w czasie można wykorzystać do wywnioskowania tajnych informacji. Tutaj atakujący analizuje czas potrzebny urządzeniu kryptograficznemu na przetworzenie każdej wiadomości w celu odkrycia tajnych parametrów.

Dźwięk

Ataki akustyczne wykorzystują dźwięk wytwarzany podczas obliczeń. Te emisje akustyczne pochodzą z klawiatur i komponentów komputerowych (np. procesora, pamięci). Uzyskane w ten sposób informacje są określane jako informacje kanału bocznego. Ataki typu side-channel różnią się od tradycyjnych/teoretycznych form ataków, takich jak ataki brute-force. Atak typu side-channel zależy od sposobu, w jaki systemy implementują algorytmy kryptograficzne, a nie od samego algorytmu.

Techniki łagodzenia ataków w kanale bocznym

Korzystaj z protokołów sprawdzających różnicową analizę mocy (DPA) z ograniczonymi charakterystykami wycieków w kanałach bocznych i aktualizuj klucze, zanim nagromadzenie wycieków stanie się znaczące.

Używaj algorytmów o stałym czasie (tj. bez opóźnień zależnych od danych).

Algorytmy maskujące i ślepe wykorzystujące losowe wartości jednorazowe.

Zaimplementuj techniki dopasowywania różnicowego, aby zminimalizować zależne od danych wycieki netto z przejść na poziomie logicznym.

Wstępnie ładuj rejestry i magistrale, aby usunąć sygnatury wycieków z przewidywalnych przejść danych.

Dodaj amplitudę lub szum czasowy, aby zmniejszyć stosunek sygnału do szumu atakującego.

Zastosuj oprogramowanie do analizy bezpieczeństwa w celu wykrywania ataków na etapie projektowania sprzętu.

Zastosuj kondycjonowanie i filtrowanie linii zasilania, aby spowolnić analizę monitorowania zasilania.

Używaj wyświetlaczy z materiałami tłumiącymi sygnał, aby blokować promieniowanie elektromagnetyczne.

Zaimplementuj techniki zaślepiania, takie jak zmiana danych wejściowych lub wyjściowych algorytmu na losowy stan.

Atak bocznym kanałem — scenariusz

Załóżmy, że zaszyfrowane dane mają zostać odszyfrowane i wyświetlone jako zwykły tekst w zaufanej strefie. W momencie deszyfrowania w kryptosystemie osoba atakująca rejestruje fizyczne czynniki środowiskowe, takie jak taktowanie i rozpraszanie mocy, działające na komponenty komputera. Atakujący następnie analizuje te informacje, aby uzyskać przydatne informacje do kryptoanalizy.

Haszowy atak kolizyjny

Atak z kolizją mieszającą jest wykonywany poprzez znalezienie dwóch różnych komunikatów wejściowych, które dają ten sam wynik mieszania. Na przykład w ataku kolizyjnym hash „hash(al) = hash(a2)", gdzie al i a2 reprezentują losowe wiadomości. Ponieważ sam algorytm losowo wybiera te wiadomości, osoby atakujące nie mają żadnej roli w zawartości tych wiadomości. Pozwala to atakującemu na przeprowadzenie kryptoanalizy poprzez wykorzystanie podpisu cyfrowego użytego do wygenerowania innej wiadomości z tą samą wartością skrótu. Jedną z najpopularniejszych funkcji skrótu jest SHA-1, która jest szeroko stosowana jako algorytm podpisu cyfrowego. SHA-1 konwertuje wiadomość wejściową na nieustrukturyzowane ciągi liczb i alfabetów o stałej długości, które działają jak odcisk palca dla wysyłanego pliku. Dlatego atakujący próbuje zidentyfikować podobne zahaszowane dane wyjściowe, aby uzyskać cyfrowe podpisy ofiary. Pozwala to atakującemu sfałszować cyfrowy podpis ofiary wiadomości al na wiadomości a2. Gdy atakujący wykryje kolizję w mieszaniu, może zidentyfikować więcej kolizji, łącząc dane z pasującymi wiadomościami.

Atak DUHK

Don't Use Hard-Coded Keys (DUHK) to luka w zabezpieczeniach kryptograficznych, która umożliwia atakującym uzyskanie kluczy szyfrowania używanych do zabezpieczania sieci VPN i sesji internetowych. Atak ten dotyczy głównie sprzętu/oprogramowania korzystającego z generatora liczb losowych (RNG) ANSI X9.31. Generatory liczb pseudolosowych (PRNG) generują losowe sekwencje bitów na podstawie początkowej tajnej wartości, zwanej ziarnem, oraz bieżącego stanu. Algorytm PRNG generuje klucze kryptograficzne, które służą do ustanowienia bezpiecznego kanału komunikacji przez VPN. W niektórych przypadkach klucz początkowy jest zakodowany na stałe w implementacji. Oba czynniki są kluczowymi kwestiami ataku DUHK, ponieważ każdy atakujący może połączyć ANSI X9.31 z zakodowanym na stałe kluczem początkowym, aby odszyfrować zaszyfrowane dane wysyłane lub odbierane przez to urządzenie. Atakujący typu man-in-the-middle używają ataku DUHK, aby poznać

wartość początkową, obserwować bieżącą sesję i uzyskać wartość bieżącego stanu. Korzystając z tego ataku, osoby atakujące mogą zidentyfikować klucze szyfrujące i wykraść poufne informacje, takie jak krytyczne dane biznesowe, poświadczenia użytkownika i dane karty kredytowej.

Atak Tęczowej Taberli

Atak tęczowej tabeli to rodzaj ataku kryptograficznego, w którym osoba atakująca używa tęczowej tabeli do odwrócenia kryptograficznych funkcji skrótu. Atak Rainbow Table wykorzystuje technikę wymiany kryptoanalitycznej pamięci czasowej, która jest mniej czasochłonna niż inne techniki. Wykorzystuje już obliczone informacje przechowywane w pamięci do szyfrowania. W ataku Rainbow Table atakujący tworzy z góry tabelę wszystkich możliwych haseł i odpowiadających im wartości skrótu, zwaną tęczową tabelą. Tęczowa tabela zawiera listy słów, takie jak pliki słowników i listy siłowe, oraz ich wartości skrótu. Jest to tabela przeglądowa, szczególnie używana do odzyskiwania hasła w postaci zwykłego tekstu z tekstu zaszyfrowanego. Atakujący używa tej tabeli do wyszukiwania hasła i próbuje odzyskać je z skrótów haseł. Atakujący oblicza skrót dla listy możliwych haseł i porównuje go z wcześniej obliczoną tabelą skrótów (tęczową tabelą). Jeśli zostanie znalezione dopasowanie, może złamać hasło. Łatwo jest odzyskać hasła, porównując przechwycone skróty haseł z wcześniej obliczonymi tabelami.

Atak powiązanym kluczem

Atakujący przeprowadzają atak związany z kluczem, wykorzystując matematyczny związek między kluczami w szyfrze i uzyskując dostęp do funkcji szyfrowania i deszyfrowania. Motywem atakującego do przeprowadzenia tego ataku jest znalezienie powiązanych kluczy prywatnych/tajnych. Aby przeprowadzić ten atak, atakujący monitoruje operację szyfrowania, w której wartości kluczy są początkowo nieznane, a następnie po dokładnym zbadaniu przechwytuje relację między tymi kluczami. Na przykład atakujący obserwuje i stwierdza, że ostatnie 80 bitów kluczy jest zawsze takich samych, ale początkowo nie wie o tych bitach. Najlepszym przykładem tego ataku jest błąd w kryptogramie WEP, tj. w sieciach bezprzewodowych. W tym ataku każdy punkt dostępowy i urządzenie interfejsu użytkownika używa tego samego klucza. Szyfrowanie używane w WEP to szyfr strumieniowy znany jako RC4; należy zauważyć, że te same klucze nie powinny się powtarzać w szyfrze strumieniowym. Aby tego uniknąć, WEP integruje 24-bitowy wektor początkowy (IV) w każdym przesyłanym pakiecie. Kluczem RC4 dla tego konkretnego pakietu jest IV powiązany z kluczem WEP. Klucze WEP należy zmienić ręcznie, jednak rzadko się to zdarza. W związku z tym atakujący zauważa, że klucze używane do szyfrowania są często takie same. Ta wada stwarza różne zagrożenia dla WEP, zwłaszcza przy użyciu paradoksu urodzinowego, ponieważ na każde 4096 pakietów dwie strony będą dzielić ten sam IV, a zatem ten sam klucz RC4. Ta prosta forma szyfrowania umożliwia atakującemu wykorzystanie słabych kluczy RC4, co ostatecznie wymusza odzyskanie klucza WEP.

Dopełnienie ataku wyroczni

W ataku wyroczni dopełniającej atakujący wykorzystują sprawdzanie poprawności dopełnienia zaszyfrowanej wiadomości w celu odszyfrowania tekstu zaszyfrowanego. Taki atak jest również znany jako atak Vaudenay. W wielu algorytmach kryptograficznych opartych na szyfrze blokowym komunikaty są dopełniane dodatkowymi losowymi bitami, tak aby długość ostatniego bloku była wymaganej wielkości. Wyrocznia dopełniania to funkcja takiego szyfrowania, która weryfikuje, czy wiadomość została poprawnie dopełniona. Atak ten jest przeprowadzany głównie na algorytmach działających w trybie CBC (Cipher Block Chaining). W tym ataku serwer (wyrocznia) ujawnia informacje o tym, czy wypełnienie zaszyfrowanej wiadomości zostało wykonane poprawnie. W niektórych przypadkach informacje te umożliwiają atakującym odszyfrowanie i opcjonalnie zaszyfrowanie wiadomości przy użyciu klucza serwera (klucza Oracle) bez dostępu do odpowiedniego klucza

szyfrującego. Rozważmy na przykład scenariusz ze standardową implementacją odszyfrowywania CBC. Serwer odszyfrowuje wszystkie bloki tekstu zaszyfrowanego, weryfikuje dopełnienie, usuwa wszystkie dodatkowe dopełnienia wykonane podczas procesu szyfrowania, a następnie zwraca oryginalną wiadomość do aplikacji lub użytkownika. Jeśli na przykład serwer nie może odszyfrować wiadomości z powodu błędu dopełnienia i zwraca komunikat o błędzie „Błąd odszyfrowywania: nieprawidłowe wypełnienie" zamiast ogólnego komunikatu „Odszyfrowanie nie powiodło się", informacje te mogą zostać wykorzystane przez atakującego. Atakujący może użyć serwera jako wyroczni do odszyfrowania zaszyfrowanych wiadomości.

DROWN Atak

Odszyfrowywanie RSA za pomocą przestarzałego i osłabionego szyfrowania (DROWN) to poważna luka, która może wpływać na ważne protokoły kryptograficzne, takie jak HTTPS i inne usługi kryptograficzne zależne od SSL i TSL. Atak DROWN to słabość międzyprotokołowa, która może komunikować się i inicjować atak na serwery obsługujące najnowsze pakiety protokołów SSLv3/TLS. Jest to nowa forma międzyprotokołowego ataku wyroczni Bleichenbachera. Serwer jest krytycznie narażony na atak DROWN, jeśli serwer zezwala na połączenie SSLv2, co jest najczęściej spowodowane błędną konfiguracją lub nieprawidłowymi ustawieniami domyślnymi. Ten sam certyfikat klucza prywatnego jest używany na innym serwerze, który umożliwia połączenie SSLv2, a także sprawia, że serwer TLS jest podatny na ataki, ponieważ serwer SSLv2 może ujawnić kluczowe informacje.

Atak DROWN umożliwia atakującemu odszyfrowanie ostatniego połączenia TLS między klientem ofiary a serwerem poprzez uruchomienie złośliwych sond SSLv2 przy użyciu tego samego klucza prywatnego. Używając tego atakujący może również zmusić zaatakowanego klienta i serwer do korzystania z wymiany kluczy RSA. Zatem, atakujący może zakłócać połączenia między najnowszymi przeglądarkami i serwerami, które sprzyjają użyciu najnowszej techniki, tj. wymiana kluczy typu perfect-forward-secret, takie jak DHE i ECDH. Atakujący przeprowadzają atak DROWN w ramach internetowego ataku man-in-the-middle (MITM), łamiąc zaszyfrowane klucze, wąchając lub kradnąc poufne informacje, takie jak hasła i dane konta bankowego oraz uzyskując dostęp do osobistych wiadomości e-mail lub wiadomości. Przeprowadzając ten atak, osoba atakująca może również udawać bezpieczną witrynę internetową, a tym samym przejąć lub zmienić jej zawartość.

Narzędzia do analizy kryptograficznej

Atakujący wykorzystują narzędzia kryptoanalizy do analizy i łamania szyfrów. Niektóre narzędzia do kryptoanalizy omówiono poniżej.

CrypTool

Projekt CrypTool rozwija programy e-learningowe w obszarze kryptografii i kryptoanalizy. Składa się z oprogramowania e-learningowego (CT1, CT2, JCT i CTO).

CrypTool 1(CT1) — jest napisany w C++ i jest programem Windows. Obsługuje klasyczne i nowoczesne algorytmy kryptograficzne (szyfrowanie i deszyfrowanie, generowanie kluczy, bezpieczne hasła, uwierzytelnianie, bezpieczne protokoły itp.). Służy do przeprowadzania kryptoanalizy kilku algorytmów (Vigenere, RSA, AES itp.)

CrypTool 2 (CT2) - Obsługuje wizualne programowanie GUI i wykonywanie kaskad procedur kryptograficznych. Działa pod Windowsem.

JCrypTool (JCT) — umożliwia wszechstronne eksperymenty kryptograficzne w systemach Linux, macOS i Windows. Pozwala także użytkownikom rozwijać i rozszerzać swoją platformę na różne sposoby za pomocą własnych wtyczek kryptograficznych.

CrypTool-Online (CTO) — działa w przeglądarce i zapewnia różnorodne metody szyfrowania oraz narzędzia analityczne.

Oto niektóre dodatkowe narzędzia do kryptoanalizy:

Cryptosense (https://cryptosense.com)

RsaCtfTool (https://github.com)

Msieve (https://sourceforge.net)

CryptoI (https://cryptol.net)

CryptoBench (http://www.oddorio.org)

Narzędzia do deszyfrowania MD5 online

Niektóre internetowe narzędzia deszyfrujące MD5, których można użyć do odszyfrowania wartości skrótu MD5 w celu odkrycia oryginalnej wiadomości, to:

MD5 Decoder (https://www.dcode.fr)

CrackStation (https://crackstation.net)

Md5 Decrypt & Encrypt (https://md5decrypt.net)

md5hashing (https://md5hashing.net)

MD5 Encrypt/Decrypt (https://10015.io)

MD5 Decryption (https://www.md5online.org)

MD5Decrypter (https://www.md5decrypter.com)

OnlineHashCrack (https://www.oniinehashcrack.com)

HashKiller.io (https://hashkiller.io)

Md5.My-Addr.com (http://md5.my-addr.com)

cmd5 (https://www.cmd5.org)

Decrypt MD5 Hash (https://hashtoolkit.com)

Online MD5 Hashed Validator (https://www.javainuse.com)

Decode MD5 Hash (https://md5.web-max.ca)

md5 decoder tool (http://md5.my-addr.com)

Środki zaradcze na ataki kryptograficzne

Atakujący wykorzystują różne metody i techniki kryptoanalizy do łamania kryptosystemów i kradzieży poufnych informacji przesyłanych w sieci. W tej sekcji omówiono niektóre środki zaradcze, które można zastosować, aby zapobiec takim atakom.

Jak bronić się przed atakami kryptograficznymi

Aby zapobiec atakom kryptograficznym, można zastosować następujące środki zaradcze:

Dostęp do kluczy kryptograficznych powinien być nadawany bezpośrednio aplikacji lub użytkownikowi.

IDS powinien zostać wdrożony w celu monitorowania wymiany i dostępu do kluczy.

Hasła i hasła muszą być użyte do zaszyfrowania klucza, jeśli są przechowywane na dysku.

Klucze nie powinny znajdować się w kodzie źródłowym ani plikach binarnych.

W przypadku podpisywania certyfikatów przesyłanie kluczy prywatnych nie powinno być dozwolone.

W przypadku algorytmów symetrycznych dla bezpiecznego systemu, zwłaszcza w przypadku dużych transakcji, preferowany powinien być klucz o długości 168 bitów lub 256 bitów.

W celu szyfrowania protokołów z kluczem symetrycznym należy zaimplementować uwierzytelnianie wiadomości.

W przypadku algorytmów asymetrycznych należy wziąć pod uwagę rozmiar klucza 1536 bitów lub 2048 bitów dla bezpiecznych i wysoce chronionych aplikacji.

W przypadku algorytmów haszujących należy wziąć pod uwagę rozmiar klucza 168 lub 256 bitów.

Należy używać tylko zalecanych narzędzi lub produktów, a nie samodzielnie zaprojektowanych algorytmów lub funkcji kryptograficznych.

Nałożyć limit liczby operacji na klucz.

Dane wyjściowe funkcji skrótu powinny mieć większą długość bitową, co utrudnia odszyfrowanie.

Projektuj aplikacje i protokoły, które pozwalają uniknąć prostych zależności między kluczami szyfrowania, tj. każdy zaszyfrowany klucz powinien być tworzony na podstawie funkcji wyprowadzania klucza (KDF).

Zaktualizuj do najnowszych standardów bezpieczeństwa.

Używaj harmonogramów silnych kluczy, aby ograniczać ryzyko związane z atakami na klucze.

Egzekwuj zabezpieczenia sprzętowe, takie jak sprzętowe moduły zabezpieczeń (HSM), aby zwiększyć bezpieczeństwo klucza kryptograficznego.

Nie używaj jednego klucza kryptograficznego do wielu celów.

Używaj redundantnych systemów kryptograficznych do wielokrotnego szyfrowania danych.

Rozciąganie klucza

Rozciąganie klucza odnosi się do procesów stosowanych w celu wzmocnienia słabego klucza, zwykle poprzez jego wydłużenie. Technika ta pomaga w obronie przed atakami brute-force. Ogólnie rzecz biorąc, hasła lub hasła generowane przez użytkowników końcowych są słabe i przewidywalne. W związku z tym rozciąganie kluczy pomaga specjalistom/użytkownikom ds. bezpieczeństwa zapobiegać takim atakom poprzez wzmacnianie ich haseł. W technice rozciągania klucza początkowy klucz jest podawany jako dane wejściowe do algorytmu, który generuje rozszerzony klucz. Klucz musi być wystarczająco odporny na ataki brute-force. Atakującym bardzo trudno jest przewidzieć ulepszony klucz, ponieważ muszą wypróbować każdą możliwą kombinację klucza lub prawdopodobne

kombinacje ulepszonego klucza. Istnieje wiele funkcji i bibliotek, które w ramach swojego działania wykonują rozciąganie klucza:

Funkcja wyprowadzania klucza oparta na hasłach 2 (PBKDF2) jest częścią PKCS #5 v. 2.01. Stosuje pewną funkcję (taką jak skrót lub HMAC) do hasła lub hasła wraz z solą w celu utworzenia klucza pochodnego.

bcrypt jest używany z hasłami i zasadniczo wykorzystuje wariant algorytmu Blowfish, przekonwertowany na algorytm haszujący, aby zaszyfrować hasło i dodać do niego Salt.

Podsumowanie modułu

W tym module omówiono podstawowe pojęcia kryptograficzne stosowane do ochrony poufnych danych, a także różne rodzaje kryptografii. Opisano również szczegółowo szyfry i różne algorytmy szyfrowania używane do szyfrowania lub deszyfrowania danych. Ponadto zilustrowano różne narzędzia kryptograficzne. Następnie podkreślono znaczenie PKI w szyfrowaniu i szczegółowo omówiono protokoły i narzędzia do szyfrowania wiadomości e-mail. Omówiono również szyfrowanie dysku wraz z różnymi narzędziami do szyfrowania dysku. Ponadto wyjaśniono różne typy metod kryptoanalizy i stosowane metody łamania kodów. Następnie przedstawiono różne typy ataków kryptoanalitycznych oraz narzędzia do kryptoanalizy. Ostatecznie zakończyło się wyjaśnieniem środków zaradczych przeciwko różnym atakom kryptograficznym.