Ministerstwo Cyfryzacji

NARODOWY STANDARD CYBERBEZPIECZEŃSTWA
NSC 800-52 wer. 1.0

1 grudnia 2023

Wskazówki dotyczące wyboru, konfigurowania i wykorzystania implementacji TLS (*Transport Layer Security*)

Publikacja dostępna pod adresem:

Narodowe Standardy Cyberbezpieczeństwa

PREAMBUŁA

Szanowni Państwo,

oddajemy w Państwa ręce zestaw publikacji - Narodowe Standardy Cyberbezpieczeństwa, o których mowa w interwencji 2.1 celu szczegółowego 2 Strategii Cyberbezpieczeństwa Rzeczypospolitej Polskiej na lata 2019 – 2024, Opracowanie i wdrożenie Narodowych Standardów Cyberbezpieczeństwa oraz promowanie dobrych praktyk i zaleceń. Standardy zostały opracowane na podstawie publikacji amerykańskiego National Institute of Science and Technology (NIST) i posiadają mapowanie na obowiązujące w polskim systemie prawnym Polskie Normy, na których oparte jest zarządzanie bezpieczeństwem informacji w podmiotach krajowego systemu cyberbezpieczeństwa.

Standardy stanowią przewodniki metodyczne, które ułatwiają zbudowanie efektywnego systemu zarządzania bezpieczeństwem informacji w oparciu o praktykę stosowaną w tym zakresie w administracji federalnej USA.

Niniejsza publikacja **NSC 800-52, *Wskazówki dotyczące wyboru, konfigurowania i wykorzystania implementacji TLS (Transport Layer Security)***, opracowana została za zgodą National Institute of Science and Technology (NIST) na podstawie specjalnej publikacji NIST SP 800-52, *Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations.*

Przytaczane i cytowane w publikacji przepisy, okólniki, rozporządzenia wykonawcze, dyrektywy, normy, standardy, polityki, memoranda itp. odnoszą się, o ile nie zaznaczono inaczej, do prawodawstwa i rynku amerykańskiego. Jeżeli cytowany fragment ma przełożenie lub odpowiednik w polskim porządku prawnym lub normalizacyjnym, wówczas informacje te wskazane są bezpośrednio w tekście lub w przypisach.

W publikacji posłużono się pojęciami zdefiniowanymi w poradniku źródłowym, na podstawie którego powstały niniejsze zalecenia. W przypadku, gdy tożsame pojęcia zostały zdefiniowane również w powszechnie obowiązujących aktach prawnych lub normatywnych, a ich definicja różni się od tej zamieszczonej w niniejszej

publikacji, wówczas należy stosować sformułowania zawarte w tych aktach / w obiegu prawnym.

Tam, gdzie to było możliwe i nie budziło kontrowersji, nazwy ról i kluczowych uczestników procesu zarządzania ryzykiem zostały podane w języku polskim. Pozostałe role i funkcje zostały przedstawione w języku angielskim.[1] Do wszystkich tych ról / funkcji zastosowano akronimy terminologii angielskiej.

Terminologia angielska i akronimy występujące w publikacji zdefiniowane są w dokumencie NSC 7298, ***Słownik kluczowych pojęć z zakresu cyberbezpieczeństwa.***

Podmioty, urządzenia lub materiały o charakterze komercyjnym prezentowane są w niniejszym dokumencie w celu odpowiedniego opisania procedury lub koncepcji eksperymentalnej. Celem ich wskazania nie jest nakłanianie do korzystania z ww. podmiotów, urządzeń lub materiałów lub ich poparcie. Wskazanie ich nie ma również na celu sugerowania, że te podmioty, materiały lub sprzęt są najlepsze z dostępnych w danej dziedzinie.

[1] Kluczowi uczestnicy zarządzania ryzykiem - patrz NSC 800-18; NSC 800-37, NSC 7298.

WSPÓLNE FUNDAMENTY BEZPIECZEŃSTWA I OCHRONY PRYWATNOŚCI

National Institute of Standards and Technology (NIST) opracował szereg standardów i wytycznych w celu zapewnienia jednolitego podejścia do problematyki bezpieczeństwa informacji i systemów informacyjnych administracji federalnej USA. Podstawową rolę w podejściu do zagadnień związanych z zapewnieniem bezpieczeństwa informacji i systemów informacyjnych oraz ochrony prywatności odgrywa elastyczny i spójny sposób zarządzania ryzykiem związanym z bezpieczeństwem i prywatnością działalności i majątku organizacji, osób fizycznych i państwa. Zarządzanie ryzykiem stanowi podstawę do wdrożenia stosownych zabezpieczeń w systemach informacyjnych, ocenę tych zabezpieczeń, wzajemną akceptację dowodów oceny bezpieczeństwa i ochrony prywatności oraz decyzji autoryzacyjnych. Dzięki jednolitemu podejściu do zarządzania ryzykiem ułatwia także wymianę informacji i współpracę pomiędzy różnymi podmiotami.

NIST kontynuuje współpracę z sektorem publicznym i prywatnym w celu stworzenia map i relacji pomiędzy opracowanymi przez siebie standardami i wytycznymi, a tymi, które zostały opracowane przez inne organizacje (m. in. ISO[2]), co zapewnia zgodność w przypadku, gdy regulacje wymagają stosowania innych standardów.

Publikacje NIST, co do zasady nie są objęte restrykcjami wynikającymi z autorskich praw majątkowych. Są powszechnie dostępne oraz dopuszczone do użytku poza administracją federalną USA. Charakteryzują się pragmatycznym podejściem do zagadnień związanych z bezpieczeństwem informacji i systemów informacyjnych oraz ochrony prywatności, przez co ułatwiają podmiotom opracowanie i eksploatację systemu zarządzania tym bezpieczeństwem.

[2] International Organization for Standardization (ISO) - Międzynarodowa Organizacja Normalizacyjna– organizacja pozarządowa zrzeszająca krajowe organizacje normalizacyjne.

Biorąc pod uwagę wszystkie powyższe aspekty, autorzy niniejszej publikacji polecają opracowania NIST, jako godne zaufania i rekomendują stosowanie ich przez polskie podmioty przy opracowywaniu systemów zarządzania bezpieczeństwem informacji, wdrażaniu zabezpieczeń i ocenie ich działania.

W niniejszej publikacji mogą znajdować się odniesienia do innych opracowywanych przez nas publikacji. Informacje tu zawarte, w tym koncepcje, praktyki i metodologie, mogą być wykorzystywane przez organizacje jeszcze przed ukończeniem innych towarzyszących temu standardowi publikacji. W związku z tym, do czasu ukończenia każdej publikacji powinny obowiązywać dotychczasowe wymagania, wytyczne i procedury, jeśli takie istnieją. W ramach planowanych przez Państwa prac zalecamy śledzenie naszych prac publikacyjnych.

W celu odpowiedniego opisania eksperymentalnej procedury lub koncepcji w dokumencie mogą zostać zidentyfikowane pewne podmioty, urządzenia lub materiały komercyjne. Taka identyfikacja nie stanowi rekomendacji, poparcia ani nie ma na celu sugerowania, że dane podmioty, materiały lub urządzenia są bezwzględnie najlepsze z dostępnych dla osiągnięcia danego celu.

Aktualne informacje o prowadzonych przez nas pracach dostępne są pod adresem:

 Narodowe Standardy Cyberbezpieczeństwa

Jesteśmy również otwarci na wszelkie Państwa sugestie, które pomogą nam w dalszych pracach nad standardami cyberbezpieczeństwa i zachęcamy do kontaktu.

 +48222455922

 sekretariat.dc@cyfra.gov.pl

Spis treści

Spis tabel

STRESZCZENIE

Protokół TLS (*ang. Transport Layer Security*) zapewnia mechanizmy ochrony danych podczas elektronicznego przesyłania ich przez Internet. Niniejsza publikacja zawiera wytyczne dotyczące wyboru i konfiguracji implementacji protokołu TLS przy jednoczesnym efektywnym wykorzystaniu algorytmów kryptograficznych zgodnych z federalnymi standardami przetwarzania informacji (*ang. Federal Information Processing Standards – FIPS*) oraz zalecanych przez NIST. Wymaga ona, aby protokół TLS 1.2 skonfigurowany do obsługi zestawu szyfrowania opartego na standardzie FIPS był obsługiwany przez wszystkie serwery sektora publicznego i aplikacje klienckie TLS oraz wymaga wprowadzenia obsługi protokołu TLS 1.3 do 1 stycznia 2024 r. Niniejsza publikacja zawiera również wskazówki dotyczące certyfikatów i rozszerzeń protokołu TLS, które mają wpływ na bezpieczeństwo.

SŁOWA KLUCZOWE

bezpieczeństwo informacji (*ang. information security*); bezpieczeństwo sieci (*ang. network security*); SSL; TLS; Transport Layer Security

ODBIORCY

W niniejszym dokumencie zakłada się, że czytelnik jest zaznajomiony z protokołami TLS i koncepcją infrastruktury klucza publicznego, w tym np. certyfikatami X.509.

Rekomendacje zawarte w tym dokumencie są skierowane w szczególności do instytucji sektora publicznego. Będą one również użyteczne dla innych czytelników, jednak informacje o zalecanych algorytmach kryptograficznych mogą nie mieć zastosowania do adresatów spoza organizacji sektora publicznego, jeśli są one niezgodne z polityką ich organizacji.

STRESZCZENIE

Okólnik A-130 Biura Zarządzania i Budżetu (*ang. Office of Management and Budget – OMB*), *Managing Information as a Strategic Resource*, wymaga od kierowników publicznie dostępnych repozytoriów informacji lub systemów rozpowszechniania, które zawierają dane wrażliwe, ale nieobjęte klauzulą tajności, zapewnienia, że dane wrażliwe są chronione współmiernie do ryzyka i skali szkód, które mogłyby wyniknąć z utraty, niewłaściwego wykorzystania, nieupoważnionego dostępu do tych danych lub ich modyfikacji. Ze względu na specyfikę połączonych sieci i wykorzystanie Internetu do wymiany informacji, ochrona tych wrażliwych danych może stać się trudna, jeśli nie zostaną zastosowane odpowiednie mechanizmy ich zabezpieczenia. Protokół TLS (*ang. Transport Layer Security*) zapewnia takie mechanizmy ochrony wrażliwych danych podczas elektronicznego przesyłania ich przez Internet.

TLS to protokół stworzony w celu zapewnienia uwierzytelnienia, poufności i ochrony integralności danych pomiędzy dwoma komunikującymi się aplikacjami. TLS jest oparty na starszym protokole o nazwie Secure Sockets Layer w wersji 3.0 (SSL 3.0) i jest uważany za jego ulepszenie. Protokół SSL 3.0 został opisany w dokumencie [32]. Protokół Transport Layer Security w wersji 1 (TLS 1.0) został opisany w dokumencie *Request for Comments* (RFC) 2246 [23]. Każdy z tych dokumentów opisuje podobny protokół, który zapewnia usługi bezpieczeństwa w Internecie. Protokół TLS 1.0 został zmieniony do wersji 1.1, co udokumentowano w RFC 4346 [24], a TLS 1.1 został następnie poprawiony do wersji 1.2, co udokumentowano w RFC 5246 [25]. Dodatkowo wprowadzono pewne rozszerzenia, aby zniwelować niektóre znane luki bezpieczeństwa w implementacjach wykorzystujących protokół TLS w wersjach 1.0, 1.1 i 1.2. Protokół TLS 1.3, opisany w dokumencie RFC 8446 [57], jest znaczącą aktualizacją poprzednich wersji, zawierającą zabezpieczenia przed zagrożeniami, które pojawiły się w poprzednich wersjach TLS.

Niniejsza publikacja zawiera wytyczne dotyczące wyboru i konfiguracji implementacji protokołu TLS przy jednoczesnym efektywnym wykorzystaniu zatwierdzonych przez NIST schematów i algorytmów kryptograficznych. W szczególności wymaga ona, aby protokół TLS 1.2 był skonfigurowany do obsługi zestawu szyfrowania, w którym

wykorzystano zatwierdzone przez NIST schematy i algorytmy jako minimalny właściwy bezpieczny protokół transportowy, oraz wymaga wprowadzenia protokołu TLS 1.3 do dnia 1 stycznia 2024 r.[3] Jeśli konieczne jest współdziałanie z systemami innymi niż systemy organizacji sektora publicznego, można wykorzystywać protokoły TLS 1.1 i TLS 1.0. Niniejsza publikacja opisuje również rozszerzenia protokołu TLS, których wykorzystanie jest obowiązkowe, a także wskazuje inne zalecane rozszerzenia.

Stosowanie zaleceń zawartych w niniejszej publikacji ma na celu promowanie:

- bardziej spójnego stosowania mechanizmów uwierzytelniania, poufności i integralności do ochrony informacji przesyłanych przez Internet;
- konsekwentnego stosowania zalecanych zestawów szyfrowania, w których wykorzystano algorytmy zatwierdzone przez NIST oraz otwarte standardy;
- ochrony przed znanymi i przewidywanymi atakami na protokół TLS; oraz
- podejmowanie świadomych decyzji przez administratorów i osoby zarządzające systemami organizacji sektora publicznego w zakresie integracji implementacji protokołu TLS.

Wytyczne te są przeznaczone przede wszystkim dla użytkowników i administratorów systemów sektora publicznego, aby odpowiednio chronić wrażliwe, ale nieobjęte klauzulą tajności dane przed poważnymi zagrożeniami w Internecie. Mogą być również stosowane w zamkniętych środowiskach sieciowych do segregowania danych (omawiany model klient-serwer i usługi bezpieczeństwa mają również zastosowanie w takich sytuacjach). Niniejsza publikacja powinna być stosowana w połączeniu z istniejącymi politykami i procedurami.

[3] Protokół SSL 3.0 jest najbezpieczniejszą wersją protokołu SSL, jednak nie jest ona zatwierdzona do stosowania w ochronie informacji rządowych USA, ponieważ częściowo opiera się na wykorzystaniu algorytmów kryptograficznych, które nie są zatwierdzone przez NIST. Protokół TLS 1.2 jest zatwierdzony do ochrony informacji rządowych, pod warunkiem, że jest odpowiednio skonfigurowany. Protokół TLS w wersji 1.1 i 1.0 jest zatwierdzony tylko wtedy, gdy jest konieczny do zapewnienia współdziałania z systemami innymi niż rządowe i jest skonfigurowany zgodnie z niniejszymi wytycznymi. W polskim porządku prawnym nie ma obowiązku stosowania tego protokołu. Jednakże jest to jedno z najbezpieczniejszych rozwiązań kryptograficznych stosowanych do ochrony informacji wrażliwych.

1. WPROWADZENIE

Protokoły *Transport Layer Security* (TLS) są wykorzystywane do zabezpieczania komunikacji w wielu transakcjach internetowych, takich jak transakcje finansowe (np. bankowość, handel akcjami, handel elektroniczny), transakcje związane z opieką zdrowotną (np. przeglądanie dokumentacji medycznej lub planowanie wizyt lekarskich) oraz transakcje społecznościowe (np. poczta elektroniczna lub sieci społecznościowe). Każda usługa sieciowa, która przetwarza wrażliwe lub cenne dane, niezależnie od tego, czy są to dane osobowe (*ang. Personally Identifiable Information – PII*), finansowe czy informacje o logowaniu, musi je odpowiednio chronić. Protokół TLS zapewnia chroniony kanał do przesyłania danych między serwerem a klientem. Klientem jest często, choć nie zawsze, przeglądarka internetowa.

Zgodnie z memorandum M-15-13[4] wszystkie publicznie dostępne strony internetowe i usługi internetowe organizacji sektora publicznego muszą być dostępne wyłącznie za pośrednictwem bezpiecznego połączenia[5]. Inicjatywa zabezpieczenia połączeń poprawi prywatność i zapobiegnie modyfikacji danych z witryn rządowych podczas ich przesyłania.

TLS jest protokołem warstwowym, który działa jako górna warstwa wiarygodnego protokołu transportowego – zazwyczaj jest to Transmission Control Protocol (TCP). Protokoły aplikacji, takie jak Hypertext Transfer Protocol (HTTP) i Internet Message Access Protocol (IMAP), mogą działać jako kolejna warstwa, nad protokołem TLS. Protokół TLS jest niezależny od aplikacji i służy do zapewnienia bezpieczeństwa dowolnym dwóm komunikującym się aplikacjom, które przesyłają dane przez sieć za pomocą protokołu aplikacji.

1.1 HISTORIA PROTOKOŁU TLS

Protokół Secure Sockets Layer (SSL) został zaprojektowany przez firmę Netscape Corporation w celu zaspokojenia potrzeb w zakresie bezpieczeństwa związanych

[4] https://obamawhitehouse.archives.gov/sites/default/files/omb/memoranda/2015/m-15-13.pdf

[5] Więcej szczegółowych informacji na temat tej inicjatywy można znaleźć na stronie https://https.cio.gov/.

z aplikacjami klienckimi i serwerowymi. Wersja 1 protokołu SSL nigdy nie została wydana. Wersja SSL 2.0 została wydana w 1995 roku, ale była obarczona dobrze znanymi lukami w zabezpieczeniach, które zostały usunięte w wydanej w 1996 roku wersji SSL 3.0. W tym czasie firma Microsoft Corporation wydała protokół znany jako Private Communications Technology (PCT), a później bardziej wydajny protokół znany jako Secure Transport Layer Protocol (STLP). Protokoły PCT i STLP nigdy nie zdobyły takiego udziału w rynku jak SSL 2.0 i SSL 3.0. Grupa Robocza ds. Inżynierii Internetowej (*ang. Internet Engineering Task Force – IETF*), techniczna grupa robocza odpowiedzialna za rozwój standardów internetowych w celu zapewnienia kompatybilności komunikacyjnej pomiędzy różnymi implementacjami, próbowała rozwiązać problemy związane z inżynierią bezpieczeństwa i niekompatybilnością protokołów w najlepszy możliwy sposób. W ramach standardów IETF opisano protokół Transport Layer Security w wersji 1.0 (TLS 1.0), który został skodyfikowany przez IETF w dokumencie *Request for Comments* (RFC) 2246 [23]. Protokół TLS 1.0 bazuje na SSL 3.0 i nie ma między nimi wielkich różnic, jednak są one na tyle istotne, że protokoły te nie współdziałają ze sobą.

Protokół TLS 1.1, określony w dokumencie RFC 4346 [24], został opracowany w celu usunięcia luk wykrytych w wersji TLS 1.0, przede wszystkim w zakresie wyboru wektora inicjującego i przetwarzania błędów dopełniania (*ang. padding error*). Wektory inicjujące stały się widoczne[6], aby zapobiec pewnej klasie ataków na tryb wiązania bloków zaszyfrowanych (*ang. Cipher Block Chaining – CBC*) wykorzystywany w ramach protokołu TLS. Obsługa błędów dopełniania została zmieniona, aby traktowane błąd uzupełniania jako błędny kod uwierzytelniający komunikatu, a nie jak błąd deszyfrowania. Ponadto, w ramach protokołu TLS 1.1 RFC uwzględniono ataki na tryb CBC, które wykorzystują czas obliczania kodu uwierzytelniania komunikatu (*ang. Message Authentication Code – MAC*). W specyfikacji protokołu TLS 1.1 stwierdzono, że aby bronić się przed takimi atakami, implementacja musi przetwarzać rekordy w ten

[6] Wektor inicjujący (*ang. Initialization Vector – IV*) musi zostać wysłany; nie można go uzyskać ze stanu znanego przez obie strony, np. z poprzedniego komunikatu.

sam sposób niezależnie od tego, czy występują błędy dopełniania. Dalsze uwagi dotyczące implementacji związane z trybami CBC (które nie zostały uwzględnione w dokumencie RFC 4346 [24]) zostały omówione w podrozdziale 3.3.2.

W ramach protokołu TLS 1.2, określonego w dokumencie RFC 5246 [25], wprowadzono kilka ulepszeń w zakresie kryptografii, w szczególności w obszarze funkcji skrótu, umożliwiając wykorzystanie lub określenie rodziny algorytmów SHA-2 do obliczania funkcji skrótu, MAC i funkcji pseudolosowych (*ang. Pseudorandom Function – PRF*). W wersji TLS 1.2 dodano również uwierzytelnione szyfrowanie z powiązanymi danymi (*ang. Authenticated Encryption with Associated Data – AEAD*).

Wersja TLS 1.3, opisana w dokumencie RFC 8446 [57], zawiera wiele istotnych zmian w protokole TLS. Ma na celu przeciwdziałanie zagrożeniom, które pojawiły się na przestrzeni ostatnich lat. Wśród wprowadzonych zmian jest nowy protokół uzgodnienia, nowy proces wyprowadzania klucza, który wykorzystuje opartą na HMAC funkcję wyprowadzania klucza typu extract-and-expand (HKDF) [37]. Usunięto również zestawy szyfrowania, które wykorzystują transport klucza RSA lub statyczną wymianę klucza opartą na protokole Diffiego-Hellmana (*ang. Diffie- Hellman – DH*), tryb pracy CBC lub SHA-1. Wiele rozszerzeń zdefiniowanych do użytku z protokołem TLS 1.2 i poprzednimi wersjami nie może być używanych z TLS 1.3.

1.2 ZAKRES

Bezpieczeństwo nie jest pojedynczą właściwością, którą posiada (lub nie) protokół. Bezpieczeństwo to złożony zestaw powiązanych właściwości, które razem zapewniają wymagane cechy wiarygodności informacji i usługi ochrony informacji. Wymagania dotyczące bezpieczeństwa są zwykle określane na podstawie oceny ryzyka zagrożeń lub ataków, które adwersarz może przeprowadzić przeciwko systemowi. Adwersarz prawdopodobnie wykorzysta podatności implementacji znajdujące się w wielu komponentach systemu, w tym komputerowych systemach operacyjnych, systemach oprogramowania użytkowego oraz sieciach komputerowych, które je łączą. Dlatego aby ochronić system przed niezliczoną ilością zagrożeń, zabezpieczenia muszą być umiejętnie rozmieszczone w różnych warstwach systemów i sieci.

Niniejsze zalecenia skupiają się wyłącznie na bezpieczeństwie sieci i koncentrują się bezpośrednio na niewielkiej części stosu komunikacji sieciowej, która jest określana mianem warstwy transportowej. Kilka innych publikacji NIST dotyczy wymagań bezpieczeństwa w pozostałych częściach warstwy systemowej i sieciowej. Przestrzeganie niniejszych wytycznych umożliwia jedynie ochronę danych w trakcie przesyłania. W celu ochrony systemów i przechowywanych danych należy stosować inne obowiązujące normy i rekomendacje NIST.

Zalecenia te koncentrują się na typowych przypadkach użycia, w których aplikacje klienckie i serwerowe muszą współpracować z wieloma różnymi implementacjami, a uwierzytelnianie odbywa się przy użyciu certyfikatów klucza publicznego. Aby umożliwić współdziałanie, implementacje obsługują często szeroki wachlarz opcji kryptograficznych. Istnieją jednak znacznie bardziej ograniczone implementacje TLS, w których wymagane jest bezpieczeństwo, ale szeroka interoperacyjność nie jest konieczna, a koszt wdrożenia nieużywanych funkcji może być zaporowy. Na przykład minimalne serwery są często implementowane we wbudowanych kontrolerach i urządzeniach infrastruktury sieciowej, takich jak routery, a następnie używane z przeglądarkami do zdalnego konfigurowania i zarządzania urządzeniami. Zdarza się również, że zarówno klient, jak i serwer połączenia TLS aplikacji znajdują się pod kontrolą tego samego podmiotu, a zatem wykorzystywanie wielu opcji umożliwiających współdziałanie nie jest konieczne. W takich przypadkach dopuszczalne może być zastosowanie odpowiedniego podzbioru możliwości określonych w niniejszych rekomendacjach.

Zakres jest dodatkowo ograniczony do TLS, gdy jest on stosowany w połączeniu z protokołem TCP/IP. Na przykład Datagram Transport Layer Security (DTLS), który działa w oparciu o protokoły datagramowe, jest poza zakresem tych wytycznych. NIST może wydać osobne wytyczne dla protokołu DTLS w późniejszym terminie.

1.2.1 ALTERNATYWNE KONFIGURACJE

Protokół TLS może być wykorzystywany do zabezpieczenia komunikacji wielu różnych aplikacji w zróżnicowanych środowiskach operacyjnych. Dlatego nie istnieje jedna konfiguracja, która sprawdzi się w przypadku wszystkich scenariuszy. Niniejsze

rekomendacje są próbą sformułowania zaleceń o charakterze ogólnym. Jednak potrzeby konkretnej instytucji sektora publicznego lub aplikacji mogą się różnić od potrzeb ogólnych. **Odstępstwa od tych wskazówek są dopuszczalne, pod warunkiem że podmioty sektora publicznego i administratorzy systemów ocenią i zaakceptują ryzyko związane z alternatywnymi konfiguracjami zarówno pod względem bezpieczeństwa, jak i współdziałania.**

1.3 KONWENCJE STOSOWANE W DOKUMENCIE

W całym niniejszym dokumencie do identyfikacji wymogów używane są słowa kluczowe. Używa się słów kluczowych „**powinien**", „**nie powinien**", „**należy**" i „**nie należy**". Słowa te są podzbiorem słów kluczowych określonych w dokumencie *Request for Comments* (RFC) 2119 IETF i zostały wybrane w oparciu o konwencję występującą w innych dokumentach normatywnych [15]. Oprócz słów kluczowych, w niniejszym dokumencie używane są słowa „musi" i „może", ale nie mają one charakteru normatywnego. Słowa kluczowe „zatwierdzony przez NIST" i „zalecany przez NIST" są używane do wskazania, że schemat lub algorytm jest opisany w federalnym standardzie przetwarzania informacji (*ang. Federal Information Processing Standard – FIPS*) lub jest zalecany przez NIST w publikacji specjalnej (*ang. Special Publication – NIST SP*).

Zalecenia w tym dokumencie są pogrupowane na zalecenia dotyczące serwera i zalecenia dotyczące klienta. Rozdział 3 zawiera szczegółowe wytyczne dotyczące wyboru i konfiguracji serwerów TLS. Rozdział 4 zawiera szczegółowe wytyczne dotyczące wyboru, konfiguracji i użytkowania klientów TLS.

2. PRZEGLĄD PROTOKOŁU TLS

Protokół TLS umożliwia wymianę rekordów za pomocą protokołu rekordów TLS. Rekord TLS zawiera kilka pól, w tym informacje o wersji, dane protokołu aplikacji oraz protokół wyższego poziomu używany do przetwarzania danych aplikacji. Protokół TLS chroni dane aplikacji poprzez zastosowanie zestawu algorytmów kryptograficznych w celu zapewnienia poufności, integralności i autentyczności wymienianych danych aplikacji. Definiuje kilka protokołów do zarządzania połączeniami, które znajdują się nad protokołem rekordów, a każdy protokół ma swój własny typ rekordu. Protokoły te, omówione w podrozdziale 2.1, służą do ustalania i zmiany parametrów bezpieczeństwa oraz do przekazywania serwerowi i klientowi informacji o błędach i ostrzeżeniach. W podrozdziałach od 2.2 do 2.6 opisano usługi bezpieczeństwa zapewniane przez protokół TLS oraz sposób ich realizacji. W podrozdziale 2.7 omówiono zarządzanie kluczami.

2.1 PODPROTOKOŁY TLS

W protokole TLS istnieją trzy podprotokoły, które służą do sterowania połączeniem w ramach sesji: protokół uzgodnienia, zmiany specyfikacji szyfru oraz protokół alarmowy. Protokół uzgodnienia TLS służy do negocjowania parametrów sesji. Protokół alarmowy służy do powiadomienia drugiej strony o stanie błędu. Protokół zmiany specyfikacji szyfru jest wykorzystywany w wersjach TLS 1.0, 1.1 i 1.2 do zmiany parametrów kryptograficznych sesji. Ponadto klient i serwer wymieniają dane aplikacji, które są chronione przez usługi bezpieczeństwa zapewnione przez wynegocjowany zestaw szyfrowania. Te usługi bezpieczeństwa są negocjowane i ustanawiane w ramach protokołu uzgodnienia.

Protokół uzgodnienia polega na wymianie serii komunikatów między klientem a serwerem. Protokół uzgodnienia inicjuje wykorzystanie możliwości kryptograficznych zarówno przez klienta, jak i serwer poprzez negocjowanie zestawu algorytmów i funkcji szyfrujących, w tym algorytmów ustanawiania kluczy, podpisu cyfrowego, poufności i integralności. Klienci i serwery mogą być skonfigurowane tak, aby podczas wykonywania protokołu uzgodnienia negocjowana była jedna lub więcej z następujących usług bezpieczeństwa: poufność, integralność wiadomości, uwierzytelnianie i ochrona przed odtwarzaniem. Usługa poufności, zapobiegając podsłuchowi, daje pewność, że dane są utrzymywane w tajemnicy. Usługa integralności

wiadomości zapewnia potwierdzenie wykrycia nieautoryzowanej modyfikacji danych, zapobiegając w ten sposób niewykrytemu usunięciu, dodaniu lub modyfikacji danych. Usługa uwierzytelniania daje pewność co do tożsamości nadawcy i odbiorcy, a tym samym wykrywa fałszerstwa. Ochrona przed odtwarzaniem zapewnia, że nieautoryzowany użytkownik nie przechwyci i nie zdoła odtworzyć poprzednich danych. Aby zastosować te wytyczne, zarówno klient, jak i serwer muszą być skonfigurowane pod kątem usług poufności i integralności danych.

Protokół uzgodnienia jest używany do opcjonalnej wymiany certyfikatów X.509 z kluczem publicznym[7] w celu uwierzytelnienia serwera wobec klienta i może być również używany do uwierzytelnienia klienta wobec serwera.

Protokół uzgodnienia służy do ustalenia parametrów sesji. Klient i serwer negocjują algorytmy uwierzytelniania, poufności i integralności, a także wyprowadzają klucze symetryczne i ustalają inne parametry sesji, takie jak rozszerzenia. Wynegocjowany zestaw algorytmów kryptograficznych nazywany jest zestawem szyfrowania.

Alerty służą do przekazywania informacji o sesji, takich jak błędy lub ostrzeżenia. Na przykład, alert może być wykorzystany do zasygnalizowania błędu deszyfrowania (decrypt_error) lub odmowy dostępu (access_denied). Niektóre alerty stanowią ostrzeżenia, a inne są uznawane za krytyczne i prowadzą do natychmiastowego zakończenia sesji. Komunikat alertu close_notifytls jest używany do sygnalizowania normalnego zakończenia sesji. Podobnie jak wszystkie inne komunikaty po zakończeniu protokołu uzgodnienia, komunikaty alertów są szyfrowane (i opcjonalnie kompresowane w wersjach protokołu TLS wcześniejszych niż TLS 1.3).

Szczegóły protokołu uzgodnienia, zmiany specyfikacji szyfru (w wersjach TLS wcześniejszych niż 1.3) i protokołów alarmowych są poza zakresem niniejszych wytycznych. Zostały one opisane w dokumentach RFC 5246 [25] i RFC 8446 [57].

[7] W niniejszych dokumencie terminy „certyfikat” i „certyfikat klucza publicznego” są używane zamiennie.

2.2 NEGOCJACJA WSPÓLNEGO KLUCZA TAJNEGO

W trakcie wykonywania protokołu uzgodnienia TLS klient i serwer ustalają klucz. Wyprowadzenie wstępnego klucza tajnego (sekretu) zależy od uzgodnionej metody wymiany kluczy i użytej wersji protokołu TLS. Na przykład, gdy jako algorytm wymiany kluczy w TLS 1.2 i wcześniejszych wersjach używany jest protokół Diffiego-Hellmana, klient i serwer przesyłają sobie nawzajem swoje parametry, które są następnie wykorzystywane do obliczenia wstępnego klucza tajnego. Wstępny klucz tajny, wraz z losowymi wartościami wymienianymi przez klienta i serwer w komunikatach hello, jest wykorzystywany w funkcji pseudolosowej (*ang. pseudorandom function - PRF*) do obliczenia głównego klucza tajnego. W protokole TLS 1.3 główny klucz tajny jest wyprowadzany przez iteracyjne wywoływanie funkcji extract-then-expand z wcześniej wyprowadzonymi kluczami tajnymi. Główny klucz tajny służy do wyprowadzenia kluczy sesji, które są wykorzystywane przez wynegocjowane usługi bezpieczeństwa do ochrony danych wymienianych między klientem a serwerem, zapewniając w ten sposób bezpieczny kanał komunikacji między nimi.

Proces ustalania takich kluczy tajnych jest zabezpieczony przed podsłuchiwaniem. Jeśli protokół TLS jest używany zgodnie z niniejszymi wytycznymi, dane aplikacji, a także klucze tajne nie są narażone na ataki osób, które znajdą się między połączonymi stronami. Napastnik nie może zmodyfikować komunikatu protokołu ustalenia bez wykrycia przez klienta i serwer, ponieważ komunikat *Finished*, wymieniany po ustaleniu parametrów bezpieczeństwa, zapewnia ochronę integralności całej wymiany. Innymi słowy, atakujący nie może zmodyfikować ani obniżyć poziomu bezpieczeństwa połączenia, ustawiając się między negocjującymi stronami.

2.3 POUFNOŚĆ

Poufność sesji komunikacyjnej zapewnia wynegocjowany algorytm zestawu szyfrowania oraz klucze szyfrujące wyprowadzone z głównego klucza tajnego i wartości losowych - jeden do szyfrowania przez klienta (klucz zapisu klienta) i drugi do szyfrowania przez serwer (klucz zapisu serwera). Nadawca komunikatu (klient lub serwer) szyfruje wiadomość, używając odpowiedniego wyprowadzonego klucza szyfrującego; odbiorca używa tego samego (niezależnie wyprowadzonego) klucza do odszyfrowania komunikatu. Zarówno klient, jak

i serwer znają te klucze i odszyfrowują komunikaty, używając tego samego klucza, który został użyty do szyfrowania.

2.4 INTEGRALNOŚĆ

Algorytm MAC z kluczem, określony w wynegocjowanym zestawie szyfrowania, zapewnia integralność komunikatu. Podobnie jak w przypadku poufności, dla każdego kierunku komunikacji istnieje odrębny klucz. Nadawca komunikatu (klient lub serwer) oblicza MAC dla komunikatu używając odpowiedniego klucza MAC (klucz tajny zapisu MAC klienta lub klucz tajny zapisu MAC serwera). Gdy odbiorca przetwarza komunikat, oblicza własną wersję kodu MAC, używając algorytmu MAC i klucza MAC zapisu nadawcy. Odbiorca sprawdza, czy obliczony przez niego kod MAC odpowiada kodowi MAC otrzymanemu w komunikacie od nadawcy.

Dla algorytmów MAC w protokole TLS stosowane są dwa rodzaje konstrukcji. Protokół TLS w wersjach 1.0, 1.1 i 1.2 obsługuje wykorzystanie kodu HMAC (*ang. Keyed-Hash Message Authentication Code*) przy użyciu algorytmu skrótu określonego w wynegocjowanym zestawie szyfrowania. W przypadku kodu HMAC, MAC dla komunikatu od serwera do klienta jest zabezpieczony kluczem MAC zapisu serwera, natomiast MAC dla komunikatu od klienta do serwera jest zabezpieczony kluczem MAC zapisu klienta. Te klucze MAC są wyprowadzane ze wspólnego głównego klucza tajnego.

W wersji TLS 1.2 dodano tryby szyfrowania AEAD, takie jak CCM (*ang. Counter with CBC-MAC*) [41] i GCM (*ang. Galois Counter Mode*) [56, 61], jako alternatywny sposób zapewnienia integralności i poufności. W trybach AEAD nadawca używa swojego klucza zapisu zarówno do szyfrowania, jak i ochrony integralności. Klucze MAC zapisu klienta i serwera nie są używane. Odbiorca odszyfrowuje komunikatu i weryfikuje informację o integralności, używając klucza zapisu nadawcy. W wersji TLS 1.3 do zapewnienia poufności i integralności wykorzystywane są wyłącznie algorytmy symetryczne AEAD.

2.5 UWIERZYTELNIANIE

Uwierzytelnianie serwera jest wykonywane przez klienta przy użyciu certyfikatu klucza publicznego serwera, który serwer prezentuje podczas realizacji protokołu uzgodnienia. Dokładny charakter operacji kryptograficznej uwierzytelniania serwera zależy od

wynegocjowanych parametrów bezpieczeństwa i rozszerzeń. W wielu przypadkach uwierzytelnianie odbywa się jawnie poprzez weryfikację podpisów cyfrowych z wykorzystaniem kluczy publicznych obecnych w certyfikatach lub niejawnie poprzez wykorzystanie klucza publicznego serwera przez klienta podczas ustalania głównego klucza tajnego. Pomyślne wysłanie komunikatu Finished oznacza, że obie strony obliczyły ten sam tajny klucz główny, a więc serwer musiał znać klucz prywatny odpowiadający kluczowi publicznemu w certyfikacie serwera.

Uwierzytelnienie klienta jest opcjonalne i odbywa się tylko na żądanie serwera. Uwierzytelnienie klienta opiera się na jego certyfikacie klucza publicznego. Dokładny charakter operacji kryptograficznej uwierzytelniania klienta zależy od wynegocjowanego algorytmu wymiany kluczy w zestawie szyfrowania oraz wynegocjowanych rozszerzeń. Na przykład, gdy certyfikat klucza publicznego klienta zawiera klucz publiczny RSA, klient podpisuje część komunikatu w ramach protokołu uzgodnienia przy użyciu klucza prywatnego odpowiadającego temu kluczowi publicznemu, a serwer weryfikuje podpis przy użyciu klucza publicznego klienta, aby go uwierzytelnić.

2.6 OCHRONA PRZED ODTWARZANIEM

Protokół TLS zapewnia ochronę przed odtwarzaniem, z wyjątkiem sytuacji, gdy w ramach wykonywania protokołu TLS 1.3[8] wysyłane są dane 0-RTT (opcjonalnie w pierwszych komunikatach w ramach protokołu uzgodnienia). Koperta komunikatu z zabezpieczeniem integralności zawiera monotonicznie rosnący numer kolejny. Po sprawdzeniu integralności komunikatu numer kolejny bieżącego komunikatu jest porównywany z numerem kolejnym poprzedniego komunikatu. Numer kolejny bieżącego komunikatu musi być większy niż numer kolejny poprzedniego komunikatu, aby komunikat był dalej przetwarzany.

[8] Protokół TLS 1.3 nie ma wbudowanego zabezpieczenia przed atakami umożliwiającymi odtwarzanie w ramach wysyłania danych 0-RTT, jednak w jego specyfikacji zalecane są mechanizmy chroniące przed takimi atakami (patrz rozdział 8 [57]).

2.7 ZARZĄDZANIE KLUCZAMI

Bezpieczeństwo klucza prywatnego serwera jest krytyczne dla bezpieczeństwa protokołu TLS. Jeśli klucz prywatny serwera jest słaby lub może być uzyskany przez stronę trzecią, strona trzecia może podszywać się pod serwer przed wszystkimi klientami. Podobnie, jeśli strona trzecia może uzyskać certyfikat klucza publicznego dla klucza publicznego odpowiadającego jej własnemu kluczowi prywatnemu w nazwie legalnego serwera od urzędu certyfikacji (*ang. Certification Authority – CA*) uznanego za zaufany przez klientów, strona trzecia może podszywać się pod serwer dla klientów. Wymagania i zalecenia mające na celu przeciwdziałanie takim działaniom zostały omówione w dalszej części niniejszych rekomendacji.

Podobne zagrożenia istnieją w przypadku klientów. Jeśli klucz prywatny klienta jest słaby lub może być uzyskany przez stronę trzecią, strona trzecia może podszywać się pod klienta przed wszystkimi serwerami. Podobnie, jeśli strona trzecia może uzyskać certyfikat klucza publicznego dla klucza publicznego odpowiadającego jej własnemu kluczowi prywatnemu w nazwie klienta od urzędu certyfikacji (CA), do którego zaufanie ma serwer, strona trzecia może podszywać się pod klienta wobec serwera. Wymagania i zalecenia mające na celu przeciwdziałanie takim działaniom zostały omówione w dalszej części niniejszych wytycznych.

Ponieważ liczby losowe generowane przez klienta i serwer wpływają na losowość kluczy sesji, klient i serwer muszą być zdolne do generowania liczb losowych o poziomie bezpieczeństwa wynoszącym co najmniej 112[9] bitów każda. Różne klucze sesji w ramach protokołu TLS wyprowadzone z tych losowych wartości i innych danych są ważne przez czas trwania sesji. Klucze sesji są używane tylko do ochrony komunikatów wymienianych podczas aktywnej sesji TLS, nie są używane do ochrony jakichkolwiek przechowywanych danych, dlatego nie ma konieczności odzyskiwania kluczy sesji TLS. Mimo to wszystkie wersje protokołu TLS zapewniają mechanizmy

[9] Więcej informacji na temat generatorów bitów losowych można znaleźć w publikacji z serii NIST SP 800-90 (https://csrc.nist.gov/projects/random-bit-generation).

przechowywania klucza związanego z sesją, które pozwalają na wznowienie sesji w przyszłości. Klucze dla wznowionej sesji są wyprowadzane podczas wykonywania skróconego protokołu uzgodnienia, w ramach którego przechowywany klucz jest wykorzystywany jako forma uwierzytelnienia.

3. MINIMALNE WYMAGANIA DLA SERWERÓW TLS

W tym rozdziale przedstawiono minimalny zestaw wymagań, które serwer musi spełnić, aby zapewnić zgodność z niniejszymi rekomendacjami. Wymagania zostały uporządkowane w podrozdziałach o następującej tematyce: obsługiwane wersje protokołu TLS, klucze i certyfikaty serwera, obsługa kryptografii, obsługa rozszerzeń protokołu TLS, uwierzytelnianie klienta, wznawianie sesji, metody kompresji oraz aspekty operacyjne.

Szczegółowe wymagania zostały określone jako wymagania dotyczące wdrożenia lub wymagania dotyczące konfiguracji. Zgodnie z wymaganiami dotyczącymi wdrożenia organizacje **nie powinny** zamawiać implementacji serwera TLS, chyba że obejmuje ona wymagane funkcje lub może być rozszerzona o dodatkowe produkty komercyjne w celu spełnienia wymagań. Zgodnie z wymaganiami dotyczącymi konfiguracji, administratorzy serwerów TLS są zobowiązani do sprawdzenia, czy poszczególne funkcje są włączone lub wyłączone, a w niektórych przypadkach odpowiednio skonfigurowane, jeśli są dostępne.

3.1 OBSŁUGIWANE WERSJE PROTOKOŁU

Serwery obsługujące wyłącznie aplikacje rządowe[10] **należy** skonfigurować tak, aby wykorzystywały protokół TLS 1.2, a także TLS 1.3. Serwerów **nie należy** konfigurować do używania protokołu TLS 1.1 i **nie należy** wykorzystywać protokołu TLS 1.0, SSL 3.0 ani SSL 2.0. Wersje TLS 1.2 i 1.3 są oznaczane za pomocą krotek nadrzędnych i podrzędnych (3, 3) i (3, 4), i mogą występować w tym formacie podczas konfiguracji[11].

Serwery obsługujące aplikacje skierowane do obywateli lub organizacji (tzn. klient nie jest częścią rządowego systemu informacyjnego)[12] **należy** skonfigurować do negocjowania korzystania z protokołu TLS 1.2, a także **powinny** być skonfigurowane do

[10] Aplikacja przeznaczona wyłącznie dla rządu to aplikacja, której planowanymi użytkownikami są wyłącznie pracownicy rządowi lub wykonawcy pracujący dla rządu. Dotyczy to aplikacji, do których dostęp uzyskuje się w ramach systemu „przynieś własne urządzenie" (*ang. Bring Your Own Device – BYOD*) dla pracowników rządowych.

[11] Wcześniej oznaczeniem wersji TLS 1.0 były dwie liczby. Pierwsza oznaczała wersję główną, a druga — wydanie w ramach wersji (3,1). W ten sposób zrównano ją z wersją protokołu SSL 3.1. Wersja TLS 1.1 jest oznaczona parą liczb (3,2).

[12] Na potrzeby niniejszego dokumentu klienci, którzy działają w systemach typu „przynieś własne urządzenie" (*ang. Bring Your Own Device – BYOD*) lub w systemach prywatnych wykorzystywanych do wykonywania pracy zdalnej, są uważani za część rządowego systemu informacyjnego, ponieważ uzyskują dostęp do usług, które nie są dostępne publicznie.

negocjowania TLS 1.3. Stosowanie protokołu TLS w wersji 1.1 i 1.0 jest ogólnie odradzane, ale wersje te mogą być skonfigurowane, aby w razie potrzeby umożliwić interakcję z obywatelami i organizacjami. Omówienie kwestii ewentualnej obsługi protokołów TLS 1.0 i TLS 1.1 znajduje się w załączniku F. Serwery tego typu **nie powinny** umożliwiać korzystania z protokołu SSL 2.0 lub SSL 3.0.

Instytucje państwowe **powinny** móc obsługiwać protokół TLS 1.3 do dnia 1 stycznia 2024 r. Po tej dacie serwery **powinny** obsługiwać protokół TLS 1.3 zarówno dla aplikacji przeznaczonych wyłącznie dla rządu, jak i skierowanych do obywateli lub organizacji. Generalnie, serwery obsługujące protokół TLS 1.3 **powinny** być skonfigurowane tak, aby mogły korzystać również z TLS 1.2. Można jednak zablokować możliwość korzystania z TLS 1.2 na serwerach obsługujących TLS 1.3, jeśli ustalono, że wersja TLS 1.2 nie jest konieczna ze względu na interoperacyjność.

Znane są przypadki implementacji serwerów, w których negocjacja wersji jest realizowana niepoprawnie. Na przykład istnieją serwery TLS 1.0, które przerywają połączenie, gdy klient oferuje wersję nowszą niż TLS 1.0. **Nie należy** używać serwerów, w których nieprawidłowo zaimplementowano negocjację wersji TLS.

3.2 KLUCZE I CERTYFIKATY SERWERA

Serwer TLS **należy** skonfigurować z wykorzystaniem jednego lub więcej certyfikatów klucza publicznego i powiązanych kluczy prywatnych. Implementacje serwerów TLS **powinny** obsługiwać korzystanie z wielu certyfikatów serwera wraz z powiązanymi z nimi kluczami prywatnymi, aby zapewniać elastyczność w zakresie algorytmów i rozmiarów kluczy.

Kilka opcji certyfikatów serwera TLS spełnia wymogi w zakresie kryptografii NIST: certyfikat podpisu RSA, algorytm podpisu cyfrowego krzywej eliptycznej (*ang. Elliptic Curve Digital Signature Algorithm – ECDSA*), algorytm podpisu cyfrowego (*ang. Digital Signature Algorithm – DSA*)[13], certyfikat Diffiego-Hellmana (*ang. Diffie-Hellman – DH*) i certyfikat krzywej eliptycznej Diffiego-Hellmana (*ang. Elliptic Curve Diffie-Hellman –*

[13] W nazwach zestawów szyfrowania TLS certyfikat DSA ze względów historycznych określany jest mianem standardu podpisu cyfrowego (*ang. Digital Signature Standard – DSS*).

ECDH). Serwery TLS zgodne z niniejszą specyfikacją **należy** skonfigurować przynajmniej z wykorzystaniem certyfikatu podpisu RSA lub certyfikatu podpisu ECDSA. Pozostałe typy certyfikatów i związane z nimi zestawy szyfrowania nie są powszechnie stosowane, zwłaszcza w przypadku serwerów dostępnych z zewnątrz, ale zostały ujęte w niniejszej publikacji w celu zapewnienia kompleksowych informacji i uwzględnienia skrajnych przypadków. Jeśli serwer jest skonfigurowany z wykorzystaniem certyfikatu podpisu ECDSA, dla klucza publicznego w certyfikacie **należy** użyć krzywej P-256 lub krzywej P-384[14].

Serwery TLS należy konfigurować przy użyciu certyfikatów wydanych przez CA, który publikuje informacje o unieważnieniu w odpowiedziach dotyczących weryfikacji statusu certyfikatów w trybie on-line (*ang. Online Certificate Status Protocol – OCSP*) [63]. CA może dodatkowo publikować informacje o unieważnieniu na liście unieważnionych certyfikatów (*ang. Certificate Revocation List – CRL*) [19]. Źródła informacji o unieważnieniu **powinny** być zawarte w certyfikacie wydanym przez CA w odpowiednim rozszerzeniu w celu promowania interoperacyjności.

Serwer TLS, który otrzymał certyfikaty od wielu CA, może wybrać odpowiedni certyfikat na podstawie określonego przez klienta rozszerzenia protokołu TLS „Klucze od zaufanych CA" (patrz podrozdział 3.4.2.6). Serwer TLS, który otrzymał certyfikaty dla wielu nazw serwera, może wybrać odpowiedni certyfikat na podstawie określonego przez klienta rozszerzenia protokołu TLS „Nazwa serwera" (patrz podrozdział 3.4.1.2). Certyfikat serwera TLS może również zawierać wiele nazw w ramach rozszerzenia dotyczącego alternatywnych nazw podmiotów (*ang. Subject Alternative Name – SAN*), aby umożliwić użycie wielu nazw serwerów o tym samym formacie nazwy, takim jak np. system nazw domen (*ang. Domain Name System – DNS*), lub wielu nazw serwerów o wielu formatach nazwy (np. nazw DNS, adresów IP itd.).

[14] Zalecane krzywe eliptyczne wymienione obecnie w standardzie FIPS 186-4 [45] zostaną przeniesione do publikacji SP 800-186. Do czasu pojawienia się publikacji SP 800-186, przy wyborze zalecanych krzywych eliptycznych należy kierować się standardem FIPS 186-4.

Procesy aplikacji służące do uzyskania certyfikatów różnią się i wymagają różnych poziomów dowodów przy łączeniu certyfikatów z domenami. Wnioskodawca może uzyskać certyfikat potwierdzający domenę (*ang. Domain Validated - DV*), jeśli udowodni, że ma kontrolę nad domeną DNS. Certyfikat potwierdzający organizację (*ang. Organization Validation – OV*) wymaga dalszej weryfikacji. Certyfikat o rozszerzonej walidacji (*ang. Extended Validation – EV*) charakteryzuje się najbardziej szczegółowym procesem weryfikacji tożsamości. Niniejsze zalecenie nie zawiera wytycznych dotyczących tego, jaki poziom weryfikacji należy zastosować.

W podrozdziale 3.2.1 przedstawiono szczegółowy profil certyfikatów serwera. Przedstawiono tam podstawowe wytyczne dotyczące certyfikatów RSA, ECDSA, DSA, DH i ECDH. W podrozdziale 3.2.2 określono wymogi dotyczące kontroli unieważnienia. W podrozdziale 3.5.4 określono wymogi dotyczące „listy wskazówek".

3.2.1 PROFIL CERTYFIKATU SERWERA

Profil certyfikatu serwera, opisany w tym podrozdziale, zawiera wymagania i zalecenia dotyczące formatu certyfikatu serwera. Aby spełnić niniejsze wytyczne, certyfikat serwera TLS **powinien** być certyfikatem X.509 w wersji 3. Zarówno klucz publiczny zawarty w certyfikacie, jak i podpis **powinny** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów. Przed wersją TLS 1.2 komunikat dotyczący certyfikatu (*ang. Certificate*) serwera wymagał, aby algorytm podpisywania certyfikatu był taki sam jak algorytm dla klucza certyfikatu (patrz punkt 7.4.2 publikacji [24]). Jeśli serwer obsługuje wersje protokołu TLS wcześniejsze niż TLS 1.2, certyfikat powinien być podpisany przy pomocy algorytmu zgodnego z kluczem publicznym[15].

- Certyfikaty zawierające klucze publiczne RSA, ECDSA lub DSA **powinny** być podpisane odpowiednio przy pomocy tych samych algorytmów podpisu.
- Certyfikaty zawierające klucze publiczne Diffiego-Hellmana **powinny** być podpisywane za pomocą algorytmu DSA.

[15] Wytyczne dotyczące generowania par kluczy publicznych i prywatnych zależą od stosowanego algorytmu. Wytyczne dotyczące generowania par kluczy DH i ECDH znajdują się w publikacji NIST SP 800-56A [6]. Wytyczne dotyczące generowania par kluczy RSA, DSA i ECDSA znajdują się w publikacji [45].

- Certyfikaty zawierające klucze publiczne ECDH **powinny** być podpisane za pomocą algorytmu ECDSA.

Rozszerzenie umożliwiające wykorzystanie klucza rozszerzonego ogranicza sposób wykorzystania kluczy w certyfikacie. Istnieje klucz przeznaczony specjalnie do uwierzytelniania serwera, a serwer **powinien** być skonfigurowany tak, aby umożliwić jego użycie. Zastosowanie rozszerzenia umożliwiającego wykorzystanie klucza rozszerzonego ułatwi pomyślne uwierzytelnienie serwera, ponieważ niektórzy klienci mogą wymagać obecności rozszerzenia umożliwiającego wykorzystanie klucza rozszerzonego. Użycie nazwy DNS serwera w polu alternatywnej nazwy podmiotu zapewnia, że wszelkie ograniczenia dotyczące nazwy na ścieżce certyfikacji będą prawidłowo egzekwowane.

Profil certyfikatu serwera został przedstawiony w tabeli 3-1. W przypadku braku wymogów dotyczących profilu certyfikatu specyficznych dla danej instytucji państwowej, **należy** stosować niniejszy profil dla certyfikatu serwera.

Tabela 3-1: Profil certyfikatu serwera TLS

Pole	Krytyczność	Wartość	Opis
Wersja	**Nd.**	**2**	**Wersja 3**
Numer seryjny	Nd.	Unikalna dodatnia liczba całkowita	Musi być unikalna
Algorytm podpisu wystawcy	Nd.	Wartości zależne od typu klucza CA:	
		sha256WithRSAEncryption {1 2 840 113549 1 1 11} lub silniejsza	CA z kluczem RSA
		id-RSASSA-PSS {1 2 840 113549 1 1 10}	CA z kluczem RSA
		ecdsa-with-SHA256 {1 2 840 10045 4 3 2} lub silniejsza	CA z kluczem opartym na krzywej eliptycznej
		id-dsa-with-sha256 {2 16 840 1 101 3 4 3 2} lub silniejsza	CA z kluczem DSA
Nazwa wyróżniająca (DN) wystawcy	Nd.	Unikalna nazwa DN CA wystawiającego zgodna ze standardem X.500	W każdej względnej nazwie wyróżniającej (*Relative Distinguished Name – RDN*) **powinna** być zakodowana pojedyncza wartość. Wszystkie atrybuty typu DirectoryString **powinny** być zakodowane jako PrintableString.
Okres ważności	Nd.	3 lata lub mniej	Daty do roku 2049 wyrażone w czasie UTC
Nazwa wyróżniająca podmiotu	Nd.	Unikalna nazwa DN podmiotu zgodna ze standardem X.500 i wymogami danej instytucji państwowej	W każdej nazwie RDN powinna być zakodowana pojedyncza wartość. Wszystkie atrybuty typu DirectoryString powinny być zakodowane jako PrintableString. Jeśli jest obecny, atrybut CN powinien mieć format: CN={adres IP hosta | nazwa DNS hosta}

Pole	Krytyczność	Wartość	Opis
Informacje o kluczu publicznym podmiotu	Nd.	*Wartości według typu certyfikatu:*	
		rsaEncryption {1 2 840 113549 1 1 1}	2048-bitowy moduł klucza RSA certyfikatu podpisu RSA lub inne zatwierdzone długości, jak określono w publikacjach [45] i [5] Parametry: NULL
		ecPublicKey {1 2 840 10045 2 1}	Certyfikat podpisu ECDSA lub certyfikat ECDH Parametry: namedCurve OID dla nazwanej krzywej określonej w publikacji NIST SP 800-186[16]. Powinna to być krzywa P-256 lub P-384. Klucz SubjectPublic: Nieskompresowany punkt EC.
		id-dsa {1 2 840 10040 4 1}	Certyfikat podpisu DSA Parametry: p, q, g (2048-bitowy large prime, odp., p)
		dhpublicnumber {1 2 840 10046 2 1}	Certyfikat DH Parametry: p, g, q (2048-bitowy large prime, tj.., p)
Podpis wystawcy	Nd.	Wartość taka sama jak w przypadku algorytmu podpisu wydawcy	

[16] Zalecane krzywe eliptyczne wymienione obecnie w standardzie FIPS 186-4 [45] zostaną przeniesione do publikacji SP 800-186. Do czasu pojawienia się publikacji SP 800-186, przy wyborze zalecanych krzywych eliptycznych należy kierować się standardem FIPS 186-4.

Pole	Krytyczność	Wartość	Opis
Rozszerzenia			
Identyfikator klucza urzędu	Nie	Octet String	Taki sam jak identyfikator klucza podmiotu w certyfikacie CA będącego wydawcą Zabronione: DN wystawcy, krotka numeru seryjnego
Identyfikator klucza podmiotu	Nie	Octet String	Taki sam jak w standardach kryptografii klucza publicznego (*Public-Key Cryptography Standards – PKCS*) 10 lub obliczony przez wydający CA
Użycie klucza	Tak	*Wartości według typu certyfikatu:*	
		digitalSignature	Certyfikat podpisu RSA, certyfikat podpisu ECDSA lub certyfikat podpisu DSA
		keyAgreement	Certyfikat ECDH, certyfikat DH
Użycie klucza rozszerzonego	Nie	id-kp-serverAuth {1 3 6 1 5 5 7 3 1}	Wymagane
		id-kp-clientAuth {1 3 6 1 5 5 7 3 2}	Opcjonalne
			Zabronione: anyExtendedKeyUsage; wszystkie inne, chyba że są zgodne z użyciem klucza rozszerzonego
Zasady certyfikatu	Nie		Opcjonalne
Alternatywna nazwa podmiotu (*Subject Alternative Name – SAN*)	Nie	Nazwa hosta DNS lub adres IP, jeśli nie ma przypisanej nazwy DNS. W razie potrzeby można uwzględnić inne formy nazwy.	Wymagane. Dopuszcza się stosowanie wielu nazw SAN, np. w przypadku środowisk o zrównoważonym obciążeniu.

Pole	Krytyczność	Wartość	Opis
Dostęp do informacji o urzędzie	Nie	id-ad-caIssuers	Wymagane. Wpis dotyczący metody dostępu zawiera adres URL HTTP certyfikatów wydanych dla CA będącego wydawcą
		Patrz uwagi	Wymagane. Wpis dotyczący metody dostępu zawiera adres URL HTTP obiektu odpowiadającego OCSP wydającego CA
Punkty dystrybucji listy CRL	Nie	Patrz uwagi	Opcjonalne. Wartość HTTP w polu distributionPoint kierująca do pełnej i kompletnej listy CRL.
Lista znaczników czasu podpisanego certyfikatu	Nie		Opcjonalne. Rozszerzenie to zawiera sekwencję znaczników czasu podpisanego certyfikatu, które stanowią dowód, że certyfikat został przekazany do dzienników transparentności certyfikatu
Funkcja TLS	Nie	status_request(5)	Opcjonalne. To rozszerzenie, czasami określane jako rozszerzenie „konieczność zszywania” (*ang. “must staple”*), może być obecne, aby poinformować klientów, że serwer obsługuje zszywanie OCSP i dostarczy zszytą odpowiedź OCSP, gdy pojawi się żądanie.

3.2.2 UZYSKIWANIE INFORMACJI O STANIE UNIEWAŻNIENIA DLA CERTYFIKATU KLIENTA

Serwer **powinien** przeprowadzić sprawdzanie stanu unieważnienia certyfikatu klienta, gdy używane jest uwierzytelnianie klienta. Informacje o unieważnieniu powinny być uzyskane przez serwer z jednej lub kilku z następujących lokalizacji:

1. Lista unieważnień certyfikatów (*ang. Certificate Revocation List – CRL*) lub odpowiedź OCSP [63] w lokalnym magazynie serwera.
2. Odpowiedź OCSP z lokalnie skonfigurowanego obiektu odpowiadającego OCSP.
3. Odpowiedź OCSP z lokalizacji obiektu odpowiadającego OCSP zidentyfikowanej w polu OCSP w rozszerzeniu dotyczącym dostępu do informacji o urzędzie w certyfikacie klienta.
4. Lista CRL z rozszerzenia dotyczącego punktów dystrybucji listy CRL w certyfikacie klienta.

Jeśli w lokalnym magazynie nie ma aktualnej lub wiążącej[17] listy CRL lub odpowiedzi OCSP, a obiekt odpowiadający OCSP i punkt dystrybucji listy CRL są niedostępne lub nieosiągalne w czasie nawiązywania sesji TLS, serwer odmówi połączenia lub zaakceptuje potencjalnie unieważniony lub zagrożony certyfikat. Decyzja o przyjęciu lub odrzuceniu certyfikatu w takiej sytuacji **powinna** być podjęta zgodnie z polityką danej instytucji państwowej.

3.2.3 WIARYGODNOŚĆ CERTYFIKATU KLUCZA PUBLICZNEGO SERWERA

Zasady, procedury i środki bezpieczeństwa, na podstawie których CA wydaje certyfikat z kluczem publicznym, są udokumentowane w zasadach certyfikatów. Stosowanie zasad certyfikatów zaprojektowanych z myślą o bezpiecznym funkcjonowaniu infrastruktury klucza publicznego (*ang. Public Key Infrastucture – PKI*) oraz przestrzeganie ustalonych zasad certyfikatów zmniejsza zagrożenie, że CA wydający certyfikat może zostać narażony na niebezpieczeństwo lub że system rejestracji, osoby lub proces mogą zostać

[17] Lista CRL jest uznawana za „wiążącą", gdy „zakres listy CRL" [19] jest odpowiedni dla danego certyfikatu.

naruszone w celu uzyskania nieautoryzowanego certyfikatu na nazwę legalnego podmiotu i w ten sposób narazić na niebezpieczeństwo klientów. Mając to na uwadze, CA Browser Forum, organizacja z sektora prywatnego, podjęła pewne działania w tym zakresie, tworząc wymagania dla wydawania certyfikatów z publicznie zaufanych CA, aby te CA i ich kotwice zaufania pozostały w magazynach zaufania przeglądarek [16]. W ramach innych działań organizacja CA Browser Forum opracowała wytyczne dotyczące wydawania certyfikatów z rozszerzoną weryfikacją [17].

Trwają prace nad kilkoma koncepcjami, które mają jeszcze bardziej ograniczyć ryzyko związane z naruszeniem systemu, procesu lub personelu CA lub systemu rejestracji certyfikatów X.509. Obejmują one projekt Certificate Transparency (patrz podrozdział 3.4.2.15) i inne pojawiające się koncepcje, które zostały omówione w załączniku E.

Zasady, na podstawie których został wydany certyfikat, mogą być opcjonalnie przedstawione w certyfikacie za pomocą rozszerzenia Certificate Policies, opisanego w publikacji [19] i zaktualizowanego w publikacji [72]. W przypadku stosowania, w ramach tego rozszerzenia podaje się jeden lub więcej identyfikatorów obiektów (*ang. Object Identifier – OID*) zasad certyfikatów, przy czym każdy OID reprezentuje określone zasady certyfikatów. Wiele klientów TLS (np. przeglądarki) nie oferuje jednak możliwości akceptowania lub odrzucania certyfikatów na podstawie zasad, według których zostały one wydane. Dlatego zasadniczo konieczne jest, aby certyfikaty serwera TLS były wydawane przez CA, które wydają certyfikaty wyłącznie w sposób zgodny z zasadami certyfikatów, w których zawarto odpowiednie środki bezpieczeństwa.

Gdy instytucja rządowa uzyskuje certyfikat dla serwera TLS, w przypadku którego wszyscy klienci są pod kontrolą tej instytucji, może ona wystawić certyfikat z własnego CA, jeśli może skonfigurować klientów pod kątem zaufania do tego CA. W innych przypadkach organizacja powinna uzyskać certyfikat od publicznie zaufanego CA (CA, dla którego klienci, którzy będą się łączyć z serwerem, mają już skonfigurowane zaufanie).

3.3 OBSŁUGA KRYPTOGRAFII

Obsługa kryptografii w ramach protokołu TLS jest możliwa dzięki zastosowaniu różnych zestawów szyfrowania. Zestaw szyfrowania określa zbiór algorytmów

do wymiany kluczy (tylko w wersji TLS 1.2 i wcześniejszych)[18] oraz do zapewnienia usług poufności i integralności danych aplikacji. Negocjacja zestawu szyfrowania następuje podczas wykonywania protokołu uzgodnienia TLS. Klient przedstawia serwerowi zestawy szyfrowania, które obsługuje, a serwer wybiera jeden z nich do zabezpieczenia danych sesji.

Oprócz wyboru odpowiednich zestawów szyfrowania, administratorzy mogą również uwzględnić dodatkowe uwagi dotyczące implementacji algorytmów kryptograficznych, a także wymagania dotyczące walidacji modułów kryptograficznych. Dopuszczalne zestawy szyfrowania są wymienione w podrozdziale 3.3.1, pogrupowane według typu certyfikatu i wersji protokołu. Zagadnienia związane z implementacją zestawów szyfrowania omówiono w podrozdziale 3.3.2, a zalecenia dotyczące walidacji modułów kryptograficznych w podrozdziale 3.3.3.

3.3.1 ZESTAWY SZYFROWANIA

Zestawy szyfrowania określają algorytmy kryptograficzne, które będą używane w danej sesji. Zestawy szyfrowania w wersjach protokołu TLS od 1.0 do 1.2 mają postać:

TLS_KeyExchangeAlg_WITH_EncryptionAlg_MessageAuthenticationAlg

Przykładowo, zestaw szyfrowania TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA wykorzystuje proces ustalania efemerycznego klucza ECDH z parametrami podpisanymi przy użyciu RSA, poufność zapewnioną przez AES-128 (zaawansowany standard szyfrowania z kluczem 128-bitowym) w trybie wiązania bloków szyfrogramu oraz uwierzytelnianie komunikatu przy użyciu HMAC_SHA[19]. Więcej informacji na temat interpretacji zestawów szyfrowania znajduje się w załączniku B.

W wersji TLS 1.3 zestawy szyfrowania są definiowane inaczej. Takie zestawy szyfrowania nie określają algorytmu wymiany kluczy i mają postać:

TLS_AEAD _HASH

[18] W protokole TLS 1.3 algorytm wymiany kluczy jest określony wyłącznie w ramach rozszerzeń (patrz podrozdział 3.4.2.3 i 3.4.2.10).

[19] Akronim SHA informuje o zastosowaniu algorytmu skrótu SHA-1.

Na przykład, zestaw szyfrowania TLS_AES_128_GCM_SHA256 wykorzystuje standard AES-128 w trybie Galois/licznika do zapewnienia poufności i uwierzytelnienia komunikatu oraz wykorzystuje SHA-256 do funkcji PRF. Zestawy szyfrowania TLS 1.3 nie mogą być wynegocjowane dla połączeń TLS 1.2, a zestawy szyfrowania TLS 1.2 nie mogą być wynegocjowane dla połączeń TLS 1.3.

Podczas negocjacji zestawu szyfrowania klient wysyła komunikat protokołu uzgodnienia z listą zestawów szyfrowania, które może zaakceptować. Serwer wybiera z listy i wysyła komunikat protokołu uzgodnienia z informacją, który zestaw szyfrów zaakceptuje. Chociaż klient może uporządkować listę wymieniając zestawy szyfrowania, które uważa za najsilniejsze, jako pierwsze, jednak serwer może zignorować tę kolejność i wybrać dowolny z zestawów szyfrowania zaproponowanych przez klienta. Serwer może mieć swoją własną kolejność preferowanych zestawów szyfrowania i może ona być inna niż u klienta. Dlatego *nie ma* gwarancji, że negocjacje zakończą się na najsilniejszym wspólnym zestawie. Jeśli żaden zestaw szyfrowania nie jest wspólny dla klienta i serwera, połączenie jest przerywane.

Serwer **należy** skonfigurować w taki sposób, aby korzystał wyłącznie z zestawów szyfrowania składających się w całości z algorytmów zatwierdzonych przez NIST (tj. [6, 7, 9, 26–28, 44-46, 49]). Pełna lista akceptowalnych zestawów szyfrowania do ogólnego użytku znajduje się w niniejszym podrozdziale, pogrupowana według typu certyfikatu i wersji protokołu TLS. Wartość organizacji nadającej adresy IP (*ang. Internet Assigned Numbers Authority – IANA*) dla każdego zestawu szyfrowania jest podana po jego opisie tekstowym w nawiasie[20].

W niektórych sytuacjach, na przykład w zamkniętych środowiskach, właściwe może być stosowanie kluczy wstępnych. Klucze wstępne to klucze symetryczne, które istnieją już przed zainicjowaniem sesji TLS i są wykorzystywane przy wyprowadzaniu

[20] Pełną listę wartości IANA dla parametrów TLS można znaleźć pod adresem https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml.

wstępnego klucza tajnego. Informacje o zestawach szyfrowania, które są dopuszczalne w środowiskach z kluczami wstępnymi, znajdują się w załączniku C.

NIST wycofuje się z używania transportu klucza RSA stosowanego w ramach protokołu TLS. Niektóre aplikacje lub środowiska mogą wymagać użycia transportu klucza RSA w okresie przejściowym. Informacje o dopuszczalnych zestawach szyfrowania do wykorzystania w takiej sytuacji znajdują się w załączniku D.

Poniższe zestawienia zestawów szyfrowania są pogrupowane według typu certyfikatu i wersji protokołu TLS. Zestawy szyfrowania wymienione w tych listach to zestawy szyfrowania zawierające algorytmy kryptograficzne zatwierdzone przez NIST. Zestawów szyfrowania, których nie wymieniono w niniejszym podrozdziale, załączniku C lub załączniku D, **nie należy** stosować.

Zestawy szyfrowania wykorzystujące efemeryczny klucz DH i efemeryczny klucz ECDH (tj. te, które mają DHE lub ECDHE w drugim mnemoniku) zapewniają doskonałe utajnienie przekazywania[21]. W przypadku gdy klucze efemeryczne są wykorzystywane do ustanowienia głównego klucza tajnego, każda para kluczy efemerycznych (tj. para kluczy efemerycznych serwera i para kluczy efemerycznych klienta) **powinna** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów.

3.3.1.1 ZESTAWY SZYFROWANIA DLA PROTOKOŁU TLS 1.2 I WCZEŚNIEJSZYCH WERSJI

W pierwszej aktualizacji niniejszych wytycznych znalazł się wymóg obsługi niewielkiej grupy zestawów szyfrowania w celu promowania interoperacyjności i dostosowania do specyfikacji protokołu TLS. Nie ma już żadnych obowiązkowych wymogów dotyczących zestawów szyfrowania. Dopuszczalne jest stosowanie wszystkich zestawów szyfrowania, które obejmują AES i inne algorytmy zatwierdzone przez NIST, chociaż niekoniecznie

[21] Doskonałe utajnienie przekazywania (*ang. perfect forward secrecy*) to stan, w którym ujawnienie długoterminowego klucza prywatnego po jego użyciu do ustanowienia klucza sesji nie powoduje ujawnienia tego klucza sesji.

zapewniają one ten sam poziom bezpieczeństwa. Zestawy szyfrowania, w których wykorzystywany jest algorytm trzykrotnego szyfrowania danych (*ang. Triple Data Encryption Aglorithm – TDEA*, spotyka się również skrót *3DES*) nie są już dozwolone ze względu na ograniczoną ilość danych, które mogą być przetwarzane przy użyciu jednego klucza. Serwer **powinien** być skonfigurowany w taki sposób, aby używał tylko tych zestawów szyfrowania, dla których posiada ważny certyfikat zawierający podpis zapewniający poziom bezpieczeństwa wynoszący co najmniej 112 bitów.

Dzięki usunięciu wymogów dotyczących obsługi określonych zestawów szyfrowania, administratorzy systemów mają większą swobodę w spełnianiu potrzeb swoich środowisk i aplikacji. Zwiększa to również elastyczność działania, umożliwiając administratorom natychmiastowe wyłączenie określonych zestawów szyfrowania po wykryciu ataków, bez naruszania zgodności z wymogami.

W przypadku, gdy obsługiwana jest podgrupa zestawów szyfrowania dopuszczalnych dla certyfikatów serwera, w poniższej liście można znaleźć ogólne wskazówki dotyczące wyboru najsilniejszych opcji:

1. Należy preferować klucze efemeryczne zamiast statycznych (tzn. preferować DHE zamiast DH, a ECDHE zamiast ECDH). Klucze efemeryczne zapewniają doskonałe utajnienie przekazywania.
2. Należy preferować tryb GCM lub CCM zamiast trybu CBC. Użycie uwierzytelnionego trybu szyfrowania zapobiega kilku rodzajom ataków (więcej informacji w podrozdziale 3.3.2). Należy pamiętać, że nie są one dostępne w wersjach wcześniejszych niż TLS 1.2.
3. Należy preferować tryb CCM zamiast CCM_8. Ten drugi obejmuje krótszy znacznik uwierzytelniania, co przekłada się na mniejszą siłę uwierzytelniania.

Lista ta nie musi być ściśle przestrzegana, ponieważ w przypadku niektórych środowisk lub aplikacji mogą wystąpić specjalne okoliczności. Należy pamiętać, że ta lista może stać się nieaktualna, jeśli pojawi się atak na jeden z preferowanych komponentów. Jeśli atak znacząco wpłynie na zalecane zestawy szyfrowania, NIST odniesie się do tej kwestii w komunikacie na stronie centrum zasobów bezpieczeństwa komputerowego NIST (https://csrc.nist.gov).

3.3.1.1.1 ZESTAWY SZYFROWANIA DLA CERTYFIKATÓW ECDSA

Protokół TLS w wersji 1.2 obejmuje uwierzytelnione tryby szyfrowania oraz obsługę algorytmów skrótu SHA-256 i SHA-384, które nie są obsługiwane w poprzednich wersjach. Takie zestawy szyfrowania opisano w publikacji [61] i [56]. Serwery TLS 1.2 skonfigurowane przy użyciu certyfikatów ECDSA mogą być skonfigurowane do obsługi następujących zestawów szyfrowania, które są obsługiwane tylko przez protokół TLS 1.2:

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xC0, 0x2B)
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xC0, 0x2c)
- TLS_ECDHE_ECDSA_WITH_AES_128_CCM (0xC0, 0xAC)
- TLS_ECDHE_ECDSA_WITH_AES_256_CCM (0xC0, 0xAD)
- TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8 (0xC0, 0xAE)
- TLS_ECDHE_ECDSA_WITH_AES_256_CCM_8 (0xC0, 0xAf)
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 (0xC0, 0x23)
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 (0xC0, 0x24)

W przypadku stosowania certyfikatów ECDSA z protokołem TLS w wersji 1.2, 1.1 lub 1.0 serwery TLS mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA[22] (0xC0, 0x09)
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xC0, 0x0A)

[22] W przypadku protokołu TLS w wersjach 1.0 i 1.1 zestawy szyfrowania DHE i ECDHE wykorzystują algorytm SHA-1 do generowania podpisu na parametrach efemerycznych (w tym kluczach) w komunikacie ServerKeyExchange. Chociaż stosowanie algorytmu SHA-1 do generowania podpisu cyfrowego zostało generalnie zabronione w ramach publikacji [10], mogą istnieć wyjątki zawarte w wytycznych dotyczących poszczególnych protokołów. Dopuszczalne jest stosowanie algorytmu SHA-1 do generowania podpisów cyfrowych na parametrach efemerycznych w ramach protokołu TLS. Ze względu na losowy charakter kluczy efemerycznych jest mało prawdopodobne, aby strona trzecia mogła spowodować rzeczywistą kolizję. Serwer i klient nie mają nic do zyskania poprzez wywołanie kolizji dla połączenia. Ze względu na wartości losowe klienta i serwera, serwer, klient lub strona trzecia nie może użyć kolidującego zestawu komunikatu do podszywania się pod klienta lub serwer w kolejnych połączeniach. Jakakolwiek modyfikacja parametrów przez stronę trzecią podczas wykonywania protokołu uzgodnienia ostatecznie spowoduje niepowodzenie w nawiązaniu połączenia.

3.3.1.1.2 ZESTAWY SZYFROWANIA DLA CERTYFIKATÓW RSA

Serwery TLS 1.2, które są skonfigurowane przy użyciu certyfikatów RSA mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xC0, 0x2F)
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xC0, 0x30)
- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 (0x00, 0x9E)
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 (0x00, 0x9F)
- TLS_DHE_RSA_WITH_AES_128_CCM (0xC0, 0x9E)
- TLS_DHE_RSA_WITH_AES_256_CCM (0xC0, 0x9F)
- TLS_DHE_RSA_WITH_AES_128_CCM_8 (0xC0, 0xA2)
- TLS_DHE_RSA_WITH_AES_256_CCM_8 (0xC0, 0xA3)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xC0, 0x27)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xC0, 0x28)
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 (0x00, 0x67)
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 (0x00, 0x6B)

W przypadku stosowania certyfikatów RSA z protokołem TLS w wersji 1.2, 1.1 lub 1.0 serwery TLS mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xC0, 0x13)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xC0, 0x14)
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x00, 0x33)
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x00, 0x39)

3.3.1.1.3 ZESTAWY SZYFROWANIA DLA CERTYFIKATÓW DSA

Serwery TLS 1.2, które są skonfigurowane przy użyciu certyfikatów DSA, mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_DHE_DSS_WITH_AES_128_GCM_SHA256 (0x00, 0xA2)
- TLS_DHE_DSS_WITH_AES_256_GCM_SHA384 (0x00, 0xA3)
- TLS_DHE_DSS_WITH_AES_128_CBC_SHA256 (0x00, 0x40)
- TLS_DHE_DSS_WITH_AES_256_CBC_SHA256 (0x00, 0x6A)

W przypadku stosowania certyfikatów DSA z protokołem TLS w wersji 1.2, 1.1 lub 1.0 serwery TLS mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_DHE_DSS_WITH_AES_128_CBC_SHA (0x00, 0x32)
- TLS_DHE_DSS_WITH_AES_256_CBC_SHA (0x00, 0x38)

3.3.1.1.4 ZESTAWY SZYFROWANIA DLA CERTYFIKATÓW DH

Certyfikaty DH zawierają klucz statyczny i są podpisywane przy użyciu algorytmu DSA lub RSA. W przeciwieństwie do zestawów szyfrowania, w których wykorzystuje się efemeryczne klucze DH, te zestawy obejmują statyczne parametry DH. Chociaż stosowanie kluczy statycznych jest technicznie dopuszczalne, zaleca się stosowanie zestawów szyfrowania z kluczem efemerycznym i jest ono preferowane w stosunku do stosowania zestawów szyfrowania wymienionych w niniejszym podrozdziale.

Serwery TLS 1.2, które są skonfigurowane przy użyciu certyfikatów DH z podpisem DSA, mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_DH_DSS_WITH_AES_128_GCM_SHA256 (0x00, 0xA4)
- TLS_DH_DSS_WITH_AES_256_GCM_SHA384 (0x00, 0xAs)
- TLS_DH_DSS_WITH_AES_128_CBC_SHA256 (0x00, 0x3E)
- TLS_DH_DSS_WITH_AES_256_CBC_SHA256 (0x00, 0x68)

W przypadku stosowania certyfikatów DH z podpisem DSA i protokołem TLS w wersji 1.2, 1.1 lub 1.0 serwery TLS mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_DH_DSS_WITH_AES_128_CBC_SHA (0x00, 0x30)
- TLS_DH_DSS_WITH_AES_256_CBC_SHA (0x00, 0x36)

Serwery TLS 1.2, które są skonfigurowane przy użyciu certyfikatów DH z podpisem RSA, mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_DH_RSA_WITH_AES_128_GCM_SHA256 (0x00, 0xA0)
- TLS_DH_RSA_WITH_AES_256_GCM_SHA384 (0x00, 0xA1)
- TLS_DH_RSA_WITH_AES_128_CBC_SHA256 (0x00, 0x3F)
- TLS_DH_RSA_WITH_AES_256_CBC_SHA256 (0x00, 0x69)

W przypadku stosowania certyfikatów DH z podpisem RSA i protokołem TLS w wersji 1.2, 1.1 lub 1.0 serwery TLS mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_DH_RSA_WITH_AES_128_CBC_SHA (0x00, 0x31)
- TLS_DH_RSA_WITH_AES_256_CBC_SHA (0x00, 0x37)

3.3.1.1.5 ZESTAWY SZYFROWANIA DLA CERTYFIKATÓW ECDH

Certyfikaty ECDH zawierają klucz statyczny i są podpisywane przy użyciu algorytmu ECDSA lub RSA. W przeciwieństwie do zestawów szyfrowania, w których wykorzystuje się efemeryczne klucze ECDH, te zestawy obejmują statyczne parametry ECDH. Zaleca się stosowanie zestawów szyfrowania z kluczem efemerycznym i jest ono preferowane w stosunku do stosowania zestawów szyfrowania wymienionych w niniejszym podrozdziale.

Serwery TLS 1.2, które są skonfigurowane przy użyciu certyfikatów ECDH z podpisem ECDSA, mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256 (0xC0, 0x2D)
- TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384 (0xC0, 0x2E)
- TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256 (0xC0, 0x25)
- TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384 (0xC0, 0x26)

W przypadku stosowania certyfikatów ECDH z podpisem ECDSA i protokołem TLS w wersji 1,2, 1.1 lub 1.0 serwery TLS mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA (0xC0, 0x04)
- TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA (0xC0, 0x05)

Serwery TLS 1.2, które są skonfigurowane przy użyciu certyfikatów ECDH z podpisem RSA, mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256 (0xC0, 0x31)
- TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384 (0xC0, 0x32)
- TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256 (0xC0, 0x29)
- TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384 (0xC0, 0x2A)

W przypadku stosowania certyfikatów ECDH z podpisem RSA i protokołem TLS w wersji 1.2, 1.1 lub 1.0 serwery TLS mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_ECDH_RSA_WITH_AES_128_CBC_SHA (0xC0, 0x0E)
- TLS_ECDH_RSA_WITH_AES_256_CBC_SHA (0xC0, 0x0F)

3.3.1.1.6 ZESTAWY SZYFROWANIA DLA PROTOKOŁU TLS 1.3

Serwery TLS 1.3 mogą być skonfigurowane do obsługi następujących zestawów szyfrowania:

- TLS_AES_128_GCM_SHA256 (0x13, 0x01)
- TLS_AES_256_GCM_SHA384 (0x13, 0x02)
- TLS_AES_128_CCM_SHA256 (0x13, 0x04)
- TLS_AES_128_CCM_8_SHA256 (0x13, 0x05)

Te zestawy szyfrowania można stosować z certyfikatami serwera RSA lub ECDSA. Certyfikaty DSA i DH nie są obsługiwane w ramach protokołu TLS 1.3. Takie zestawy szyfrowania mogą być również używane z kluczami wstępnymi, jak określono w załączniku C.

3.3.2 UWAGI DOTYCZĄCE IMPLEMENTACJI

Administratorzy systemów muszą w pełni zrozumieć konsekwencje wyboru określonych zestawów szyfrowania i skonfigurowania aplikacji do obsługi tylko tych zestawów. Poziom bezpieczeństwa zapewniany przez kryptografię jest limitowany przez najsłabszy zestaw szyfrowania obsługiwany przez daną konfigurację. Podczas konfigurowania implementacji występuje szereg czynników, które wpływają na wybór obsługiwanych zestawów szyfrowania.

W dokumencie RFC 4346 [24] opisano ataki czasowe na zestawy szyfrowania CBC, a także techniki umożliwiające zapobieganie im. Implementacje TLS **powinny** używać błędu bad_record_mac do sygnalizowania błędu dopełnienia, gdy komunikacja jest zabezpieczona przy użyciu zestawu szyfrowania CBC. Implementacje **powinny** obliczać kod MAC niezależnie od tego, czy występują błędy dopełnienia.

Oprócz ataków na zestawy CBC, o których mowa w dokumencie RFC 4346 [24], również ataki typu „lucky thirteen" [1] pokazują, że do zapobiegania atakom czasowym potrzebna jest procedura deszyfracji w czasie stałym. Implementacje TLS **powinny** obsługiwać deszyfrowanie w czasie stałym (*ang. constant-time*) lub prawie stałym (*ang. near constant-time*).

Atak typu POODLE wykorzystuje niedeterministyczne dopełnianie w ramach protokołu SSL 3.0 [43]. Podatność ta nie występuje w protokołach TLS, ale może istnieć w implementacji TLS, gdy kod dekodera SSL jest ponownie wykorzystywany do przetwarzania danych TLS [38]. Implementacje TLS powinny prawidłowo dekodować bajty dopełnienia CBC.

Należy pamiętać, że atakom opartym na CBC można zapobiec stosując zestawy szyfrowania AEAD (np. GCM, CCM), które są obsługiwane w ramach protokołu TLS 1.2.

3.3.2.1 OBSŁUGA ALGORYTMÓW

Wiele serwerów i klientów TLS obsługuje zestawy szyfrowania, które nie składają się wyłącznie z algorytmów zatwierdzonych przez NIST. Dlatego ważne jest skonfigurowanie serwera w taki sposób, aby korzystał wyłącznie z zestawów szyfrowania zalecanych przez NIST. Jest to szczególnie ważne w przypadku

implementacji serwerów, które nie umożliwiają administratorowi serwera określenia kolejności preferencji. W przypadku takich serwerów jedynym sposobem zapewnienia, że serwer korzysta z algorytmów zatwierdzonych przez NIST, jest wyłączenie zestawów szyfrowania, w których stosuje się inne algorytmy.

Jeśli implementacja serwera umożliwia administratorowi serwera określenie preferencji, zaleca się, aby skorzystał on z rekomendacji dotyczących preferencji wymienionych w podrozdziale 3.3.1.1.

3.3.3 WALIDACJA KRYPTOGRAFII

Moduł kryptograficzny używany przez serwer **powinien** być modułem kryptograficznym o potwierdzonej zgodności ze standardem FIPS 140 [50, 51]. Wszystkie algorytmy kryptograficzne, które wchodzą w skład skonfigurowanych zestawów szyfrowania i generatora liczb losowych, **powinny** być objęte walidacją.

Należy zauważyć, że funkcja pseudolosowa (PRF) protokołu TLS 1.0 i 1.1 używa równolegle algorytmów MD5 i SHA-1, więc jeśli jedna z funkcji skrótu ulegnie awarii, bezpieczeństwo nie jest zagrożone. MD5 nie jest algorytmem zatwierdzonym przez NIST, jednak funkcja PRF została określona jako dopuszczalna w publikacji NIST SP 800-135 [21]. W wersji TLS 1.2 domyślną funkcją skrótu w PRF jest SHA-256. W wersji TLS 1.3 zastąpiono funkcję PRF opartą na HMAC funkcją wyprowadzania klucza typu extract-and-expand (*Extract-and-Expand Key Derivation Function – HKDF*), opisaną w dokumencie RFC 5869 [37].

Należy również zauważyć, że w wersjach TLS wcześniejszych niż 1.2 użycie algorytmu SHA-1 jest uważane za dopuszczalne do podpisywania kluczy efemerycznych oraz do uwierzytelniania klienta za pomocą podpisów cyfrowych. Wynika to z tego, że stronie trzeciej trudno byłoby spowodować kolizję, która nie zostałaby wykryta.

Poza wyjątkiem dla algorytmu SHA-1, który dotyczył szczególnych przypadków wymienionych powyżej, wszystkie stosowane metody kryptograficzne **powinny** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów. Wszystkie certyfikaty serwera i klienta **powinny** zawierać klucze publiczne zapewniające poziom bezpieczeństwa wynoszący co najmniej 112 bitów. Wszystkie certyfikaty serwera

i klienta oraz certyfikaty w ich ścieżkach certyfikacji **należy** podpisywać przy użyciu par kluczy zapewniających poziom bezpieczeństwa wynoszący co najmniej 112 bitów oraz SHA-224 lub silniejszego algorytmu skrótu. Wszystkie klucze efemeryczne używane przez klienta i serwer **powinny** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów. Wszystkie algorytmy symetryczne stosowane do ochrony danych TLS **powinny** wykorzystywać klucze, które zapewniają poziom bezpieczeństwa wynoszący co najmniej 112 bitów.

Certyfikat zgodności ze standardem FIPS 140 dla modułu kryptograficznego wykorzystywanego przez serwer **powinien** wskazywać, że generator bitów losowych (RBG) został poddany walidacji zgodnie z publikacjami z serii NIST SP 80090 [8, 48, 66][23]. Wartość losowa serwera, wysyłana w komunikacie ServerHello, zawiera wartość 4-bajtowego znacznika czasu[24] i 28-bajtową wartość losową w wersjach TLS 1.0, 1.1 i 1.2 oraz zawiera 32-bajtową wartość losową w wersji TLS 1.3. Do generowania losowych bajtów wartości losowej serwera należy stosować generator liczb losowych poddany walidacji[25]. Poddanego walidacji generatora liczb losowych należy również używać do generowania 4-bajtowego znacznika czasu wartości losowej serwera.

3.4 OBSŁUGA ROZSZERZEŃ PROTOKOŁU TLS

W dokumentach RFC opisanych jest kilka rozszerzeń protokołu TLS. Niniejszy podrozdział zawiera zalecenia dotyczące podzbioru rozszerzeń protokołu TLS, które **należy** stosować w instytucjach rządowych, które **powinny** być stosowane lub których **nie należy** stosować, jako że stają się one powszechne w komercyjnie dostępnych serwerach i klientach TLS.

---

[23] Walidacja będzie obejmowała zgodność z publikacją NIST SP 800-90C, gdy tylko będzie ona dostępna.

[24] W przypadku protokołu TLS wartość znacznika czasu nie musi być prawidłowa. Może to być dowolna 4-bajtowa wartość, o ile nie istnieją inne ograniczenia wynikające z protokołów wyższego poziomu lub aplikacji.

[25] Implementacje TLS 1.3 zawierają mechanizm ochrony przed atakami typu *downgrade* związany z wartością losową, który nadpisuje ostatnie osiem bajtów wartości losowej serwera stałą wartością. Podczas negocjacji protokołu TLS 1.2 ostatnie osiem bajtów losowej liczby serwera zostanie ustawione na wartości 44 4F 57 4E 47 52 44 01. Gdy negocjowany jest protokół TLS 1.1 lub starsza wersja, ostatnie osiem bajtów wartości losowej zostanie ustawione na 44 4F 57 4E 47 52 44 00. To nadpisanie jest niezależne od poddanego walidacji generatora bitów losowych.

Administratorzy systemów muszą dokładnie rozważyć ryzyko związane z obsługą rozszerzeń, które nie są wymienione jako obowiązkowe. Omówiono tu tylko te rozszerzenia, których specyfikacja ma wpływ na bezpieczeństwo, ale czytelnik powinien wiedzieć, że obsługa każdego rozszerzenia może mieć niezamierzone konsekwencje dla bezpieczeństwa. Korzystanie z rozszerzeń zwiększa w szczególności prawdopodobieństwo wystąpienia błędów w implementacji i może spowodować, że system będzie podatny na ataki. Na przykład, błąd bezpieczeństwa Heartbleed [70] był wadą w implementacji rozszerzenia Heartbeat [64]. Chociaż rozszerzenie nie ma bezpośredniego związku z bezpieczeństwem, jednak błąd w implementacji narażał dane serwera, w tym klucze prywatne, na atak.

Ogólnie rzecz biorąc, serwery **powinny** być skonfigurowane wyłącznie do obsługi rozszerzeń, które są wymagane przez aplikację lub które zwiększają bezpieczeństwo. **Nie należy** stosować rozszerzeń, które nie są potrzebne.

3.4.1 OBOWIĄZKOWE ROZSZERZENIA PROTOKOŁU TLS

Serwer **powinien** obsługiwać rozszerzenia protokołu TLS wymienione poniżej.

1. Sygnalizacja renegocjacji (*ang. Renegotiation Indication*)
2. Identyfikacja nazwy serwera (*ang. Server Name Indication*)
3. Rozszerzony główny klucz tajny (*ang. Extended Master Secret*)
4. Algorytmy podpisu (*ang. Signature Algorithms*)
5. Żądanie statusu certyfikatu (*ang. Certificate Status Request*)

3.4.1.1 SYGNALIZACJA RENEGOCJACJI (ANG. RENEGOTIATION INDICATION)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

W wersjach protokołu TLS od 1.0 do 1.2 renegocjacja sesji jest podatna na atak, w którym atakujący tworzy połączenie TLS z serwerem docelowym, wprowadza do niego wybraną przez siebie zawartość, a następnie dołącza nowe połączenie TLS od legalnego klienta. Serwer traktuje wykonanie przez klienta wstępnego protokołu uzgodnienia TLS jako renegocjację sesji wynegocjowanej przez atakującego. Dlatego uważa, że początkowe dane przekazywane przez atakującego pochodzą od uprawnionego klienta. Rozszerzenie dotyczące renegocjacji sesji jest przeznaczone do zapobiegania takiemu scalaniu lub

przechwytywaniu sesji. Rozszerzenie to wykorzystuje koncepcję kryptograficznego wiązania początkowych negocjacji sesji i renegocjacji sesji.

Implementacje serwera **powinny** wykonywać początkowe i kolejne renegocjacje zgodnie z dokumentem RFC 5746 [59] oraz RFC 8446 [57].

3.4.1.2 IDENTYFIKACJA NAZWY SERWERA (*ANG. SERVER NAME INDICATION*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Pod tym samym adresem sieciowym może istnieć wiele serwerów wirtualnych. Rozszerzenie identyfikujące nazwę serwera pozwala klientowi określić, z którym z serwerów znajdujących się pod adresem próbuje się połączyć. Rozszerzenie to jest dostępne we wszystkich wersjach protokołu TLS. Serwer **powinien** być w stanie przetworzyć i odpowiedzieć na rozszerzenie identyfikujące nazwę serwera otrzymane w komunikacie ClientHello, jak opisano w dokumencie [29].

3.4.1.3 ROZSZERZONY GŁÓWNY KLUCZ TAJNY (*ANG. EXTENDED MASTER SECRET*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

W publikacji Bhargavana i wsp. wykazano, że aktywny atakujący może zsynchronizować dwie sesje TLS w taki sposób, że będą one współdzielić ten sam główny klucz tajny, co umożliwia mu przeprowadzenie ataku typu MIM [12]. Rozszerzenie polegające na rozszerzeniu głównego klucza tajnego (*ang. Extended Master Secret*) opisane w dokumencie RFC 7627 [13] zapobiega takim atakom poprzez powiązanie głównego klucza tajnego ze skróconym rejestrem pełnego protokołu uzgodnienia. Serwer **powinien** obsługiwać to rozszerzenie.

3.4.1.4 ALGORYTMY PODPISU (ANG. SIGNATURE ALGORITHMS)

Dotyczy protokołu TLS w wersji: 1.2, 1.3

Serwery **powinny** obsługiwać przetwarzanie rozszerzenia algorytmów podpisu otrzymanego w komunikacie ClientHello. Rozszerzenie, jego składnia i zasady przetwarzania są opisane w podpunktach 7.4.1.4.1, 7.4.2 i 7.4.3 dokumentu RFC 5246 [25] i podpunkcie 4.2.3 dokumentu RFC 8446 [57]. Należy zwrócić uwagę, że rozszerzenie opisane w dokumencie RFC 8446 aktualizuje rozszerzenie opisane w RFC 5246 poprzez dodanie dodatkowego schematu podpisu.

3.4.1.5 ŻĄDANIE STATUSU CERTYFIKATU (ANG. CERTIFICATE STATUS REQUEST)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Jeżeli klient chce otrzymać od serwera TLS status unieważnienia jego certyfikatu, umieszcza w komunikacie ClientHello rozszerzenie z żądaniem statusu certyfikatu (status_request). Po otrzymaniu rozszerzenia status_request serwer z certyfikatem wydanym przez CA i obsługujący protokół OCSP **powinien** udostępnić status certyfikatu wraz ze swoim certyfikatem, wysyłając komunikat CertificateStatus bezpośrednio po komunikacie Certificate[26]. Rozszerzenie to samo w sobie jest rozszerzalne, jednak w dokumencie [29] zdefiniowany jest tylko status certyfikatu typu OCSP. Rozszerzenie to nazywane jest również przypinaniem OCSP.

3.4.2 WARUNKOWE ROZSZERZENIA PROTOKOŁU TLS

Należy umożliwić obsługę następujących rozszerzeń protokołu TLS w okolicznościach opisanych w punktach poniżej:

1. Awaryjne sygnalizowanie wartości zestawu szyfrowania (*ang. Fallback Signaling Cipher Suite Value – SCSV*) **powinno** być obsługiwane, jeżeli serwer obsługuje wersje TLS wcześniejsze niż TLS 1.2 i nie obsługuje TLS 1.3.
2. Rozszerzenie obsługiwanych grup (*ang. Supported Groups*) **powinno** być obsługiwane, jeżeli serwer obsługuje efemeryczne zestawy szyfrowania ECDH lub jeżeli serwer obsługuje protokół TLS 1.3.
3. Rozszerzenie udostępniania klucza (*ang. Key Share*) **powinno** być obsługiwane, jeżeli serwer obsługuje protokół TLS 1.3.
4. Rozszerzenie formatów przecinkowych EC (*ang. EC Point Format*) **powinno** być obsługiwane, jeżeli serwer obsługuje zestawy szyfrowania EC.
5. Rozszerzenie statusu wielu certyfikatów (*ang. Multiple Certificate Status*) **powinno** być obsługiwane, jeśli informacje o statusie certyfikatu serwera są dostępne za pośrednictwem OCSP, a rozszerzenie jest obsługiwane przez implementację serwera.

[26] W wersji TLS 1.3 serwer umieszcza status certyfikatu w komunikacie Certificate.

6. Rozszerzenie identyfikacji zaufanego CA (*ang. Trusted CA Indication*) **powinno** być obsługiwane, jeśli serwer komunikuje się z klientami o ograniczonej pamięci (np. urządzeniami klienckimi o małej pamięci w Internecie rzeczy (*ang. Internet of Things – IoT*), a dla serwera certyfikaty były wydawane przez wiele CA.
7. Rozszerzenie *Encrypt-then-MAC* **powinno** być obsługiwane, jeżeli serwer jest skonfigurowany do negocjowania zestawów szyfrowania CBC.
8. Rozszerzenie skróconego kodu HMAC (*ang. Truncated HMAC*) może być obsługiwane, jeśli serwer komunikuje się z klientami o ograniczonej liczbie urządzeń, obsługiwane są zestawy szyfrowania wykorzystujące tryb CBC, a implementacja serwera nie obsługuje dopełniania o zmiennej długości.
9. Rozszerzenie klucza wstępnego (*ang. Pre-Shared Key*) powinno być obsługiwane, jeżeli serwer obsługuje protokół TLS 1.3.
10. Rozszerzenie trybów wymiany klucza wstępnego (*ang. Pre-Shared Key Exchange Modes*) powinno być obsługiwane, jeśli serwer obsługuje protokół TLS 1.3 i rozszerzenie kluczy wstępnych (*ang. Pre-Shared Key*).
11. Rozszerzenie obsługiwanych wersji (*ang. Supported Versions*) **powinno** być obsługiwane, jeżeli serwer obsługuje protokół TLS 1.3.
12. Rozszerzenie *Cookie* **powinno** być obsługiwane, jeżeli serwer obsługuje protokół TLS 1.3.
13. Rozszerzenie algorytmów podpisu certyfikatu (*ang. Certificate Signature Algorithms*) **powinno** być obsługiwane, jeżeli serwer obsługuje protokół TLS 1.2.
14. Rozszerzenie uwierzytelniania klienta po protokole uzgodnienia (*ang. Post-handshake Client Authentication*) powinno być obsługiwane, jeżeli serwer obsługuje protokół TLS 1.3.
15. Rozszerzenie znaczników czasu podpisanego certyfikatu (*ang. Signed Certificate Timestamps*) powinno być obsługiwane, jeśli certyfikat serwera został wydany przez publicznie zaufany CA i nie zawiera on rozszerzenia listy znaczników czasu podpisanego certyfikatu.

3.4.2.1 AWARYJNE SYGNALIZOWANIE WARTOŚCI ZESTAWU SZYFROWANIA (*ANG. FALLBACK SIGNALING CIPHER SUITE VALUE – SCSV*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Protokół TLS 1.3 obejmuje mechanizm ochrony przed atakami typu *downgrade*, w który poprzednie wersje nie były wyposażone. W protokołach wcześniejszych niż TLS 1.3, atakujący może wykorzystać zewnętrzną negocjację wersji jako sposób na wymuszenie niepotrzebnego przejścia na starszy protokół w ramach połączenia. Konkretnie, atakujący może wywołać wrażenie, że połączenie nie powiodło się przy żądanej wersji protokołu TLS, a niektóre implementacje klientów spróbują połączenia ponownie ze starszą wersją protokołu. Ta wartość zestawu szyfrowania, opisana w dokumencie RFC 7507 [42], zapewnia mechanizm zapobiegający niezamierzonemu przejściu na starszy protokół w wersjach wcześniejszych niż TLS 1.3. Klienci sygnalizują, kiedy połączenie jest awaryjne, a jeśli serwer uzna je za niewłaściwe (tj. gdy serwer obsługuje nowszą wersję TLS), zwraca alert krytyczny.

Jeżeli serwer obsługuje wersje protokołu TLS wcześniejsze niż TLS 1.2, a wersja TLS 1.3 nie jest obsługiwana, **powinien** obsługiwać awaryjne rozszerzenie SCSV.

3.4.2.2 OBSŁUGIWANE GRUPY (ANG. SUPPORTED GROUPS)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Rozszerzenie obsługiwanych grup (supported_groups) pozwala klientowi wskazać obsługiwane przez niego grupy parametrów domeny serwera. Rozszerzenie było pierwotnie nazywane rozszerzeniem obsługiwanych krzywych eliptycznych (elliptic_curves) i było używane tylko dla grup krzywych eliptycznych, ale teraz może być również używane do negocjowania grup ciał skończonych. W wersji TLS 1.3 rozszerzenie obsługiwanych grup musi być używane do negocjowania zarówno grup krzywych eliptycznych, jak i grup ciał skończonych. Serwery obsługujące efemeryczne zestawy szyfrowania ECDH lub protokół TLS 1.3 **powinny** obsługiwać to rozszerzenie. W przypadku skonfigurowania zestawów szyfrowania opartych na krzywych eliptycznych serwer **powinien** obsługiwać co najmniej jedną z zatwierdzonych przez NIST krzywych, P-256 (secp256r1) i P384 (secp384r1), jak określono w dokumencie RFC 8422 [52]. Dodatkowe krzywe eliptyczne zalecane przez NIST zostały wymienione

w załączniku D do publikacji NIST SP 800-56A [6]. Dopuszcza się obsługę grup ciał skończonych, które zostały zatwierdzone do użytku z protokołem TLS w załączniku D do publikacji NIST SP 800-56A.

3.4.2.3 UDOSTĘPNIANIE KLUCZA (*ANG. KEY SHARE*)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie udostępniające klucz jest stosowane w ramach protokołu TLS 1.3 do przesyłania parametrów kryptograficznych. Serwery obsługujące protokół TLS 1.3 **powinny** obsługiwać to rozszerzenie zgodnie z opisem w punkcie 4.2.7 dokumentu RFC 8446 [57].

3.4.2.4 OBSŁUGIWANE FORMATY PRZECINKOWE (*ANG. SUPPORTED POINT FORMATS*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Serwery, które obsługują zestawy szyfrów EC w ramach wersji TLS 1.2 i starszych, **powinny** być w stanie przetwarzać obsługiwany format przecinkowy otrzymany w komunikacie ClientHello przez klienta. Serwery **powinny** przetwarzać to rozszerzenie zgodnie z punktem 5.1 dokumentu RFC 8422 [52].

Serwery obsługujące zestawy szyfrowania EC powinny również być w stanie przesłać obsługiwany format przecinkowy EC w komunikacie ServerHello, jak opisano w punkcie 5.2 dokumentu RFC 8422 [52].

3.4.2.5 STATUS WIELU CERTYFIKATÓW (ANG. MULTIPLE CERTIFICATE STATUS)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie dla statusu wielu certyfikatów ulepsza rozszerzenie żądania statusu certyfikatu (*ang. Certificate Status Request*) opisane w podrozdziale 3.4.1.5, pozwalając klientowi na żądanie statusu wszystkich certyfikatów dostarczonych przez serwer w ramach protokołu uzgodnienia TLS. Gdy serwer zwraca status unieważnienia wszystkich certyfikatów w łańcuchu certyfikatów, klient nie musi wysyłać zapytania do żadnych dostawców informacji o unieważnieniu, takich jak obiekty odpowiadające OCSP. Rozszerzenie to zostało udokumentowane w dokumencie RFC 6961 [54]. Serwery, które mają taką możliwość i posiadają

certyfikaty wydane przez CA obsługujące OCSP, **powinny** być skonfigurowane do obsługi tego rozszerzenia.

3.4.2.6 IDENTYFIKACJA ZAUFANEGO CA (*ANG. TRUSTED CA INDICATION*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie identyfikujące zaufany CA (trusted_ca_keys) umożliwia klientowi określenie, które klucze główne CA posiada. Jest to przydatne w przypadku sesji, w których klient ma ograniczoną pamięć i niewielką liczbę kluczy głównych CA. Serwery, które komunikują się z klientami o ograniczonej pamięci i w przypadku których certyfikaty zostały wydane przez wiele CA, **powinny** być w stanie przetworzyć i odpowiedzieć na rozszerzenie identyfikujące zaufany CA otrzymane w komunikacie ClientHello, jak opisano w dokumencie [29].

3.4.2.7 ENCRYPT-THEN-MAC

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Szereg ataków na zestawy szyfrowania CBC było możliwych ze względu na kolejność operacji MAC-then-encrypt (generowanie MAC, a następnie szyfrowanie) stosowaną w wersjach 1.0, 1.1 i 1.2 protokołu TLS. Rozszerzenie Encrypt-then-MAC zmienia kolejność, w jakiej operacje szyfrowania i generowania kodu MAC są stosowane wobec danych. Uważa się, że zapewnia to większe bezpieczeństwo i łagodzi skutki lub zapobiega kilku znanym atakom na zestawy szyfrowania CBC. Serwery, które są skonfigurowane do negocjowania zestawów szyfrowania CBC, **powinny** obsługiwać to rozszerzenie zgodnie z opisem w publikacji [33].

3.4.2.8 SKRÓCONY KOD HMAC (*ANG. TRUNCATED HMAC*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie skracające kod HMAC (*ang. Truncated HMAC*) pozwala na skrócenie danych wyjściowych HMAC do 80 bitów w celu wykorzystania ich jako znacznika MAC. 80-bitowy znacznik MAC jest zgodny z zaleceniami zawartymi w publikacji NIST SP 800-107 [20], ale obniża poziom bezpieczeństwa zapewniany przez algorytm integralności. Ze względu na fakt, że fałszowanie znacznika MAC jest atakiem online, a sesja TLS zostanie natychmiast przerwana po napotkaniu nieprawidłowego znacznika MAC, ryzyko wynikające z zastosowania tego rozszerzenia jest niewielkie. Jednakże skrócone

znaczniki MAC nie mogą być stosowane w połączeniu z dopełnianiem zmiennej długości ze względu na ataki opisane przez Patersona i wsp. [53]. Rozszerzenie to ma zastosowanie tylko wtedy, gdy obsługiwane są zestawy szyfrowania wykorzystujące tryby CBC.

3.4.2.9 KLUCZ WSTĘPNY (ANG. PRE-SHARED KEY)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie z kluczem wstępnym (*ang. Pre-Shared Key – PSK*, pre_shared_key), dostępne dla protokołu TLS 1.3, służy do identyfikacji klucza wstępnego, który ma być użyty do ustanowienia klucza PSK. W ramach protokołu TLS 1.3 klucze wstępne mogą być ustanawiane poza pasmem, jak w TLS 1.2 lub wersjach starszych, albo w poprzednim połączeniu, w którym to przypadku są używane do wznowienia sesji. Serwery obsługujące protokół TLS 1.3 mogą być skonfigurowane do obsługi tego rozszerzenia w celu obsługi wznowienia sesji lub obsługi wykorzystania kluczy wstępnych, które są tworzone poza pasmem.

3.4.2.10 TRYBY WYMIANY KLUCZY WSTĘPNYCH (ANG. PRE-SHARED KEY EXCHANGE MODES)

Dotyczy protokołu TLS w wersji 1.3

Klient TLS 1.3 musi wysyłać rozszerzenie trybów wymiany kluczy wstępnych (psk_key_exchange_modes), jeśli wysyła rozszerzenie z kluczem wstępnym (*ang. Pre-Shared Key*). Serwery TLS 1.3 wykorzystują listę trybów wymiany kluczy obecnych w rozszerzeniu do wyboru odpowiedniej metody wymiany kluczy. Serwery TLS, które obsługują wersję TLS 1.3 i rozszerzenie z kluczem wstępnym, **powinny** obsługiwać to rozszerzenie.

3.4.2.11 OBSŁUGIWANE WERSJE (ANG. SUPPORTED VERSIONS)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie obsługiwanych wersji jest wysyłane w komunikacie ClientHello w celu wskazania, które wersje protokołu TLS obsługuje klient. Serwer TLS 1.3 **powinien** być w stanie przetworzyć to rozszerzenie. Jeśli nie ma go w wiadomości ClientHello, serwer **powinien** zastosować negocjację wersji określoną dla protokołu TLS 1.2 i starszych wersji.

3.4.2.12 COOKIE

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie Cookie pozwala serwerowi zmusić klienta do udowodnienia, że jest osiągalny pod swoim widocznym adresem sieciowym i przenieść informacje o stanie do klienta. Serwery obsługujące protokół TLS 1.3 mogą obsługiwać rozszerzenie Cookie zgodnie z dokumentem RFC 8446 [57].

3.4.2.13 ALGORYTMY PODPISU CERTYFIKATU (ANG. CERTIFICATE SIGNATURE ALGORITHMS)

Dotyczy protokołu TLS w wersji: 1.2, 1.3

Rozszerzenie dotyczące algorytmów podpisu certyfikatu (*ang. Certificate Signature Algorithms*, signature_algorithms_cert) określa algorytmy podpisu, które mogą być używane w certyfikatach (gdy go nie ma, algorytmy z rozszerzenia dotyczącego algorytmów podpisu mają zastosowanie również do certyfikatów). Serwery TLS obsługujące protokół TLS 1.3 **powinny** obsługiwać to rozszerzenie i **powinno** ono być obsługiwane również w przypadku protokołu TLS 1.2.

3.4.2.14 UWIERZYTELNIANIE KLIENTA PO PROTOKOLE UZGODNIENIA (*ANG. POST-HANDSHAKE CLIENT AUTHENTICATION*)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie uwierzytelniania klienta po protokole uzgodnienia (*ang. Post-handshake Client Authentication*, post_handshake_auth) umożliwia serwerowi zażądanie uwierzytelnienia klienta po wykonaniu protokołu uzgodnienia. Serwery TLS, które obsługują wersję TLS 1.3 mogą obsługiwać to rozszerzenie.

3.4.2.15 ZNACZNIKI CZASU PODPISANEGO CERTYFIKATU

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Projekt Certificate Transparency (opisany w dokumencie RFC 6962 [40]) ma na celu ograniczenie skutków zagrożeń związanych z certyfikatami poprzez zwiększenie przejrzystości wydawania certyfikatów podpisywanych przez CA. Próbuje się to osiągnąć poprzez wykorzystanie publicznych rejestrów certyfikatów, monitorowanie publicznych rejestrów oraz audyt publicznych certyfikatów. Rejestry certyfikatów są kryptograficznie zabezpieczonymi zapisami certyfikatów, które są otwarte do

publicznego wglądu. Certyfikaty mogą być dołączane do rejestrów, ale nie mogą być usuwane, modyfikowane lub wstawiane do środka rejestru. Rejestry certyfikatów są monitorowane pod kątem podejrzanych certyfikatów, takich jak te, które nie zostały autoryzowane przez domenę, z której rzekomo pochodzą. Audytorzy mają możliwość sprawdzenia przynależności danego certyfikatu do rejestru, a także weryfikacji integralności i spójności rejestrów.

Dowód na to, że certyfikat serwera został przekazany do rejestrów utworzonych w ramach programu Certificate Transparency może być dostarczony klientom w samym certyfikacie lub w ramach rozszerzenia protokołu TLS dotyczącego podpisanych znaczników czasu certyfikatu (*ang. signed_certificate_timestamp*). Serwery z certyfikatami wydanymi przez publicznie zaufane CA, które nie obsługują rozszerzenia listy znaczników czasu podpisanego certyfikatu, powinny obsługiwać rozszerzenie TLS znaczników czasu podpisanego certyfikatu.

3.4.3 ODRADZANE ROZSZERZENIA PROTOKOŁU TLS

Rozszerzenia wymienione poniżej **nie powinny** być stosowane:

1. Adres URL certyfikatu klienta (*ang. Client Certificate URL*)
2. Wczesna identyfikacja danych (*ang. Early Data Indication*)

Rozszerzenie surowych kluczy publicznych (*ang. Raw Public Keys*) nie **powinno** być obsługiwane.

3.4.3.1 ADRES URL CERTYFIKATU KLIENTA (*ANG. CLIENT CERTIFICATE URL*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie adresu URL certyfikatu klienta (*ang. Client Certificate URL*) umożliwia klientowi wysłanie adresu URL kierującego do certyfikatu zamiast wysyłania certyfikatu do serwera podczas wzajemnego uwierzytelniania. Może to być bardzo przydatne do wzajemnego uwierzytelniania z klientami o ograniczonych zasobach. Rozszerzenie to może być jednak wykorzystane w złych celach. Adres URL może należeć do nieszkodliwego serwera, na którym klient chciałby przeprowadzić atak metodą DoS, zamieniając serwer TLS w atakującego. Serwer obsługujący to rozszerzenie działa również jako klient podczas pobierania certyfikatu, z czym wiążą

się dodatkowe zagrożenia. Z tych powodów rozszerzenie adresu URL certyfikatu klienta **nie powinno** być obsługiwane. Jeśli jednak dana instytucja rządowa stwierdzi, że ryzyko jest minimalne i rozszerzenie to jest potrzebne w środowiskach, w których klienci znajdują się na urządzeniach o ograniczonych zasobach, rozszerzenie to może być obsługiwane. Jeśli rozszerzenie adresu URL certyfikatu klienta jest obsługiwane, serwer **powinien być** skonfigurowany tak, aby zapobiegać zagrożeniom opisanym powyżej oraz w punkcie 11.3 dokumentu [29].

3.4.3.2 IDENTYFIKACJA WCZESNYCH DANYCH (*ANG. EARLY DATA INDICATION*)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie identyfikacji wczesnych danych (*ang. Early Data Indication*, early_data) umożliwia klientowi wysyłanie danych aplikacji w komunikacie ClientHello, jeśli używane są klucze wstępne. Dotyczy to zarówno kluczy wstępnych utworzonych poza pasmem, jak i tych używanych do wznawiania sesji. Protokół TLS nie chroni tych wczesnych danych przed atakami powtórzeniowymi. Serwery **nie powinny** przetwarzać wczesnych danych otrzymanych w komunikacie ClientHello. Jeżeli serwer jest skonfigurowany do wysyłania rozszerzenia identyfikacji wczesnych danych, powinien stosować metody ochrony przed atakami powtórzeniowymi, takie jak opisane w punkcie 8 dokumentu RFC 8446 [57]. Więcej informacji na temat wczesnych danych (nazywanych również danymi 0-RTT) można znaleźć w podrozdziale 3.6.

3.4.3.3 SUROWE KLUCZE PUBLICZNE (*ANG. RAW PUBLIC KEYS*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Rozszerzenie surowych kluczy publicznych (*ang. Raw Public Keys*), opisane w dokumencie RFC 7250 [71], zapewnia alternatywę dla uwierzytelniania opartego na certyfikatach, w ramach którego wykorzystywane są jedynie informacje zawarte w polu SubjectPublicKeyInfo w certyfikacie X.509 w wersji 3. Umożliwia to zmniejszenie rozmiaru struktury klucza publicznego i uproszczenie przetwarzania, jednak usuwa również wszelkie gwarancje, że klucz publiczny należy do konkretnego podmiotu. Aby zapewnić uwierzytelnienie podczas korzystania z tego rozszerzenia, należy użyć wiązania poza pasmem między kluczem publicznym a podmiotem.

3.5 UWIERZYTELNIANIE KLIENTA

W przypadkach, gdy konieczne jest silne kryptograficzne uwierzytelnienie klienta, serwery TLS mogą wykorzystać opcję uwierzytelnienia klienta protokołu TLS, aby zażądać od klienta certyfikatu w celu jego kryptograficznego uwierzytelnienia[27]. Na przykład certyfikat identyfikacji tożsamości (*ang. Personal Identity Verification – PIV*) [47] i związany z nim klucz prywatny, stanowi odpowiednią opcję silnego uwierzytelniania pracowników i wykonawców instytucji rządowych. Aby zapewnić organizacjom możliwość pełnego wykorzystania kart PIV, we wszystkich serwerach TLS służących do uwierzytelniania klienta **należy** wdrożyć uwierzytelnianie oparte na certyfikatach.

Opcja uwierzytelniania klienta wymaga, aby dla serwera wdrożony został mechanizm walidacji ścieżki X.509 oraz magazyn kotwic zaufania. Wymagania dotyczące tych mechanizmów określono odpowiednio w podrozdziale 3.5.1 oraz 3.5.2. Aby uwierzytelnienie kryptograficzne rzeczywiście skutkowało silnym uwierzytelnieniem, klucze klienta **powinny** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów. W podrozdziale 3.5.3 opisano mechanizmy, które mogą przyczynić się, przynajmniej pośrednio, do egzekwowania tego wymogu. W podrozdziale 3.5.4 opisano sposób korzystania przez klienta z listy wskazówek serwera.

[27] Komunikat CertificateVerify jest wysyłany w ramach protokołu uzgodnienia, aby jednoznacznie zweryfikować certyfikat klienta, który posiada zdolność do podpisywania. W wersji TLS 1.1 (i TLS 1.0) dla tego komunikatu wykorzystuje się algorytm SHA-1 do wygenerowania podpisu we wszystkich komunikatach w ramach protokołu uzgodnienia, które pojawiły się przed nim. W publikacji SP 800-131A [10] stwierdzono, że stosowanie algorytmu SHA-1 do generowania podpisu cyfrowego jest niedozwolone po 2013 roku. Nawet w przypadku stwierdzenia kolizji klient musi użyć swojego klucza prywatnego do uwierzytelnienia się poprzez podpisanie skrótu. Ze względu na wartości losowe klienta i serwera, serwer, klient lub strona trzecia nie może użyć kolidującego zestawu komunikatów do podszywania się pod klienta lub serwer w kolejnych połączeniach. Każda modyfikacja tego komunikatu, komunikatów poprzedzających lub kolejnych ostatecznie spowoduje niepowodzenie w nawiązaniu połączenia. Dlatego dopuszczalne jest stosowanie algorytmu SHA-1 do generowania podpisów cyfrowych w komunikacie CertificateVerify w ramach wykonywania protokołu TLS.

Dla serwera TLS **powinna** istnieć możliwość wprowadzenia konfiguracji powodującej zakończenie połączenia po wystąpieniu alertu o krytycznym niepowodzeniu protokołu uzgodnienia, jeżeli zażądano certyfikatu klienta, a klient nie posiada odpowiedniego certyfikatu.

3.5.1 SPRAWDZANIE POPRAWNOŚCI ŚCIEŻKI

Certyfikat klienta **powinien** być weryfikowany zgodnie z zasadami sprawdzania poprawności ścieżki certyfikacji określonymi w punkcie 6 dokumentu [19]. Ponadto status unieważnienia każdego certyfikatu na ścieżce certyfikacji **powinien** być weryfikowany za pomocą protokołu weryfikacji statusu certyfikatów w trybie on-line (*ang. Online Certificate Status Protocol – OCSP*) lub listy unieważnień certyfikatów (*ang. Certificate Revocation List – CRL*). Kontrola OCSP **powinna** być przeprowadzana zgodnie z dokumentem RFC 6960 [63].

Informacje o unieważnieniu **powinny** być uzyskiwane w sposób opisany w podrozdziale 3.2.2.

Serwer **powinien** mieć możliwość określenia zasad certyfikatu, ze względu na które certyfikat klienta jest zaufany, poprzez zastosowanie zasad sprawdzania poprawności ścieżki certyfikacji określonych w punkcie 6 dokumentu RFC 5280 [19]. Aplikacje serwerowe i wewnętrzne mogą wykorzystać wyniki takiej kontroli do zaakceptowania lub odrzucenia certyfikatu. Sprawdzanie zasad certyfikatów daje serwerowi gwarancję, że akceptowane są tylko te certyfikaty klienta, które zostały wydane przy akceptowalnym poziomie wiarygodności, jeśli chodzi o CA oraz bezpieczeństwo systemu i procesu rejestracji.

Nie wszystkie produkty komercyjne mogą obsługiwać sprawdzanie poprawności ścieżki certyfikacji klucza publicznego oraz reguły przetwarzania zasad certyfikatów wymienione i cytowane powyżej. Wdrażając uwierzytelnianie klienta, instytucje państwowe **powinny** wykorzystać produkty komercyjne spełniające niniejsze wymagania lub rozszerzyć produkty komercyjne w celu spełnienia tych wymagań.

Serwer **powinien** być w stanie dostarczyć certyfikat klienta oraz zasady certyfikatów, zgodnie z którymi ścieżka certyfikacji klienta jest ważna, do aplikacji użytkujących w celu wspomagania decyzji w zakresie kontroli dostępu.

3.5.2 MAGAZYN KOTWIC ZAUFANIA

Posiadanie nadmiernej liczby kotwic zaufania zainstalowanych w aplikacji TLS może narazić aplikację na wszystkie PKI wywodzące się z tych kotwic zaufania. Najlepszym sposobem na zminimalizowanie tego zagrożenia jest umieszczenie w magazynie kotwic zaufania tylko tych, które są absolutnie niezbędne do uwierzytelnienia klienta za pomocą certyfikatu z kluczem publicznym.

Serwer **powinien** być skonfigurowany wyłącznie przy użyciu kotwic zaufania, którym ufa właściciel systemu, a dodatkowo tylko tych, które są wymagane do uwierzytelniania klientów w przypadku, gdy serwer obsługuje uwierzytelnianie klientów w ramach protokołu TLS. Administratorzy systemu serwera TLS, który obsługuje uwierzytelnianie klientów oparte na certyfikatach, **powinni** przeprowadzić analizę wystawców certyfikatów klientów i wykorzystać te informacje do określenia minimalnego zestawu kotwic zaufania wymaganych dla serwera. Taki zestaw jest zwykle niewielkim podzbiorem kotwic zaufania, które mogą być domyślnie obecne na serwerze. Należy również zauważyć, że taki magazyn kotwic zaufania różni się od magazynu kotwic zaufania maszyny. W związku z tym należy sprawdzić domyślny zestaw kotwic zaufania, aby określić, czy którakolwiek z nich jest konieczna do uwierzytelnienia klientów. Konieczne może być dodanie kotwic zaufania konkretnych organizacji i/lub usług PKI.

3.5.3 SPRAWDZANIE ROZMIARU KLUCZA KLIENTA

Jedynym bezpośrednim mechanizmem umożliwiającym serwerowi sprawdzenie, czy rozmiar klucza i algorytmy przedstawione w certyfikacie klucza publicznego klienta są akceptowalne, jest zbadanie przez serwer klucza publicznego i algorytmu w certyfikacie klienta. Pośrednim mechanizmem jest sprawdzenie, czy rozszerzenie zasad certyfikatu w certyfikacie klucza publicznego klienta wskazuje minimalną siłę kryptograficzną użytych algorytmów podpisu i skrótu, oraz wykonanie przez serwer przetwarzania i kontroli zasad certyfikatu. Serwer **powinien** sprawdzić długość klucza klienta, jeśli przeprowadzane jest

jego uwierzytelnianie, a implementacja serwera zapewnia mechanizm służący do tego celu. Instytucje państwowe **powinny** stosować wytyczne dotyczące rozmiaru klucza podane w publikacji NIST SP 800-131A [10] w celu sprawdzenia rozmiaru klucza klienta.

3.5.4 LISTA WSKAZÓWEK SERWERA

Urządzenia klienckie mogą korzystać z listy kotwic zaufania przesyłanej przez serwer w komunikacie CertificateRequest, aby określić, czy ścieżka certyfikacji klienta kończy się na jednej z tych kotwic zaufania. Lista wysyłana przez serwer jest znana jako „lista wskazówek". Gdy serwer i klient znajdują się w różnych domenach PKI, a zaufanie jest ustanawiane poprzez bezpośrednią certyfikację krzyżową pomiędzy dwoma domenami PKI (tj. domeną PKI serwera i domeną PKI klienta) lub poprzez przechodnią certyfikację krzyżową (tj. poprzez certyfikację krzyżową pomiędzy wieloma domenami PKI), klient może błędnie zdecydować, że jego certyfikat nie zostanie zaakceptowany przez serwer, ponieważ kotwica zaufania klienta nie została przesłana w liście wskazówek. Aby uniknąć takiego niepowodzenia, serwer **powinien**: 1) utrzymywać kotwice zaufania różnych PKI, których subskrybenci są potencjalnymi klientami dla serwera i uwzględniać je w liście wskazówek albo 2) być skonfigurowany do wysyłania pustej listy wskazówek, tak aby klient mógł zawsze przedstawić certyfikat, który posiada. Lista wskazówek **powinna** być niezależna od magazynu kotwic zaufania serwera[28]. Innymi słowy, serwer **powinien** nadal umieszczać w swoim magazynie kotwic zaufania tylko te kotwice, które dotyczą domeny PKI serwera i domen, którym musi bezpośrednio zaufać w celu uwierzytelnienia klienta. Należy zauważyć, że różnica między listą wskazówek serwera a własnym magazynem zaufania serwera jest następująca: 1) lista wskazówek to lista kotwic zaufania, którym potencjalny klient może zaufać, a 2) magazyn zaufania serwera to lista kotwic zaufania, którym serwer wyraźnie ufa.

[28] W zależności od kotwic zaufania serwera i klienta, obie listy mogą być identyczne, mogą mieć pewne wspólne kotwice zaufania lub nie mieć żadnych wspólnych kotwic.

3.6 WZNAWIANIE SESJI I WCZESNE DANE

Poprzednie sesje TLS mogą zostać wznowione, co umożliwia nawiązanie połączenia przy użyciu skróconego protokołu uzgodnienia. Wszystkie wersje protokołu TLS umożliwiają wznowienie sesji, choć mechanizm przeprowadzania wznowienia jest różny. Serwer może być skonfigurowany tak, aby ignorował żądania wznowienia sesji, jeśli pozwala na to implementacja.

Opracowano dodatkowe mechanizmy wznawiania sesji, takie jak rozszerzenie bezstanowego wznawiania sesji TLS (*ang. Stateless TLS Session Resumption*) [62]. Niniejsze rekomendacje nie mają na celu zachęcenia ani zniechęcenia do stosowania takich mechanizmów, jednak ważne jest, aby zrozumieć wpływ na zabezpieczenia w przypadku naruszenia bezpieczeństwa kluczy długoterminowych lub wspólnych. Jeśli wznowienie jest dozwolone, zaleca się odpowiednio częstą wymianę kluczy i krótki czas życia informacji o wznowieniu. Omówienie wpływu mechanizmów wznawiania na bezpieczeństwo znajduje się w publikacji [67].

Protokół TLS 1.3 umożliwia klientowi wysyłanie danych (znanych jako dane 0-RTT) w pierwszych komunikatach protokołu uzgodnienia. Taka praktyka może stanowić okazję dla atakujących, np. na przeprowadzenie ataku powtórzeniowego[29].
W specyfikacji protokołu TLS 1.3 opisano dwa mechanizmy niwelujące zagrożenia związane z danymi 0-RTT. Jednym z takich mechanizmów jest bilet jednorazowy, który polega na tym, że każdy bilet sesji może być wykorzystany tylko raz. Implementacja tego mechanizmu w środowisku z rozproszonymi serwerami może być trudna, ponieważ baza danych sesji musi być współużytkowana przez serwery. Rejestracja komunikatu ClientHello to drugi mechanizm, który chroni przed atakami powtórzeniowymi poprzez zapisywanie unikalnej wartości pochodzącej z wiadomości ClientHello i odrzucanie duplikatów. Aby ograniczyć rozmiar listy, serwer może ją utrzymywać tylko w określonym przedziale czasowym. Zasadniczo dane 0-RTT **nie**

[29] Protokół TLS nie zapewnia bezpośredniej ochrony przed atakami powtórzeniowymi związanymi z danymi 0-RTT.

powinny być akceptowane przez serwer. Jeżeli serwer zezwala na przesyłanie danych 0-RTT, to powinien korzystać z mechanizmu biletów jednorazowych zgodnie z dokumentem RFC 8446 (patrz punkt 8 dokumentu [57]).

3.7 METODY KOMPRESJI

Zastosowanie kompresji może umożliwić atakującym przeprowadzenie ataków typu side-channel opartych na kompresji (np. [60], [11]). W celu obrony przed tymi atakami **należy** stosować metodę kompresji zerowej (*ang. null compression*), a wszystkie inne metody kompresji **powinny** być wyłączone.

3.8 ASPEKTY OPERACYJNE

W powyższych podrozdziałach opisano funkcje specyficzne dla protokołu TLS. Funkcje te są niezbędne, ale nie wystarczające do osiągnięcia bezpieczeństwa w środowisku operacyjnym.

Instytucje państwowe **powinny** zadbać o to, aby serwery TLS zawierały odpowiednie zabezpieczenia sieciowe określone w innych rekomendacjach, takich jak NIST SP 800-53[30] [36].

Serwer powinien działać w oparciu o bezpieczny system operacyjny[31]. Jeżeli serwer opiera się na module kryptograficznym z certyfikatem FIPS 140 poziomu 1, oprogramowanie i klucz prywatny **powinny** być chronione za pomocą mechanizmów identyfikacji, uwierzytelniania i kontroli dostępu systemu operacyjnego. W niektórych bardzo wrażliwych zastosowaniach klucze prywatne serwera mogą wymagać ochrony za pomocą sprzętowego modułu kryptograficznego z certyfikatem FIPS 140 poziomu 2 lub wyższego.

Serwer i związana z nim platforma **powinna** być na bieżąco aktualizowana pod względem poprawek bezpieczeństwa. Ma to kluczowe znaczenie dla różnych aspektów bezpieczeństwa.

---

[30] Polskojęzyczne opracowanie: NSC 800-53.

[31] Bezpieczny system operacyjny zawiera i wykorzystuje następujące funkcje: ochrona systemu operacyjnego przed aplikacjami i procesami, izolacja aplikacji i procesów za pośrednictwem systemu operacyjnego, identyfikacja i uwierzytelnianie użytkowników, kontrola dostępu oparta na uwierzytelnionej tożsamości użytkownika oraz rejestrowanie zdarzeń dotyczących działań istotnych dla bezpieczeństwa.

4. MINIMALNE WYMAGANIA DLA KLIENTÓW TLS

Rozdział ten zawiera zestaw minimalnych wymagań, które musi spełniać klient TLS, aby zachować zgodność z niniejszymi wytycznymi. Wymogi zostały uporządkowane w następujący sposób: obsługiwane wersje protokołu TLS, klucze i certyfikaty klienta, obsługa kryptografii, obsługa rozszerzeń protokołu TLS, uwierzytelnianie serwera, wznawianie sesji, metody kompresji oraz aspekty operacyjne.

Wymagania szczegółowe zostały określone jako wymagania dotyczące wdrożenia lub wymagania dotyczące konfiguracji. Zgodnie z wymaganiami dotyczącymi implementacji instytucje państwowe **nie powinny** zamawiać implementacji klienta TLS, chyba że obejmuje ona wymagane funkcje. Zgodnie z wymaganiami dotyczącymi konfiguracji, administratorzy systemów są zobowiązani do sprawdzenia, czy poszczególne funkcje są włączone, a w niektórych przypadkach odpowiednio skonfigurowane, jeśli są dostępne.

4.1 OBSŁUGIWANE WERSJE PROTOKOŁU

Klient **powinien** być skonfigurowany do korzystania z protokołu TLS 1.2, a także TLS 1.3. Klient może być skonfigurowany do korzystania z protokołu TLS 1.1, a także TLS 1.0, aby ułatwić komunikację z serwerami sektora prywatnego. Klient **nie powinien** być skonfigurowany do korzystania z protokołu SSL 2.0 lub SSL 3.0. Instytucje państwowe **powinny** być gotowe do obsługi protokołu TLS 1.3 do dnia 1 stycznia 2024 r. Po tej dacie klienci **powinni** być skonfigurowani do korzystania z protokołu TLS 1.3. Zasadniczo klienci, którzy obsługują protokół TLS 1.3, powinni być skonfigurowani tak, aby korzystali również z TLS 1.2. Można jednak zablokować możliwość korzystania z protokołu TLS 1.2 na klientach obsługujących TLS 1.3, jeżeli wersja TLS 1.2 nie jest wymagana ze względu na interoperacyjność.

4.2 KLUCZE I CERTYFIKATY KLIENTA

Niektóre aplikacje mogą wymagać uwierzytelnienia klienta. W przypadku protokołu TLS można tego dokonać poprzez wykonanie wzajemnego uwierzytelnienia z wykorzystaniem certyfikatów.

4.2.1 PROFIL CERTYFIKATU KLIENTA

W przypadku konieczności uwierzytelnienia klienta na podstawie certyfikatu, klient **powinien** być skonfigurowany przy użyciu certyfikatu zgodnego z zaleceniami przedstawionymi w tym podrozdziale. Certyfikat klienta może być skonfigurowany w systemie lub znajdować się na urządzeniu zewnętrznym (np. na karcie PIV). Aby spełnić niniejsze specyfikacje, certyfikat klienta TLS **powinien** być certyfikatem X.509 w wersji 3. Zarówno klucz publiczny zawarty w certyfikacie, jak i podpis **powinny** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów. Jeśli klient obsługuje wersje protokołu TLS wcześniejsze niż TLS 1.2, certyfikat powinien być podpisany przy pomocy algorytmu zgodnego z kluczem publicznym[32]:

- Certyfikaty zawierające klucze publiczne RSA (podpis), ECDSA lub DSA **powinny** być podpisane odpowiednio przy pomocy tych samych algorytmów podpisu.
- Certyfikaty zawierające certyfikaty Diffiego-Hellmana **powinny** być podpisywane za pomocą algorytmu DSA.
- Certyfikaty zawierające klucze publiczne ECDH **powinny** być podpisane za pomocą algorytmu ECDSA.

Profil certyfikatu klienta został przedstawiony w tabeli 4-1. W przypadku braku wymogów dotyczących profilu certyfikatu klienta specyficznych dla danej instytucji państwowej, **należy** stosować niniejszy profil.

[32] Zalecenie to jest artefaktem wymagań dla protokołu TLS 1.0 i 1.1.

Tabela 4-1: Profil certyfikatu klienta TLS

Pole	Krytyczność	Wartość	Opis
Wersja	Nd.	2	Wersja 3
Numer seryjny	Nd.	Unikalna dodatnia liczba całkowita	Musi być unikalna
		Wartości zależne od typu klucza CA:	
		sha256WithRSAEncryption {1 2 840 113549 1 1 11} lub silniejsza	CA z kluczem RSA
		id-RSASSA-PSS {1 2 840 113549 1 1 10}	CA z kluczem RSA
		ecdsa-with-SHA256 {1 2 840 10045 4 3 2} lub silniejsza	CA z kluczem opartym na krzywej eliptycznej
		id-dsa-with-sha256 {2 16 840 1 101 3 4 3 2} lub silniejsza	CA z kluczem DSA
Nazwa wyróżniająca wystawcy	Nd.	Unikalna nazwa DN CA wystawiającego zgodna ze standardem X.500	W każdej nazwie RDN powinna być zakodowana pojedyncza wartość. Wszystkie atrybuty typu directoryString powinny być zakodowane jako printable-string.
Okres ważności	Nd.	3 lata lub mniej	Daty do roku 2049 wyrażone w czasie UTC
Nazwa wyróżniająca podmiotu	Nd.	Unikalna nazwa DN podmiotu zgodna ze standardem X.500 i wymogami danej instytucji państwowej	W każdej nazwie RDN **powinna** być zakodowana pojedyncza wartość. Wszystkie atrybuty typu directoryString **powinny** być zakodowane jako printable-string.
	Nd.	*Wartości według typu certyfikatu:*	

Pole	Krytyczność	Wartość	Opis
Informacje o kluczu publicznym podmiotu		rsaEncryption {1 2 840 113549 1 1 1}	Certyfikat podpisu RSA 2048-bitowy moduł klucza RSA lub inne zatwierdzone długości, jak określono w publikacjach [45] i [5] Parametry: NULL
		ecPublicKey {1 2 840 10045 2 1}	Certyfikat podpisu ECDSA lub certyfikat ECDH Parametry namedCurve OID dla nazwanej krzywej określonej w publikacji NIST SP 800-186[33]. Powinna być to krzywa P-256 lub P – 384. Klucz SubjectPublic: Nieskompresowany punkt EC
		id-dsa {1 2 840 10040 4 1}	Certyfikat podpisu DSA Parametry: p, q, g.
		dhpublicnumber {1 2 840 10046 2 1}	Certyfikat DH Parametry: p, g, q
Podpis wystawcy	Nd.	Wartość taka sama jak w przypadku algorytmu podpisu wydawcy	
Rozszerzenia			
Identyfikator klucza urzędu	Nie	Octet String	Taki sam jak identyfikator klucza podmiotu w certyfikacie CA będącego wydawcą Zabronione: DN wystawcy, krotka numeru seryjnego

[33] Zalecane krzywe eliptyczne wymienione obecnie w standardzie FIPS 186-4 [45] zostaną przeniesione do publikacji SP 800-186. Do czasu pojawienia się publikacji SP 800-186, przy wyborze zalecanych krzywych eliptycznych należy kierować się standardem FIPS 186-4.

Pole	Krytyczność	Wartość	Opis
Identyfikator klucza podmiotu	Nie	Octet String	Taki sam jak w standardzie PKCS-10 lub obliczony przez wydający CA
Użycie klucza	Tak	digitalSignature	Certyfikat RSA, certyfikat DSA, certyfikat ECDSA
		keyAgreement	Certyfikat ECDH, certyfikat DH
		id-kp-clientAuth {1 3 6 1 5 5 7 3 2}	Wymagane
		anyExtendedKeyUsage {2 5 29 37 0}	Identyfikator OID anyExtendedKeyUsage powinien być obecny, jeśli rozszerzenie dotyczące rozszerzonego użycia klucza jest obecne, ale nie ma potrzeby ograniczania typów aplikacji, dla których certyfikat może być używany (np. jest to certyfikat uwierzytelniający ogólnego przeznaczenia).
			Zabronione: anyExtendedKeyUsage; wszystkie inne, chyba że są zgodne z użyciem klucza rozszerzonego
Zasady certyfikatu	Nie	Zasady certyfikatów zgodne ze standardem X.509 wystawcy	
Alternatywna nazwa podmiotu	Nie	Adres e-mail zgodny z RFC 822, uniwersalna nazwa użytkownika (Universal Principal Name – UPN), nazwa DNS i/lub inne	Opcjonalne

Pole	Krytyczność	Wartość	Opis
Dostęp do informacji o urzędzie	Nie	id-ad-caIssuers	Wymagane. Wpis dotyczący metody dostępu zawiera adres URL HTTP certyfikatów wydanych dla CA będącego wydawcą
		id-ad-ocsp	Opcjonalne. Wpis dotyczący metody dostępu zawiera adres URL HTTP obiektu odpowiadającego OCSP wydającego CA
Punkty dystrybucji listy CRL	Nie	Patrz uwagi	Opcjonalne: Wartość HTTP w polu distributionPoint kierująca do pełnej i kompletnej listy CRL. Zabronione: pola reasons I cRLIssuer, oraz nameRelativetoCRLIssuer CHOICE

Jeżeli klient ma wiele certyfikatów, które spełniają wymagania serwera TLS, klient TLS (np. przeglądarka) może poprosić użytkownika o dokonanie wyboru z listy certyfikatów. Rozszerzenie dotyczące rozszerzonego użycia klucza (*ang. Extended Key Usage – EKU*) ogranicza operacje, do których mogą być użyte klucze w certyfikacie, a więc użycie rozszerzenia EKU w certyfikatach klienta może wyeliminować konieczność dokonywania tego wyboru. Jeżeli rozszerzenie EKU jest zawarte w certyfikatach klienta, to identyfikator OID celu klucza id-kp-client-auth **powinien** być zawarty w certyfikatach używanych do uwierzytelniania klienta TLS i **powinien** być pominięty we wszystkich innych certyfikatach.

Certyfikaty klientów są również filtrowane przez klientów TLS na podstawie możliwości zbudowania ścieżki do jednej z kotwic zaufania w liście wskazówek wysyłanych przez serwer, jak opisano w podrozdziale 3.5.4.

4.2.2 UZYSKIWANIE INFORMACJI O STANIE UNIEWAŻNIENIA DLA CERTYFIKATU SERWERA

Klient **powinien** przeprowadzić sprawdzanie stanu unieważnienia certyfikatu serwera. Informacje o unieważnieniu mogą być uzyskane przez klienta z jednej z następujących lokalizacji:

1. Odpowiedź OCSP lub odpowiedź w komunikacie CertificateStatus serwera ([29], [54]) (lub komunikacie Certificate w protokole TLS 1.3).
2. Lista unieważnień certyfikatów (*ang. Certificate Revocation List – CRL*) lub odpowiedź OCSP w lokalnym magazynie certyfikatów klienta.
3. Odpowiedź OCSP z lokalnie skonfigurowanego obiektu odpowiadającego OCSP.
4. Odpowiedź OCSP z lokalizacji obiektu odpowiadającego OCSP zidentyfikowanej w polu OCSP w rozszerzeniu dotyczącym dostępu do informacji o urzędzie (*ang. Authority Information Access*) w certyfikacie serwera.
5. Lista CRL z rozszerzenia dotyczącego punktu dystrybucji listy CRL (*ang. CRL Distribution Point*) w certyfikacie serwera.

Jeśli serwer nie podaje statusu unieważnienia, w lokalnym magazynie certyfikatów nie ma aktualnej lub wiążącej listy CRL lub odpowiedzi OCSP, a obiekt odpowiadający OCSP i punkt dystrybucji listy CRL są niedostępne lub nieosiągalne w czasie nawiązywania sesji TLS, klient przerwie połączenie, albo zaakceptuje potencjalnie unieważniony certyfikat lub certyfikat o naruszonym bezpieczeństwie. Decyzja o przyjęciu lub odrzuceniu certyfikatu w takiej sytuacji **powinna** być podjęta zgodnie z polityką danej instytucji państwowej.

4.2.3 WIARYGODNOŚĆ CERTYFIKATU KLUCZA PUBLICZNEGO KLIENTA

Certyfikat klucza publicznego klienta może być zaufany przez serwery na podstawie zasad, procedur i środków bezpieczeństwa stosowanych przy jego wydawaniu, jak opisano w podrozdziale 3.5.1. Na przykład, w niniejszych rekomendacjach zaleca się, aby certyfikat uwierzytelniania PIV był normą dla uwierzytelniania pracowników rządowych i długoterminowych wykonawców. Zasady certyfikatów uwierzytelniających PIV zostały określone w dokumencie *Federal PKI Common Policy Framework* [31], a zasady certyfikatów uwierzytelniających PIV-I zostały określone w standardzie X.509, *Certificate Policy for the Federal Bridge Certification Authority* [68][34]. W zależności od wymagań aplikacji po stronie serwera, dopuszczalne mogą być również inne zasady certyfikatów. Porady dotyczące innych zasad certyfikatów są poza zakresem niniejszych rekomendacji.

4.3 OBSŁUGA KRYPTOGRAFII

4.3.1 ZESTAWY SZYFROWANIA

Dopuszczalne zestawy szyfrowania dla klienta TLS są takie same jak dla serwera TLS. Zestawy szyfrowania ogólnego przeznaczenia zostały wymienione w podrozdziale 3.3.1. Zestawy szyfrowania odpowiednie dla środowisk z kluczami wstępnymi dla TLS 1.2 i wcześniejszych wersji zostały wymienione w załączniku C. Aplikacje, które wymagają transportu klucza RSA jako metody wymiany kluczy, mogą podczas okresu przejściowego używać zestawów szyfrowania wymienionych w załączniku D. W przypadku gdy klucze efemeryczne są wykorzystywane do ustanowienia głównego klucza tajnego, każda para kluczy efemerycznych (tj. para kluczy efemerycznych serwera i para kluczy efemerycznych klienta) **powinna** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów.

Klient nie powinien być skonfigurowany do korzystania z zestawów szyfrowania innych niż wymienione w podrozdziale 3.3.1, załączniku C lub załączniku D[35].

[34] Do wykorzystania przez zainteresowanych

[35] Wymogi dotyczące zestawów szyfrowania dla klientów są mniej restrykcyjne niż dla serwerów, ponieważ dla wielu klientów, takich jak przeglądarki internetowe, może nie być możliwy taki sam poziom konfiguracji jak w przypadku serwerów.

Aby zapobiegać atakom na tryb CBC, implementacje TLS, które obsługują wersje wcześniejsze niż TLS 1.3, **powinny** używać błędu bad_record_mac do sygnalizowania błędu dopełnienia. Implementacje **powinny** obliczać kod MAC niezależnie od tego, czy występują błędy dopełnienia. Implementacje TLS **powinny** obsługiwać deszyfrowanie w czasie stałym lub prawie stałym. Nie dotyczy to implementacji TLS 1.3, ponieważ nie obsługują one zestawów szyfrowania wykorzystujących tryb CBC.

4.3.2 WALIDACJA KRYPTOGRAFII

Klient **powinien** stosować techniki kryptografii poddane walidacji tak, jak zostało to opisane dla serwera w podrozdziale 3.3.3.

Do generowania losowych bajtów (32 bajty w TLS 1.3; 28 bajtów we wcześniejszych wersjach protokołu TLS) wartości losowej klienta **należy** stosować poddany walidacji generator liczb losowych. Poddanego walidacji generatora liczb losowych **należy** również używać do generowania 4-bajtowego znacznika czasu wartości losowej klienta w przypadku wersji protokołu TLS wcześniejszych niż TLS 1.3.

4.4 OBSŁUGA ROZSZERZEŃ PROTOKOŁU TLS

Zasadniczo zaleca się, aby klienci byli skonfigurowani do obsługi rozszerzeń, które są wymagane do interoperacyjności lub zwiększają bezpieczeństwo. **Nie należy** stosować rozszerzeń, które nie są wymagane.

4.4.1 OBOWIĄZKOWE ROZSZERZENIA PROTOKOŁU TLS

Klient **powinien** być skonfigurowany do wykorzystywania następujących rozszerzeń:

1. Sygnalizacja renegocjacji (*ang. Renegotiation Indication*)
2. Identyfikacja nazwy serwera (*ang. Server Name Indication*)
3. Rozszerzony główny klucz tajny (*ang. Extended Master Secret*)
4. Algorytmy podpisu (*ang. Signature Algorithms*)
5. Żądanie statusu certyfikatu (*ang. Certificate Status Request*)

4.4.1.1 SYGNALIZACJA RENEGOCJACJI (ANG. RENEGOTIATION INDICATION)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie sygnalizacji renegocjacji jest wymagane przez niniejsze rekomendacje, jak opisano w podrozdziale 3.4.1.1. Klienci **powinni** wykonywać początkowe i kolejne renegocjacje zgodnie z dokumentem RFC 5746 [59].

4.4.1.2 IDENTYFIKACJA NAZWY SERWERA (*ANG. SERVER NAME INDICATION*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Rozszerzenie identyfikacji nazwy serwera zostało opisane w podrozdziale 3.4.1.2. Klient **powinien** być w stanie dołączyć to rozszerzenie do komunikatu ClientHello, jak opisano w dokumencie RFC 6066 [29].

4.4.1.3 ROZSZERZONY GŁÓWNY KLUCZ TAJNY (*ANG. EXTENDED MASTER SECRET*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie polegające na rozszerzeniu głównego klucza tajnego (*ang. Extended Master Secret*) opisane w podrozdziale 3.4.1.3 zapobiega atakom typu „man-in-the-middle" poprzez powiązanie głównego klucza tajnego ze skróconym rejestrem pełnego protokołu uzgodnienia. Klient **powinien** obsługiwać to rozszerzenie.

4.4.1.4 ALGORYTMY PODPISU (ANG. SIGNATURE ALGORITHMS)

Dotyczy protokołu TLS w wersji: 1.2, 1.3

Klienci **powinni** stosować dopuszczalne pary algorytmów skrótu i podpisu w ramach tego rozszerzenia w komunikatach ClientHello wysyłanych w protokole TLS 1.2 i TLS 1.3. Rozszerzenie, jego składnia i zasady przetwarzania są opisane w podpunktach 7.4.1.4.1, 7.4.4, 7.4.6 i 7.4.8 dokumentu RFC 5246 [25] i podpunkcie 4.2.3 dokumentu RFC 8446 [57]. Należy zwrócić uwagę, że rozszerzenie opisane w dokumencie RFC 8446 aktualizuje rozszerzenie opisane w RFC 5246 poprzez dodanie dodatkowego schematu podpisu.

4.4.1.5 ŻĄDANIE STATUSU CERTYFIKATU (ANG. CERTIFICATE STATUS REQUEST)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Klient **powinien** uwzględniać rozszerzenie „status_request" w komunikacie ClientHello.

4.4.2 WARUNKOWE ROZSZERZENIA PROTOKOŁU TLS

Klient TLS obsługuje następujące rozszerzenia protokołu TLS w opisanych okolicznościach:

1. Awaryjne sygnalizowanie wartości zestawu szyfrowania (*ang. Fallback Signaling Cipher Suite Value – SCSV*) powinno być obsługiwane, jeżeli klient obsługuje wersje TLS wcześniejsze niż TLS 1.2 i nie obsługuje TLS 1.3.
2. Rozszerzenie obsługiwanych grup (*ang. Supported Groups*) powinno być obsługiwane, jeżeli klient obsługuje efemeryczne zestawy szyfrowania ECDH lub jeżeli klient obsługuje protokół TLS 1.3.
3. Rozszerzenie udostępniania klucza (*ang. Key Share*) powinno być obsługiwane, jeżeli klient obsługuje protokół TLS 1.3.
4. Rozszerzenie formatów przecinkowych EC (*EC Point Format*) powinno być obsługiwane, jeżeli klient obsługuje zestawy szyfrowania EC.
5. Rozszerzenie statusu wielu certyfikatów (*ang. Multiple Certificate Status*) powinno być obsługiwane, jeśli rozszerzenie to jest obsługiwane przez implementację klienta.
6. Rozszerzenie identyfikacji zaufanego CA (*ang. Trusted CA Indication*) powinno być obsługiwane przez klientów, którzy działają na urządzeniach o ograniczonej pamięci, gdzie przechowywana jest tylko niewielka liczba kluczy głównych CA.
7. Rozszerzenie Encrypt-then-MAC powinno być obsługiwane, jeżeli skonfigurowano zestawy szyfrowania w trybie CBC.
8. Rozszerzenie skróconego kodu HMAC (*ang. Truncated HMAC*) może być obsługiwane przez klientów, którzy działają na urządzeniach o ograniczonych zasobach, gdy nie jest obsługiwane dopełnianie zmiennej długości i są obsługiwane zestawy szyfrowania, które wykorzystują tryb CBC.
9. Rozszerzenie klucza wstępnego (*ang. Pre-Shared Key*) może być obsługiwane przez klientów TLS 1.3.
10. Rozszerzenie trybów wymiany klucza wstępnego (*ang. Pre-Shared Key Exchange Modes*) powinno być obsługiwane przez klientów TLS 1.3, jeśli obsługują oni rozszerzenie kluczy wstępnych.
11. Rozszerzenie obsługiwanych wersji (*Supported Versions*) powinno być obsługiwane przez klientów TLS 1.3.

12. Rozszerzenie Cookie powinno być obsługiwane przez klientów TLS 1.3.
13. Rozszerzenie algorytmów podpisu certyfikatu (*ang. Certificate Signature Algorithms*) powinno być obsługiwane, jeżeli klient obsługuje protokół TLS 1.3 i TLS 1.2.
14. Rozszerzenie uwierzytelniania klienta po protokole uzgodnienia (*ang. Post-handshake Client Authentication*) powinno być obsługiwane, jeżeli klient obsługuje protokół TLS 1.3.

4.4.2.1 AWARYJNE SYGNALIZOWANIE WARTOŚCI ZESTAWU SZYFROWANIA (*ANG. FALLBACK SIGNALING CIPHER SUITE VALUE – SCSV*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Wartość zestawu szyfrowania, opisana w podrozdziale 3.4.2.1, zapewnia mechanizm zapobiegający niezamierzonemu przejściu na starszy protokół TLS w wersjach wcześniejszych niż TLS 1.3. Klienci sygnalizują, kiedy połączenie jest awaryjne, a jeśli serwer obsługuje nowszą wersję protokołu TLS, zwraca alert krytyczny. Jeżeli klient nie obsługuje wersji TLS 1.3 i próbuje nawiązać połączenie przy użyciu wersji protokołu TLS wcześniejszej niż TLS 1.2, klient dołącza TLS_FALLBACK_SCSV na końcu listy zestawów szyfrowania w komunikacie ClientHello.

4.4.2.2 OBSŁUGIWANE GRUPY (ANG. SUPPORTED GROUPS)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2, 1.3

Rozszerzenie obsługiwanych grup (*ang. Supported Groups*, supported_groups) zostało opisane w podrozdziale 3.4.2.2. Implementacje klientów **powinny** wysyłać to rozszerzenie w komunikatach ClientHello w ramach protokołu TLS 1.3 oraz w komunikatach ClientHello, które zawierają efemeryczne zestawy szyfrowania ECDH. W przypadku skonfigurowania zestawów szyfrowania opartych na krzywych eliptycznych serwer **powinien** obsługiwać co najmniej jedną z zatwierdzonych przez NIST krzywych, P-256 (secp256r1) i P-384 (secp384r1), jak określono w dokumencie RFC 8422 [52]. Dodatkowe krzywe eliptyczne zalecane przez NIST zostały wymienione w załączniku D do publikacji NIST SP 800-56A [6]. Dopuszcza się obsługę grup ciał skończonych, które zostały zatwierdzone do użytku z protokołem TLS w załączniku D do publikacji NIST SP 800-56A.

4.4.2.3 UDOSTĘPNIANIE KLUCZA (*ANG. KEY SHARE*)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie udostępniania klucza (*ang. Key Share*) jest stosowane do przesyłania parametrów kryptograficznych. Klienci obsługujący protokół TLS 1.3 **powinni** obsługiwać to rozszerzenie zgodnie z opisem w punkcie 4.2.7 dokumentu RFC 8446 [57].

4.4.2.4 OBSŁUGIWANE FORMATY PRZECINKOWE (*ANG. SUPPORTED POINT FORMATS*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Klienci, którzy obsługują zestawy szyfrów EC w ramach protokołu TLS 1.2 i wcześniejszych wersji, **powinny** być w stanie określić obsługiwane formaty przecinkowe w komunikacie ClientHello zgodnie z podpunktem 5.1 dokumentu [52]. Klienci obsługujący zestawy szyfrów EC **powinni** obsługiwać przetwarzanie co najmniej jednego[36] z formatów przecinkowych EC otrzymanych w komunikacie ServerHello, jak opisano w podpunkcie 5.2 dokumentu [52].

4.4.2.5 STATUS WIELU CERTYFIKATÓW (ANG. MULTIPLE CERTIFICATE STATUS)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie statusu wielu certyfikatów zostało opisane w podrozdziale 3.4.2.5. Rozszerzenie to ulepsza rozszerzenie żądania statusu certyfikatu (*ang. Certificate Status Request*) opisane w podrozdziale 3.4.1.5, pozwalając klientowi na żądanie statusu wszystkich certyfikatów dostarczonych przez serwer w ramach protokołu uzgodnienia TLS. Rozszerzenie to zostało udokumentowane w dokumencie RFC 6961 [54]. Implementacje klientów, dla których istnieje taka możliwość, **powinny** być skonfigurowane tak, aby uwzględniały to rozszerzenie w komunikacie ClientHello.

4.4.2.6 IDENTYFIKACJA ZAUFANEGO CA (*ANG. TRUSTED CA INDICATION*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Klienci, którzy działają na urządzeniach o ograniczonej pamięci, gdzie przechowywana

[36] Musi być obsługiwany nieskompresowany format przecinkowy. Wszystkie pozostałe zostały wycofane z użytku w ramach protokołu TLS, jak opisano w podpunkcie 5.1.2 dokumentu RFC 8422 [52].

jest tylko niewielka liczba kluczy głównych CA, **powinni** być w stanie dołączyć rozszerzenie identyfikacji zaufanego CA (*ang*. trusted_ca_keys) do komunikatu ClientHello, jak opisano w dokumencie [29].

4.4.2.7 ENCRYPT-THEN-MAC

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie Encrypt-then-MAC opisane w podrozdziale 3.4.2.7 może łagodzić skutki lub zapobiegać kilku znanym atakom na zestawy szyfrowania CBC. Aby ta zmodyfikowana kolejność operacji mogła być stosowana, zarówno serwer, jak i klient muszą zaimplementować rozszerzenie Encrypt-then-MAC i wynegocjować jego użycie. Klienci, dla których skonfigurowano zestawy szyfrowania w trybie CBC, **powinny** obsługiwać to rozszerzenie, jak opisano w dokumencie RFC 7366 [33]. Klient **powinien** dołączyć to rozszerzenie do komunikatu ClientHello, jeśli zawiera on zestawy szyfrowania CBC.

4.4.2.8 SKRÓCONY KOD HMAC (*ANG. TRUNCATED HMAC*)

Dotyczy protokołu TLS w wersji: 1.0, 1.1, 1.2

Rozszerzenie skróconego kodu HMAC zostało opisane w podrozdziale 3.4.2.8.
Klienci działający na urządzeniach z ograniczonymi zasobami mogą obsługiwać to rozszerzenie. Rozszerzenie skróconego kodu HMAC (*ang. Truncated HMAC*) **nie powinno** być stosowane w połączeniu z dopełnianiem zmiennej długości ze względu na ataki opisane przez Patersona i wsp. [53]. Rozszerzenie to ma zastosowanie tylko wtedy, gdy obsługiwane są zestawy szyfrowania wykorzystujące tryby CBC.

4.4.2.9 KLUCZ WSTĘPNY (ANG. PRE-SHARED KEY)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie z kluczem wstępnym (*ang. Pre-Shared Key*, pre_shared_key) służy do identyfikacji klucza wstępnego, który ma być użyty do ustanowienia klucza PSK.
W ramach protokołu TLS 1.3 klucze wstępne mogą być ustanawiane poza pasmem, jak w TLS 1.2 lub wersjach poprzednich, albo w poprzednim połączeniu, w którym to przypadku są używane do wznowienia sesji. Klienci obsługujący protokół TLS 1.3 mogą być skonfigurowani do stosowania tego rozszerzenia w celu obsługi wznowienia sesji lub wykorzystania kluczy wstępnych, które są tworzone poza pasmem.

4.4.2.10 TRYBY WYMIANY KLUCZY WSTĘPNYCH (ANG. PRE-SHARED KEY EXCHANGE MODES)

Dotyczy protokołu TLS w wersji 1.3

Klient TLS 1.3 musi wysyłać rozszerzenie trybów wymiany kluczy wstępnych (*ang.* psk_key_exchange_modes), jeśli wysyła rozszerzenie z kluczem wstępnym (*ang. Pre-Shared Key*). W przeciwnym razie serwer przerwie wykonywanie protokołu uzgodnienia. Klienci TLS, którzy obsługują wersję TLS 1.3 i rozszerzenie z kluczem wstępnym (*ang. Pre-Shared Key*), **powinni** obsługiwać to rozszerzenie.

4.4.2.11 OBSŁUGIWANE WERSJE (ANG. SUPPORTED VERSIONS)

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie obsługiwanych wersji określa, które wersje protokołu TLS klient może negocjować. Klient TLS 1.3 **powinien** wysłać to rozszerzenie w komunikacie ClientHello.

4.4.2.12 COOKIE

Dotyczy protokołu TLS w wersji 1.3

Rozszerzenie Cookie pozwala serwerowi zmusić klienta do udowodnienia, że jest osiągalny pod swoim widocznym adresem sieciowym i przenieść informacje o stanie do klienta. Klienci obsługujący protokół TLS 1.3 **powinni** obsługiwać rozszerzenie Cookie zgodnie z dokumentem RFC 8446 [57].

4.4.2.13 ALGORYTMY PODPISU CERTYFIKATU (ANG. CERTIFICATE SIGNATURE ALGORITHMS)

Dotyczy protokołu TLS w wersji: 1.2, 1.3

Rozszerzenie dotyczące algorytmów podpisu certyfikatu (*ang. Certificate Signature Algorithms*, signature_algorithms_cert) określa algorytmy podpisu, które mogą być używane w certyfikatach. Umożliwia to podmiotowi żądającemu certyfikatu (klientowi lub serwerowi) zażądanie innych algorytmów podpisu dla certyfikatu niż dla protokołu uzgodnienia TLS. Klient może wysłać to rozszerzenie do serwera i może otrzymać to rozszerzenie od serwera, który żąda uwierzytelnienia klienta na podstawie certyfikatu. Rozszerzenie to nie musi być wysyłane, jeśli algorytmy z rozszerzenia algorytmów podpisu dotyczą również certyfikatów. Implementacje klientów TLS obsługujące

protokół TLS 1.3 **powinny** obsługiwać to rozszerzenie i **powinno** ono być obsługiwane również w przypadku protokołu TLS 1.2.

4.4.2.14 UWIERZYTELNIANIE KLIENTA PO PROTOKOLE UZGODNIENIA (*ANG. POST-HANDSHAKE CLIENT AUTHENTICATION*)

Dotyczy protokołu TLS w wersji 1.3

Klient wysyła rozszerzenie uwierzytelniania klienta po protokole uzgodnienia (*ang. Post-handshake Client Authentication*, post_handshake_auth), aby zasygnalizować, że może odpowiedzieć na zażądanie uwierzytelnienia klienta po wykonaniu protokołu uzgodnienia. Klienci TLS, którzy obsługują wersję TLS 1.3 mogą obsługiwać to rozszerzenie.

4.4.3 ODRADZANE ROZSZERZENIA PROTOKOŁU TLS

Rozszerzenia wymienione poniżej **nie powinny** być stosowane:

1. Adres URL certyfikatu klienta (*ang. Client Certificate URL*)
2. Wczesna identyfikacja danych (*ang. Early Data Indication*)

Rozszerzenie surowego klucza publicznego (*ang. Raw Public Key*) nie **powinno** być obsługiwane.

Powody, dla których odradza się stosowanie tych rozszerzeń, można znaleźć w podrozdziale 3.4.3.

4.5 UWIERZYTELNIANIE SERWERA

Klient **powinien** być w stanie zbudować ścieżkę certyfikacji dla certyfikatu serwera przedstawianego w ramach wykonywania protokołu uzgodnienia TLS z co najmniej jedną z kotwic zaufania w magazynie zaufania klienta, jeśli odpowiednia kotwica jest obecna w tym magazynie. Do budowy ścieżki certyfikacji klient może wykorzystać wszystkie lub podzbiór następujących zasobów: lokalny magazyn certyfikatów, certyfikaty otrzymane od serwera podczas wykonywania protokołu uzgodnienia, protokół dostępu do usług katalogowych (*ang. Lightweight Directory Access Protocol – LDAP*), zasoby zadeklarowane w polu repozytorium CA rozszerzenia dostępu do informacji o podmiocie w różnych certyfikatach CA oraz zasoby zadeklarowane w polu wystawcy CA rozszerzenia dostępu do informacji o urzędzie w różnych certyfikatach.

4.5.1 SPRAWDZANIE POPRAWNOŚCI ŚCIEŻKI

Klient **powinien** weryfikować certyfikat serwera zgodnie z zasadami sprawdzania poprawności ścieżki certyfikacji określonymi w punkcie 6 dokumentu [19]. Status unieważnienia każdego certyfikatu na ścieżce certyfikacji **powinien** być kontrolowany za pomocą protokołu weryfikacji statusu certyfikatów w trybie on-line (*ang. Online Certificate Status Protocol – OCSP*) lub listy unieważnień certyfikatów (*ang. Certificate Revocation List – CRL*). Kontrola OCSP **powinna** być przeprowadzana zgodnie z dokumentem [63]. Informacje o unieważnieniu **powinny** być uzyskiwane w sposób opisany w podrozdziale 4.2.2.

Nie wszyscy klienci obsługują sprawdzanie ograniczenia nazwy. Instytucje państwowe **powinny** korzystać tylko z klientów, które wykonują kontrolę ograniczenia nazwy, aby mieć pewność, że nieautoryzowane certyfikaty są prawidłowo odrzucane.

Klient **powinien** zakończyć połączenie TLS, jeżeli weryfikacja ścieżki nie powiedzie się.

Instytucje państwowe **powinny** używać tylko klientów, które sprawdzają, czy nazwa DNS lub adres IP (w zależności od tego, co jest zawarte w żądaniu TLS klienta) pasuje do nazwy DNS lub adresu IP zawartego w certyfikacie serwera. Klient **powinien** zakończyć połączenie TLS, jeżeli weryfikacja nazwy nie powiedzie się.

4.5.2 MAGAZYN KOTWIC ZAUFANIA

Posiadanie zbyt dużej liczby kotwic zaufania zainstalowanych w kliencie TLS może zwiększyć prawdopodobieństwo, że klient ten padnie ofiarą podszywania się. Wraz ze wzrostem liczby kotwic zaufania wzrasta liczba CA, którym klient ufa, a także wzrasta szansa, że dojdzie do naruszenia bezpieczeństwa jednego z tych CA, jego systemu rejestracji lub procesu w celu nieuprawnionego wydania certyfikatów serwera TLS.

Klienci **nie powinni** przepełniać swoich magazynów zaufania różnymi certyfikatami CA, które mogą być weryfikowane na zasadzie certyfikacji krzyżowej[37]. Bezpośrednie zaufanie do tych certyfikatów może nadmiernie narażać klientów na różne sytuacje, w tym m.in. na unieważnienie lub naruszenie bezpieczeństwa tych kotwic zaufania. Bezpośrednie zaufanie zwiększa również obciążenie klientów pod względem operacyjnym i bezpieczeństwa, związane z promulgacją dodawania i usuwania kotwic zaufania. Zamiast tego klient **powinien** polegać na tym, że serwer przepełni lub nie dostarczy listy wskazówek, aby złagodzić problem wyboru certyfikatu klienta i budowania ścieżki, jak to zostało omówione w podrozdziale 3.5.4.

4.5.3 SPRAWDZANIE ROZMIARU KLUCZA SERWERA

Jedynym bezpośrednim mechanizmem umożliwiającym klientowi sprawdzenie czy rozmiar klucza przedstawiony w certyfikacie publicznym serwera jest akceptowalny, jest zbadanie klucza publicznego serwera w certyfikacie. Pośrednim mechanizmem jest sprawdzenie czy certyfikat klucza publicznego serwera został wydany w ramach zasad, które gwarantują minimalną siłę kryptograficzną użytych algorytmów podpisu i skrótu. W niektórych przypadkach można to zrobić, jeśli klient wykona przetwarzanie i sprawdzanie zasad certyfikatu. Ponieważ jednak wielu klientów TLS nie można skonfigurować do akceptowania lub odrzucania certyfikatów na podstawie zasad, zgodnie z którymi zostały one wydane, konieczne może być zagwarantowanie, że magazyn kotwic zaufania zawiera tylko kotwice zaufania dla CA, które wydają certyfikaty w ramach akceptowalnych zasad. Klient **powinien** sprawdzić długość klucza publicznego serwera, jeśli implementacja klienta jest wyposażona w mechanizm służący do tego celu. Klient **powinien** sprawdzić długość klucza publicznego serwera, jeśli serwer używa kluczy efemerycznych do tworzenia głównego klucza tajnego, a implementacja klienta jest wyposażona w mechanizm służący do tego celu.

[37] Certyfikaty CA, które mogą być weryfikowane poprzez certyfikację krzyżową, mogą być dodawane do magazynu klienta jako certyfikaty niezaufane lub pośredniczące. Klienci używają certyfikatów przechowywanych jako niezaufane lub pośredniczące, aby uprościć budowanie ścieżki, ale nie traktują ich jako kotwic zaufania.

Długość każdego klucza zapisu jest określona przez wynegocjowany zestaw szyfrowania. Ograniczenia długości wspólnych kluczy sesji mogą być wymuszone poprzez skonfigurowanie klienta do obsługi wyłącznie zestawów szyfrowania, które spełniają wymagania dotyczące długości kluczy.

4.5.4 INTERFEJS UŻYTKOWNIKA

Gdy klient TLS jest przeglądarką, interfejs przeglądarki może być użyty do określenia, czy sesja TLS jest w toku. Oznaki tego, że trwa sesja TLS różnią się w zależności od przeglądarki. Przykłady takich oznak to ikona kłódki w pasku adresu URL, słowo „bezpieczny" poprzedzające adres URL lub inny kolor paska adresu URL. Niektórzy klienci, np. przeglądarki, mogą umożliwić dalsze sprawdzanie certyfikatu serwera i wynegocjowanych parametrów sesji poprzez kliknięcie ikony kłódki (lub innego wskaźnika). Użytkownicy **powinni** sprawdzać interfejs pod kątem obecności takiego wskaźnika, aby się upewnić, że sesja TLS jest w toku, a także **powinni** wizualnie badać adresy URL stron internetowych, aby się upewnić, że użytkownik zamierzał odwiedzić podaną stronę internetową. Użytkownicy **powinni** być świadomi, że adresy URL mogą wydawać się prawidłowe, ale nie być prawdziwe. Na przykład cyfra „1" i litera „l" wydają się dość podobne lub takie same dla ludzkiego oka.

Klucze uwierzytelniające klienta mogą znajdować się poza klientem (np. na kartach PIV). Użytkownicy **powinni** przestrzegać odpowiednich polityk i procedur dotyczących ochrony kluczy uwierzytelniających klienta znajdujących się poza klientem.

4.6 WZNAWIANIE SESJI I WCZESNE DANE

Uwagi dotyczące wznawiania sesji i zalecenia dotyczące serwerów zostały przedstawione w podrozdziale 3.6. Nie ma żadnych zaleceń odnoszących się konkretnie do klientów dotyczących wznawiania sesji przy użyciu protokołu TLS 1.2, 1.1 lub 1.0. Klienci zazwyczaj nie będą wiedzieć, czy zostały wdrożone jakiekolwiek mechanizmy zapobiegające atakom powtórzeniowym na dane 0-RTT w protokole TLS 1.3. Dlatego klienci stosujący protokół TLS 1.3 **nie powinni** wysyłać danych 0- RTT.

W dokumencie RFC 7918 [39] opisano technikę zwaną falstartem (*ang. False Start*), która umożliwia klientowi TLS 1.2 wysyłanie wczesnych danych. Chociaż koncepcja ta

jest podobna do danych 0-RTT w protokole TLS 1.3, istnieją różnice, które wpływają na bezpieczeństwo. Na przykład, atakujący może wykonać ataki typu downgrade, zarówno dotyczące wersji protokołu, jak i zestawów szyfrowania, i uzyskać dane klienta zanim protokół uzgodnienia zostanie uznany za nieważny. W dokumencie RFC 7918 znajdują się zalecenia umożliwiające poprawę bezpieczeństwa, jednak najbezpieczniej jest wyłączyć funkcję falstartu, chyba że istnieje rzeczywista potrzeba jej zastosowania. Klienci TLS 1.2 **nie powinni** używać techniki falstartu.

4.7 METODY KOMPRESJI

W przypadku klienta **należy** stosować te same zalecenia dotyczące kompresji, co w przypadku serwera, opisane w podrozdziale 3.7.

4.8 ASPEKTY OPERACYJNE

Klient i powiązana z nim platforma **powinni** być na bieżąco aktualizowani pod względem poprawek bezpieczeństwa. Ma to kluczowe znaczenie dla różnych aspektów bezpieczeństwa.

Po odebraniu na kliencie danych chronionych przez protokół TLS i ich odszyfrowaniu oraz uwierzytelnieniu przez warstwę TLS systemu klienckiego, dane niezaszyfrowane są dostępne dla aplikacji na platformie klienckiej.

Niniejsze rekomendacje nie ograniczają zagrożeń związanych z niewłaściwym wykorzystaniem lub ujawnieniem danych uwierzytelniających klienta, które znajdują się na jego komputerze. Takie dane uwierzytelniające mogą obejmować klucz prywatny używany do uwierzytelniania klienta lub inne dane uwierzytelniające (np. hasło jednorazowe (*ang. One-Time Password – OTP*) lub identyfikator użytkownika i hasło) służące do uwierzytelniania aplikacji po stronie serwera.

Z tych powodów stosowanie protokołu TLS nie zwalnia z konieczności stosowania odpowiednich środków bezpieczeństwa dotyczących klienta, opisanych w obowiązujących standardach przetwarzania informacji i publikacjach stosownych organów, w celu ochrony systemów i aplikacji komputerowych. Użytkownicy **powinni** obsługiwać systemy klienta zgodnie z instrukcjami instytucji rządowych i administratora.

ZAŁĄCZNIK A–AKRONIMY

Wybrane akronimy i skróty użyte w niniejszej publikacji zostały rozwinięte poniżej.

Dodatkowo patrz: NSC 7298, *Słownik kluczowych pojęć z zakresu cyberbezpieczeństwa*

Akronim	Terminologia angielska	Terminologia polska
3DES	Mnemonik IETF for Triple Data Encryption Algorithm	Algorytm trzykrotnego szyfrowania danych
AEAD	Authenticated Encryption with Associated Data	Uwierzytelnione szyfrowanie z powiązanymi danymi
AES	Advanced Encryption Standard	Zaawansowany standard szyfrowania
CA	Certification Authority	Urząd certyfikacji
CBC	Cipher Block Chaining	Wiązanie bloków zaszyfrowanych
CCM	Counter with CBC-MAC	Counter with CBC-MAC
CRL	Certificate Revocation List	Lista unieważnionych certyfikatów
DES	Data Encryption Standard	Standard szyfrowania danych
DH	Algorytm wymiany kluczy Diffiego-Hellmana	Algorytm wymiany kluczy Diffiego-Hellmana
DHE	Algorytm wymiany kluczy efemerycznych Diffiego-Hellmana	Algorytm wymiany kluczy efemerycznych Diffiego-Hellmana
DNS	Domain Name System	System nazw domen
DNSSEC	DNS Security Extensions	Rozszerzenia bezpieczeństwa DNS
DSA	Digital Signature Algorithm	Algorytm podpisu cyfrowego
DSS	Digital Signature Standard	Standard podpisu cyfrowego (implikuje DSA)
EC	Elliptic Curve	Krzywa eliptyczna
ECDHE	Ephemeral Elliptic Curve Diffie-Hellman	Efemeryczne krzywe eliptyczne Diffiego-Hellmana
ECDSA	Elliptic Curve Digital Signature Algorithm	Algorytm podpisu cyfrowego krzywej eliptycznej

FIPS	Federal Information Processing Standard	Federalny standard przetwarzania informacji
GCM	Galois Counter Mode	Tryb licznika Galois
HKDF	HMAC-based Extract-and-Expand Key Derivation Function	Oparta na HMAC funkcja wyprowadzania klucza typu extract-and-expand
HMAC	Keyed-hash Message Authentication Code	Kod uwierzytelniania komunikatów ze skróconym kluczem
IETF	Internet Engineering Task Force	Grupa Robocza ds. Inżynierii Internetowej
KDF	Key Derivation Function	Funkcja wyprowadzania klucza
MAC	Message Authentication Code	Kod uwierzytelniania komunikatu
OCSP	Online Certificate Status Protocol	Protokół weryfikacji statusu certyfikatów w trybie on-line
OID	Object Identifier	Identyfikator obiektu
PIV	Personal Identity Verification	Certyfikat identyfikacji tożsamości
PKCS	Public-Key Cryptography Standards	Standardy kryptografii klucza publicznego
PKI	Public Key Infrastructure	Infrastruktura klucza publicznego
PRF	Pseudo-random Function	Funkcja pseudolosowa
PSK	Pre-Shared Key	Klucz wstępny
RFC	Request for Comments	Memorandum zatytułowane „Prośba o komentarze”
SHA	Secure Hash Algorithm	Bezpieczna funkcja skrótu
SSL	Secure Socket Layer	Protokół SSL
TDEA	Triple Data Encryption Algorithm	Algorytm trzykrotnego szyfrowania danych
TLS	Transport Layer Security	Bezpieczeństwo warstwy transportowej
URL	Uniform Resource Locator	Adres URL

ZAŁĄCZNIK B–INTERPRETACJA NAZW ZESTAWÓW SZYFROWANIA

Nazwy zestawów szyfrowania TLS składają się z zestawu mnemoników oddzielonych od siebie znakami podkreślenia (tj. „_”). Konwencja nazewnictwa w protokole TLS 1.3 różni się od wspólnej konwencji stosowanej w TLS 1.0, 1.1 i 1.2. Podrozdział B.1 zawiera wskazówki dotyczące interpretacji nazw zestawów szyfrowania zalecanych w niniejszych rekomendacjach dla protokołu TLS w wersjach 1.0, 1.1 i 1.2. Podrozdział B.2 zawiera wskazówki dotyczące interpretacji nazw zestawów szyfrowania dla protokołu TLS 1.3. We wszystkich zestawach szyfrowania TLS pierwszym mnemonikiem jest nazwa protokołu (czyli „TLS”).

B.1 INTERPRETACJA NAZW ZESTAWÓW SZYFROWANIA DLA PROTOKOŁU TLS 1.0, 1.1 ORAZ 1,2

Jak przedstawiono w podrozdziale 3.3.1, nazwy tych zestawów szyfrowania mają następującą postać:

TLS_*KeyExchangeAlg*_WITH_*EncryptionAlg_MessageAuthenticationAlg*
KeyExchangeAlg składa się z jednego lub dwóch mnemoników.

- Jeśli jest tylko jeden mnemonik, musi to być PSK w oparciu o zalecenia zawarte w niniejszych rekomendacjach. Pojedynczy mnemonik PSK oznacza, że wstępny klucz tajny jest ustalany przy użyciu wyłącznie algorytmów symetrycznych z kluczami wstępnymi, jak opisano w dokumencie RFC 4279 [30]. Zestawy szyfrowania z kluczem wstępnym, które są zatwierdzone do użytku z protokołem TLS 1.2, zostały wymienione w załączniku C.
- Jeśli po nazwie protokołu występują dwa mnemoniki, pierwszym mnemonikiem wymiany klucza powinien być DH, ECDH, DHE lub ECDHE.
  - ✓ Gdy pierwszym mnemonikiem wymiany klucza jest DH lub ECDH, oznacza to, że klucz publiczny serwera w jego certyfikacie jest używany w wymianie kluczy DH lub ECDH, a drugi mnemonik informuje o algorytmie podpisu, który został użyty przez wydający CA do podpisania certyfikatu serwera.
  - ✓ Gdy pierwszym mnemonikiem wymiany kluczy jest DHE lub ECDHE, oznacza to, że do wymiany kluczy zostanie użyty efemeryczny algorytm DH

lub ECDH, a drugi mnemonik wskazuje typ klucza publicznego podpisu serwera, który zostanie użyty do uwierzytelnienia efemerycznego klucza publicznego serwera[38].

Część *EncryptionAlg* informuje o symetrycznym algorytmie szyfrowania i związanym z nim trybie działania.

MessageAuthenticationAlg to ogólnie algorytm skrótu, który ma być użyty dla kodu HMAC, jeśli dotyczy[39]. W przypadkach, gdy kod HMAC nie ma zastosowania (np. AES-GCM) lub zestaw szyfrowania został zdefiniowany po wydaniu dokumentu RFC na temat TLS 1.2, mnemonik ten informuje o algorytmie skrótu stosowanym w funkcji PRF.

Poniższe przykłady ilustrują sposób interpretacji nazw zestawów szyfrowania:

- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256: Do wymiany klucza używany jest efemeryczny algorytm DH. Efemeryczny klucz publiczny serwera jest uwierzytelniany za pomocą klucza publicznego RSA serwera. Po zakończeniu wykonywania protokołu uzgodnienia komunikaty są szyfrowane przy użyciu standardu AES-256 w trybie CBC. Algorytm SHA-256 jest używany zarówno do obliczenia PRF, jak i HMAC.
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384: Do wymiany klucza używany jest efemeryczny algorytm ECDH. Efemeryczny klucz publiczny serwera jest uwierzytelniany za pomocą klucza publicznego ECDSA serwera. Po zakończeniu wykonywania protokołu uzgodnienia, komunikaty są szyfrowane i uwierzytelniane przy użyciu standardu AES-256 w trybie GCM, a dla PRF używany jest algorytm SHA-384. Ponieważ stosowany jest tryb szyfrowania uwierzytelnionego, komunikaty nie mają ani nie wymagają kodu uwierzytelniania wiadomości HMAC.

[38] W tym przypadku algorytm podpisu używany przez CA do podpisania certyfikatu nie jest oznaczony w nazwie zestawu szyfrowania.

[39] Kod HMAC nie jest stosowany, gdy trybem działania szyfrowania symetrycznego jest szyfrowanie uwierzytelnione. Należy zauważyć, że dla zestawu szyfrowania w trybie CCM nie jest określany ostatni mnemonik i wymagane jest użycie algorytmu SHA-256 dla PRF.

B.2 INTERPRETACJA NAZW ZESTAWÓW SZYFROWANIA DLA PROTOKOŁU TLS 1.3

Jak przedstawiono w podrozdziale 3.3.1, nazwy tych zestawów szyfrowania mają następującą postać:

TLS_*AEAD* _*HASH*

Skrót AEAD informuje o algorytmie AEAD, który jest używany do zapewnienia poufności, integralności i uwierzytelniania komunikatów. Zatwierdzone przez NIST algorytmy TLS 1.3 AEAD obejmują zalecany przez NIST szyfr blokowy i zalecany przez NIST tryb AEAD.

HASH informuje o algorytmie skrótu, który jest używany z funkcją HKDF podczas wyprowadzania klucza.

Poniższe przykłady ilustrują sposób interpretacji nazw zestawów szyfrowania TLS 1.3:

- TLS_AES_256_GCM_SHA384: Komunikaty są szyfrowane i uwierzytelniane przy użyciu standardu AES- 256 w trybie GCM, a algorytm SHA-384 jest używany w ramach funkcji HKDF.
- TLS_AES_128_CCM_SHA256: Komunikaty są szyfrowane i uwierzytelniane przy użyciu standardu AES- 128 w trybie CCM, a algorytm SHA-256 jest używany w ramach funkcji HKDF.

Negocjacja metody wymiany kluczy jest prowadzona w innym miejscu protokołu uzgodnienia TLS.

ZAŁĄCZNIK C–KLUCZE WSTĘPNE

Klucze wstępne (*ang. Pre-shared keys – PSK*) to klucze symetryczne, które istnieją już przed rozpoczęciem sesji TLS (np. w wyniku ręcznej dystrybucji). Zastosowanie kluczy PSK w wersjach protokołu TLS wcześniejszych niż TLS 1.3 zostało opisane w publikacji RFC 4279 [30], RFC 5487 [3] oraz RFC 5489 [4]. W protokole TLS 1.3 klucze wstępne są wykorzystywane do wznawiania sesji. Zasadniczo klucze wstępne **nie powinny** być używane w wersjach TLS wcześniejszych niż TLS 1.3 lub do pierwotnego ustanowienia sesji w TLS 1.3. Stosowanie kluczy wstępnych może być jednak wskazane w niektórych zamkniętych środowiskach, które posiadają odpowiednie wsparcie w zakresie zarządzania kluczami. Na przykład, może być odpowiednie dla środowisk o ograniczonym przetwarzaniu, pamięci lub mocy. Jeśli klucze PSK są odpowiednie i obsługiwane, **należy** przestrzegać dodatkowych zaleceń podanych poniżej.

Zalecane zestawy szyfrowania z kluczem wstępnym (PSK) dla protokołu TLS 1.2 zostały wymienione poniżej. Zestawy szyfrowania dla protokołu TLS 1.3 (patrz podrozdział 3.3.1.2) mogą być używane z kluczami wstępnymi. Klucze wstępne **powinny** być dystrybuowane w bezpieczny sposób, np. poprzez bezpieczną dystrybucję ręczną lub z wykorzystaniem certyfikatu ustanowienia klucza. W tych zestawach szyfrowania klucz wstępny jest wykorzystywany do uwierzytelniania podmiotu (zarówno serwera, jak i klienta), a do ustanawiania klucza mogą być również wykorzystywane efemeryczne algorytmy Diffiego-Hellmana (DHE) lub efemeryczne algorytmy krzywej eliptycznej Diffiego-Hellmana (ECDHE). Na przykład, gdy używane jest algorytm DHE, wynik obliczeń w ramach protokołu Diffiego-Hellmana jest łączony z kluczem wstępnym i innymi danymi wejściowymi w celu określenia wstępnego klucza tajnego.

Klucz wstępny **powinien** zapewniać poziom bezpieczeństwa wynoszący co najmniej 112 bitów. Ponieważ te zestawy szyfrowania wymagają kluczy wstępnych, nie są one zasadniczo stosowane w powszechnych bezpiecznych aplikacjach internetowych i nie oczekuje się, że będą powszechnie obsługiwane w klientach lub serwerach TLS. NIST sugeruje, aby zestawy szyfrowania z kluczem wstępnym były brane pod uwagę w zastosowaniach w infrastrukturze, szczególnie jeśli wymagane jest częste

uwierzytelnianie podmiotów sieciowych.

Zestawy szyfrowania z kluczem wstępnym mogą być używane tylko w sieciach, w których zarówno klient, jak i serwer należą do tej samej organizacji. Zestawy szyfrowania używające kluczy wstępnych **nie powinny** być stosowane z protokołem TLS 1.0 lub 1.1 i **nie powinny** być stosowane z TLS 1.2, w przypadku, gdy klient lub serwer administracji państwowej komunikuje się z systemami spoza administracji państwowej.

Serwery i klienci TLS 1.2 używające kluczy wstępnych mogą obsługiwać następujące zestawy szyfrowania:

- TLS_DHE_PSK_WITH_AES_128_GCM_SHA256 (0X00, 0XAA)
- TLS_DHE_PSK_WITH_AES_256_GCM_SHA384 (0X00, 0XAB)
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA256 (0XC0, 0X37)
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA384 (0XC0, 0X38)
- TLS_DHE_PSK_WITH_AES_128_CCM (0XC0, 0XA6)
- TLS_DHE_PSK_WITH_AES_256_CCM (0XC0, 0XA7)
- TLS_PSK_DHE_WITH_AES_128_CCM_8 (0XC0, 0XAA)
- TLS_PSK_DHE_WITH_AES_256_CCM_8 (0XC0, 0XAB)
- TLS_DHE_PSK_WITH_AES_128_CBC_SHA256 (0x00, 0xB2)
- TLS_DHE_PSK_WITH_AES_256_CBC_SHA384 (0x00, 0xB3)
- TLS_PSK_WITH_AES_128_GCM_SHA256 (0x00, 0xA8)
- TLS_PSK_WITH_AES_256_GCM_SHA384 (0x00, 0xA9)
- TLS_PSK_WITH_AES_128_CCM (0xC0, 0xA4)
- TLS_PSK_WITH_AES_256_CCM (0xC0, 0xAs)
- TLS_PSK_WITH_AES_128_CCM_8 (0xC0, 0xA8)
- TLS_PSK_WITH_AES_256_CCM_8 (0xC0, 0xA9)
- TLS_PSK_WITH_AES_128_CBC_SHA256 (0x00, 0xAE)

- TLS_PSK_WITH_AES_256_CBC_SHA384 (0x00, 0xAF)
- TLS_ECDHE_PSK_WITH_AES_128_CBC_SHA (0xC0, 0x35)
- TLS_ECDHE_PSK_WITH_AES_256_CBC_SHA (0xC0, 0x36)
- TLS_DHE_PSK_WITH_AES_128_CBC_SHA (0x00, 0x90)
- TLS_DHE_PSK_WITH_AES_256_CBC_SHA (0x00, 0x91)
- TLS_PSK_WITH_AES_128_CBC_SHA (0x00, 0x8C)
- TLS_PSK_WITH_AES_256_CBC_SHA (0x00, 0x8D)

ZAŁĄCZNIK D–TRANSPORT KLUCZA RSA

Transport klucza RSA to mechanizm wymiany kluczy, w którym wstępny klucz tajny jest wybierany przez klienta, szyfrowany kluczem publicznym serwera i wysyłany do serwera. Jest on dostępny w wersjach TLS od 1.0 do 1.2, ale nie jest obsługiwany przez TLS 1.3. Chociaż jest to wygodna metoda wymiany kluczy, gdy certyfikat serwera zawiera klucz publiczny RSA, jednak metoda ta ma kilka wad:

1. Klient ponosi wyłączną odpowiedzialność za generowanie wstępnego klucza tajnego. Jeśli klient nie ma wystarczającej entropii, aby wygenerować wstępny klucz tajny, poziom bezpieczeństwa sesji ulegnie pogorszeniu.
2. Nie zapewnia doskonałego utajnienia przekazywania.
3. Schemat dopełniania, który jest wykorzystywany w ramach TLS do tej operacji, ma znaną lukę, która powoduje, że w implementacjach TLS należy stosować środki zabezpieczające przed atakami.

Z tych powodów w niniejszych wytycznych nie zaleca się stosowania zestawów szyfrowania, które do wymiany kluczy wykorzystują transport klucza RSA (patrz podrozdział 3.3.1).

Doskonałe utajnienie przekazywania (patrz przypis 20) jest często celem bezpieczeństwa, ponieważ zapobiega sytuacji, w której naruszenie bezpieczeństwa kluczy długoterminowych umożliwia odszyfrowanie sesji. Jedynym sposobem na osiągnięcie tego stanu w ramach wykonywania protokołu TLS jest użycie mechanizmu wymiany kluczy, który opiera się na efemerycznych parametrach (tj. zestawów szyfrowania zawierających algorytm DHE lub ECDHE), jak określono w dokumencie RFC 5246 [25].

Transport kluczy RSA z wykorzystaniem standardu PKCS #1 v1.5 jest podatny na ataki typu Bleichenbacher oracle. Dokument RFC 5246 zawiera opis działań mających na celu zapobieganie atakom polegającym na przetwarzaniu nieprawidłowo sformatowanych komunikatów w sposób nieodróżnialny od przetwarzania komunikatów prawidłowo sformatowanych (patrz [25], podpunkt 7.4.7.1). Techniki zapobiegania nie zawsze są skuteczne w praktyce (patrz przykłady w [14]).

D.1 OKRES PRZEJŚCIOWY

Chociaż w niniejszych rekomendacjach nie zaleca się stosowania zestawów szyfrowania wykorzystujących transport klucza RSA, jednak w praktyce mogą wystąpić okoliczności, w których będzie on wymagany. Na przykład, jeśli instytucja rządowa używa urządzenia sieciowego do celów regulacyjnych lub bezpieczeństwa na poziomie korporacyjnym, które działa wyłącznie w oparciu o takie zestawy szyfrowania, to te zestawy szyfrowania mogą wymagać włączenia. Zaleca się, aby instytucje państwowe przeszły na nową metodę zaspokajającą ich potrzeby tak szybko, jak to możliwe.

Jeśli transport klucza RSA jest konieczny podczas opracowywania nowej strategii kontroli ruchu, można stosować wyłącznie zestawy szyfrowania z transportem klucza RSA z poniższej listy. Ogólne informacje na temat kolejności preferowanych rozwiązań znajdują się w podrozdziale 3.3.1.1.

- TLS_RSA_WITH_AES_128_CCM (xC0, x9C)
- TLS_RSA_WITH_AES_256_CCM (xC0, x9D)
- TLS_RSA_WITH_AES_128_CCM_8 (xC0, xA0)
- TLS_RSA_WITH_AES_256_CCM_8 (xC0, xA1)
- TLS_RSA_WITH_AES_128_CBC_SHA (x00, x2F)
- TLS_RSA_WITH_AES_256_CBC_SHA (x00, x35)
- TLS_RSA_WITH_AES_128_CBC_SHA256 (x00, 3C)
- TLS_RSA_WITH_AES_256_CBC_SHA256 (x00, 3D)
- TLS_RSA_WITH_AES_128_GCM_SHA256 (x00, x9C)
- TLS_RSA_WITH_AES_256_GCM_SHA384 (x00, x9D)

Informacje na temat harmonogramu wycofywania z użytku znajdują się w wytycznych dotyczących okresu przejściowego w publikacji NIST SP 800-131A [10].

ZAŁĄCZNIK E– PRZYSZŁE MOŻLIWOŚCI[40]

W tej części publikacji określono nowe koncepcje i możliwości, które można zastosować w ramach TLS. W miarę rozwoju tych koncepcji i dostępności komercyjnych produktów obsługujących je, niniejsze rekomendacje zostaną poddane przeglądowi w celu przedstawienia konkretnych zaleceń.

E.1 INFRASTRUKTURA PKI ZAUFANIA PUBLICZNEGO (*ANG. PUBLIC TRUST PKI*)

Podkomitet ds. tożsamości, uwierzytelniania i zarządzania dostępem (*ang. Identity, Credential, and Access Management – ICAM*) komitetu ds. bezpieczeństwa informacji i zarządzania tożsamością rady głównego urzędu ds. Informatyki (*ang. Federal CIO Council's Information Security and Identity Management Committee*) opracowuje nową główną infrastrukturę zaufania publicznego i wystawiający urząd CA w celu wydawania certyfikatów serwera TLS dla rządowych usług internetowych w publicznym Internecie. Intencją jest, aby ten nowy główny ośrodek CA został włączony do wszystkich powszechnie używanych magazynów zaufania, tak aby instytucje państwowe mogły uzyskać swoje certyfikaty serwera TLS z tej infrastruktury PKI, a nie z komercyjnych CA. Zasady certyfikatu dla tej infrastruktury PKI są opracowywane pod adresem https://devicepki.idmanagement.gov.

Gdy ta infrastruktura PKI będzie działać i zostanie włączona do powszechnie używanych magazynów zaufania, instytucje państwowe powinny rozważyć pozyskiwanie swoich certyfikatów serwera TLS z tej infrastruktury PKI.

E.2 UWIERZYTELNIANIE NAZWANYCH PODMIOTÓW OPARTE NA DNS (*ANG. DNS-BASED AUTHENTICATION OF NAMED ENTITIES – DANE*)

W protokole DANE wykorzystano rozszerzenia bezpieczeństwa DNS (DNSSEC), aby zapewnić mechanizmy bezpiecznego pozyskiwania informacji o certyfikatach serwerów TLS z systemu DNS. W dokumencie RFC 6698 [34] określono rekord

[40] Dla zainteresowanych.

zasobów, który może zostać udostępniony w systemie DNS, zawierający certyfikat (lub klucz publiczny certyfikatu) wraz ze wskazaniem sposobu wykorzystania certyfikatu. Są cztery opcje:

1. Rekord DNS zawiera certyfikat jednostki końcowej. Oprócz walidacji certyfikatu klucza publicznego serwera, w sposób określony w podrozdziale 4.5, klient sprawdza, czy certyfikat serwera TLS odpowiada certyfikatowi umieszczonemu w rekordach DNS.
2. Rekord DNS zawiera certyfikat jednostki końcowej wydany przez domenę[41]. Klient może użyć certyfikatu, jeżeli sprawdzi, czy certyfikat serwera TLS odpowiada certyfikatowi zawartemu w rekordach DNS (tzn. klient rezygnuje ze sposobu walidacji certyfikatu klucza publicznego serwera, o którym mowa w podrozdziale 4.5).
3. Rekord DNS zawiera certyfikat CA. Oprócz walidacji certyfikatu klucza publicznego serwera, w sposób określony w podrozdziale 4.5, klient sprawdza, czy ścieżka certyfikacji serwera TLS obejmuje certyfikat CA znajdujący się w rekordach DNS.
4. Rekord DNS zawiera certyfikat, który ma być użyty jako kotwica zaufania. Klient dokonuje walidacji certyfikatu serwera TLS, w sposób określony w podrozdziale 4.5, używając kotwicy zaufania zawartej w rekordach DNS zamiast kotwic zaufania z lokalnego magazynu klienta.

W każdym przypadku klient weryfikuje podpisy cyfrowe na rekordach DNS przy użyciu rozszerzenia DNSSEC opisanego w dokumencie RFC 4033 [2].

[41] W tym kontekście certyfikat „wydawany przez domenę" to taki, który jest wydawany przez administratora nazwy domeny bez udziału zewnętrznego CA. Odpowiada to scenariuszowi użycia nr 3 w podpunkcie 2.1.1 dokumentu RFC 6698.

E.3 ZASZYFROWANA IDENTYFIKACJA NAZWY SERWERA

W TLS 1.3 w ramach protokołu uzgodnienia szyfrowanych jest więcej informacji niż we wcześniejszych wersjach TLS, jednak klient nadal wysyła do serwera rozszerzenie identyfikacji nazwy serwera (*ang. Server Name Indication – SNI*) w postaci zwykłego tekstu (patrz podrozdział 3.4.1.2). Oznacza to, że podsłuchujący może ustalić nazwę domeny serwera, z którym łączy się klient, nawet jeśli nie był w stanie podsłuchać wyszukiwania DNS dokonywanego przez klienta i nawet jeśli serwer hostuje wiele nazw domen pod tym samym adresem IP.

Grupa robocza TLS pracuje nad opracowaniem mechanizmu szyfrowania rozszerzenia SNI, który umożliwiłby klientowi wysłanie SNI do serwera w postaci zaszyfrowanej, dzięki czemu informacja ta nie byłaby dostępna dla podsłuchujących [35, 58]. Włączenie obsługi tego mechanizmu w klientach i serwerach TLS zapewniłoby dodatkowy poziom prywatności klientów TLS.

ZAŁĄCZNIK F–OKREŚLANIE POTRZEBY KORZYSTANIA Z PROTOKOŁU TLS W WERSJI 1.0 I 1.1

Włączenie możliwości korzystania z protokołów TLS 1.0 lub 1.1, gdy nie są one wymagane, może sprawić, że systemy i użytkownicy będą podatni na ataki (takie jak BEAST i Klima [65]). Jednak wyłączenie starszych wersji TLS, gdy są one potrzebne, może uniemożliwić dostęp użytkownikom, którzy nie będą w stanie zainstalować lub aktualizować klienta do wersji obsługującej TLS 1.3 lub 1.2.

Administrator systemu musi rozważyć korzyści i zagrożenia związane ze stosowaniem protokołu TLS 1.0 lub 1.1 w kontekście aplikacji obsługiwanych przez serwer i zdecydować, czy korzyści wynikające z zastosowania protokołu TLS 1.0 lub 1.1 przeważają nad ryzykiem. Decyzja ta powinna być podyktowana usługami działającymi na serwerze oraz wersjami obsługiwanymi przez klientów uzyskujących dostęp do serwera. Usługi, które nie mają dostępu do wartościowych informacji (takich jak dane osobowe lub dane finansowe) mogą skorzystać na zastosowaniu TLS 1.0 poprzez zwiększenie dostępności przy niewielkim wzroście ryzyka. Z drugiej strony, usługi, które mają dostęp do danych o dużej wartości, mogą zwiększyć prawdopodobieństwo naruszenia bezpieczeństwa w zamian za stosunkowo niewielki zysk w zakresie dostępności. Decyzja o obsłudze TLS 1.0 lub 1.1 musi być podejmowana indywidualnie dla każdego przypadku w oparciu o ocenę techniczną. Ma to na celu upewnienie się, że obsługa starszych wersji TLS jest absolutnie konieczna, a związane z tym ryzyko i konsekwencje biznesowe są zrozumiałe i akceptowane.

W niniejszych wytycznych nie podano konkretnych zaleceń dotyczących działań, które można podjąć w celu dokonania takiego ustalenia. Dostępne są narzędzia (takie jak Data Analytics Program [69]), dostarczające administratorom systemów informacji, które mogą być wykorzystane do oceny skutków obsługi lub braku obsługi wersji protokołu TLS wcześniejszych niż TLS 1.2. Na przykład dane DAP dotyczące systemów operacyjnych odwiedzających i wersji przeglądarek mogą pomóc administratorom określić, jaki procent osób odwiedzających witryny instytucji państwowej nie może domyślnie negocjować w zalecanych wersjach protokołu TLS.

W wielu produktach, w których wdrożono wersję TLS 1.1, wdraża się również TLS 1.2. Z tego względu obsługa protokołu TLS 1.1 przez serwery może się okazać zbędna. Administratorzy mogą określić, czy protokół TLS 1.1 jest wymagany, oceniając, czy musi on obsługiwać połączenia z klientami, w których TLS 1.1 jest najnowszą dostępną wersją.

ZAŁĄCZNIK G–REFERENCJE

NARODOWE STANDARDY CYBERBEZPIECZEŃSTWA[42]	
NSC 199	Standardy kategoryzacji bezpieczeństwa – na podstawie FIPS 199
NSC 200	Minimalne wymagania bezpieczeństwa informacji i systemów informacyjnych podmiotów publicznych – na podstawie FIPS 200
NSC 800-30	Przewodnik dotyczący postępowania w zakresie szacowania ryzyka w podmiotach realizujących zadania publiczne – na podstawie NIST SP 800-30
NSC 800-34	Poradnik planowania awaryjnego – na podstawie NIST SP 800-34
NSC 800-37	Ramy zarządzania ryzykiem w organizacjach i systemach informacyjnych. Bezpieczeństwo i ochrona prywatności w cyklu życia systemu – na podstawie NIST SP 800-37
NSC 800-39	Zarządzanie ryzykiem bezpieczeństwa informacji. Przegląd struktury organizacyjnej, misji i systemu informacyjnego – na podstawie NIST SP 800-39
NSC 800-53	Zabezpieczenia i ochrona prywatności systemów informacyjnych oraz organizacji – na podstawie NIST SP 800-53
NSC 800-53A	Ocenianie środków bezpieczeństwa i ochrony prywatności systemów informacyjnych oraz organizacji. Tworzenie skutecznych planów oceny – na podstawie NIST SP 800-53A

[42] Narodowe Standardy Cyberbezpieczeństwa - Baza wiedzy - Portal Gov.pl (www.gov.pl)

NARODOWE STANDARDY CYBERBEZPIECZEŃSTWA[42]	
NSC 800-53B	Zabezpieczenia bazowe systemów informacyjnych oraz organizacji – na podstawie NIST SP 800-53B
NSC 800-53 MAP	Mapowanie środków bezpieczeństwa: NSC 800-53 wer. 2 – PN-ISO/IEC 27001:2013; PN-ISO/IEC 27001:2013 – NSC 800-53 wer. 2 Patrz: SP 800-53 Rev. 5, Security and Privacy Controls for Info Systems and Organizations | CSRC (nist.gov)
NSC 800-60	Wytyczne w zakresie określania kategorii bezpieczeństwa informacji I kategorii bezpieczeństwa systemu informacyjnego – na podstawie NIST SP 800-60
NSC 800-61	Podręcznik postępowania z incydentami naruszenia bezpieczeństwa komputerowego – na podstawie NIST SP 800-61

PUBLIKACJE ANGLOJĘZYCZNE[43]

[1] AlFardan NJ, Paterson KG (2013) Lucky Thirteen: Breaking the TLS and DTLS Record Protocols. Available at http://www.isg.rhul.ac.uk/tls/TLStiming.pdf

[2] Arends R, Austein R, Larson M, Massey D, Rose S (2005) DNS Security Introduction and Requirements. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 4033. https://doi.org/10.17487/RFC4033

[3] Badra M (2009) Pre-Shared Key Cipher Suites for TLS with SHA-256/384 and AES Galois Counter Mode. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 5487. https://doi.org/10.17487/RFC5487

[4] Badra M, Hajjeh I (2009) ECDHE_PSK Cipher Suites for Transport Layer Security (TLS). (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 5489. https://doi.org/10.17487/RFC5489

[5] Barker EB (2016) Recommendation for Key Management Part 1: General. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-57 Part 1, Rev. 4. https://doi.org/10.6028/NIST.SP.800-57pt1r4

[6] Barker EB, Chen L, Roginsky A, Vassilev A, Davis R (2018) Recommendation for PairWise Key-Establishment Schemes Using Discrete Logarithm Cryptography. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-56A, Rev. 3. https://doi.org/10.6028/NIST.SP.800-56Ar3

[43] Publikacje angielski zostały podane w celach uzupełniających dla osób zainteresowanych.

PUBLIKACJE ANGLOJĘZYCZNE[43]

[7] Barker EB, Chen L, Roginsky A, Vassilev A, Davis R, Simon S (2019) Recommendation for Pair-Wise Key-Establishment Using Integer Factorization Cryptography. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-56B, Rev. 2. https://doi.org/10.6028/NIST.SP.800-56Br2

[8] Barker EB, Kelsey JM (2015) Recommendation for Random Number Generation Using Deterministic Random Bit Generators. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-90A, Rev. 1. *https://doi.org/10.6028/NIST.SP.800-90Ar1*

[9] Barker EB, Mouha N (2017) Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-67, Rev. 2. *https://doi.org/10.6028/NIST.SP.800-67r2*

[10] Barker EB, Roginsky A (2019) Transitioning the Use of Cryptographic Algorithms and Key Lengths. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-131A, Rev. 2. https://doi.org/10.6028/NIST.SP.800-131Ar2

[11] Be'ery T, Shulman A (2013) A Perfect CRIME? Only TIME Will Tell. Blackhat Europe 2013 (Amsterdam, The Netherlands). Available at https://media.blackhat.com/eu- 13/briefings/Beery/bh-eu-13-a-perfect-crime-beery-wp.pdf

[12] Bhargavan K, Delignat-Lavaud A, Fournet C, Pironti A, Strub PY (2014) Triple Handshakes and Cookie Cutters: Breaking and Fixing Authentication over TLS. 2014 IEEE Symposium on Security and Privacy (IEEE, San Jose, CA), pp 98-113. https://doi.org/10.1109/SP.2014.14

PUBLIKACJE ANGLOJĘZYCZNE[43]

[13] Bhargavan K, Delignat-Lavaud A, Pironti A, Langley A, Ray M (2015) Transport Layer Security (TLS) Session Hash and Extended Master Secret Extension. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 7627. https://doi.org/10.17487/RFC7627

[14] Böck H, Somorovsky J, Young C (2017) Return Of Bleichenbacher's Oracle Threat (ROBOT). Cryptology ePrint Archive, Report 2017/1189. https://eprint.iacr.org/2017/1189

[15] Bradner S (1997) Key words for use in RFCs to Indicate Requirement Levels. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 2119. https://doi.org/10.17487/RFC2119

[16] CA/Browser F orum (2019) Baseline Requirements Certificate Policy for the Issuance and Management of Publicly-Trusted Certificates. Available at https://cabforum.org/baseline-requirements-documents/

[17] CA/Browser F orum (2019) Guidelines For The Issuance And Management Of Extended Validation Certificates. Available at https://cabforum.org/extended-validation

[18] Chernick CM, Edington C, III, Fanto MJ, Rosenthal R (2005) Guidelines for the Selection and Use of Transport Layer Security (TLS) Implementations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-52. https://doi.org/10.6028/NIST.SP.800-52

[19] Cooper D, Santesson S, Farrell S, Boeyen S, Housley R, Polk W (2008) Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 5280. https://doi.org/10.17487/RFC5280

PUBLIKACJE ANGLOJĘZYCZNE[43]

[20] Dang QH (2012) Recommendation for Applications Using Approved Hash Algorithms. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-107, Rev. 1. https://doi.org/10.6028/NIST.SP.800-107r1

[21] Dang QH (2011) Recommendation for Existing Application-Specific Key Derivation Functions. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-135, Rev. 1. https://doi.org/10.6028/NIST.SP.800-135r1

[22] Dang QH, Barker EB (2015) Recommendation for Key Management, Part 3: Application-Specific Key Management Guidance. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-57 Part 3, Rev. 1. https://doi.org/10.6028/NIST.SP.800-57pt3r1

[23] Dierks T, Allen C (1999) The TLS Protocol Version 1.0. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 2246. https://doi.org/10.17487/RFC2246

[24] Dierks T, Rescorla E (2006) The Transport Layer Security (TLS) Protocol Version 1.1. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 4346. https://doi.org/10.17487/RFC4346

[25] Dierks T, Rescorla E (2008) The Transport Layer Security (TLS) Protocol Version 1.2. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 5246. https://doi.org/10.17487/RFC5246

PUBLIKACJE ANGLOJĘZYCZNE[43]

[26] Dworkin MJ (2007) Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-38D. https://doi.org/10.6028/NIST.SP.800-38D

[27] Dworkin MJ (2001) Recommendation for Block Cipher Modes of Operation: Methods and Techniques. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-38A. https://doi.org/10.6028/NIST.SP.800-38A

[28] Dworkin MJ (2004) Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-38C, Includes updates as of July 20, 2007. https://doi.org/10.6028/NIST.SP.800-38C

[29] Eastlake D, III, (2011) Transport Layer Security (TLS) Extensions: Extension Definitions. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6066. https://doi.org/10.17487/RFC6066

[30] Eronen P, Tschofenig H (2005) Pre-Shared Key Ciphersuites for Transport Layer Security (TLS). (Internet Engineering Task Force (IETF), Network Working Group), IETF Request for Comments (RFC) 4279. https://doi.org/10.17487/RFC4279

[31] Federal Public Key Infrastructure Policy Authority (2019) X.509 Certificate Policy For The U.S. Federal PKI Common Policy Framework. *https://www.idmanagement.goV/topics/fpki/#certificate-policies*

PUBLIKACJE ANGLOJĘZYCZNE[43]

[32] Freier A, Karlton P, Kocher P (2011) The Secure Sockets Layer (SSL) Protocol Version 3.0. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6101. https://doi.org/10.17487/RFC6101

[33] Gutmann P (2014) Encrypt-then-MAC for Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 7366. https://doi.org/10.17487/RFC7366

[34] Hoffman P, Schlyter J (2012) The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6698. https://doi.org/10.17487/RFC6698

[35] Huitema C, Rescorla E (2018) Issues and Requirements for SNI Encryption in TLS. (Internet Engineering Task Force (IETF) Transport Layer Security Working Group), Internet-Draft draft-ietf-tls-sni-encryption-04 https://datatracker.ietf.org/doc/draft-ietf-tls-sni-encryption/

[36] Joint Task Force Transformation Initiative (2013) Security and Privacy Controls for Federal Information Systems and Organizations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-53, Rev. 4, Includes updates as of January 22, 2015. https://doi.org/10.6028/NIST.SP.800-53r4

[37] Krawczyk H, Eronen P (2010) HMAC-based Extract-and-Expand Key Derivation Function (HKDF). (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 5869. https://doi.org/10.17487/RFC5869

[38] Langley A, (2014) The POODLE bites again. Available at https://www.imperialviolet.org/2014/12/08/poodleagain.html

PUBLIKACJE ANGLOJĘZYCZNE[43]

[39] Langley A, Modadugu N, Moeller B (2016) Transport Layer Security (TLS) False Start. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 7918. https://doi.org/10.17487/RFC7918

[40] Laurie B, Langley A, Kasper E (2013) Certificate Transparency. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6962. *https://doi.org/10.17487/RFC6962*

[41] McGrew D, Bailey D (2012) AES-CCM Cipher Suites for Transport Layer Security (TLS). (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6655. https://doi.org/10.17487/RFC6655

[42] Moeller B, Langley A (2015) TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 7507. https://doi.org/10.17487/RFC7507

[43] Möller B, Duong T, Kotowicz K (2014) This POODLE Bites: Exploiting The SSL 3.0 Fallback. Available at https://www.openssl.org/~bodo/ssl-poodle.pdf

[44] National Institute of Standards and Technology (2001) Advanced Encryption Standard (AES). (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 197. https://doi.org/10.6028/NIST.FIPS.197

[45] National Institute of Standards and Technology (2013) Digital Signature Standard (DSS). (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 186-4. https://doi.org/10.6028/NIST.FIPS.186-4

PUBLIKACJE ANGLOJĘZYCZNE[43]

[46] National Institute of Standards and Technology (2008) The Keyed-Hash Message Authentication Code (HMAC). (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 198-1. https://doi.org/10.6028/NIST.FIPS.198-1

[47] National Institute of Standards and Technology (2013) Personal Identity Verification (PIV) of Federal Employees and Contractors. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 201-2. https://doi.org/10.6028/NIST.FIPS.201-2

[48] National Institute of Standards and Technology (2019) Random Bit Generation. Available at https://csrc.nist. gov/Projects/Random-Bit-Generation

[49] National Institute of Standards and Technology (2015) Secure Hash Standard (SHS). (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 180-4. https://doi.org/10.6028/NIST.FIPS.180-4

[50] National Institute of Standards and Technology (2001) Security Requirements for Cryptographic Modules. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 140-2, Change Notice 2 December 03, 2002. https://doi.org/10.6028/NIST.FIPS.140-2

[51] National Institute of Standards and Technology (2019) Security Requirements for Cryptographic Modules. (U.S. Department of Commerce, Washington, DC), Federal Information Processing Standards Publication (FIPS) 140-3. https://doi.org/10.6028/NIST.FIPS.140-3

PUBLIKACJE ANGLOJĘZYCZNE[43]

[52] Nir Y, Josefsson S, Pegourie-Gonnard M (2018) Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 8422. https://doi.org/10.17487/RFC8422

[53] Paterson KG, Ristenpart T, Shrimpton T (2011) Tag size does matter: attacks and proofs for the TLS record protocol. Advances in Cryptology - ASIACRYPT2011, Lecture Notes in Computer Science, eds Lee DH, Wang X (Springer, Berlin), Vol. 7073, pp 372-389. https://doi.org/10.1007/978-3-642-25385-0 20

[54] Pettersen Y (2013) The Transport Layer Security (TLS) Multiple Certificate Status Request Extension. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6961. https://doi.org/10.17487/RFC6961

[55] Polk T, McKay KA, Chokhani S (2014) Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-52, Rev. 1. https://doi.org/10.6028/NIST.SP.800-52r1

[56] Rescorla E (2008) TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM). (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 5289. https://doi.org/10.17487/RFC5289

[57] Rescorla E (2018) The Transport Layer Security (TLS) Protocol Version 1.3. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 8446. *https://doi.org/10.17487/RFC8446*

PUBLIKACJE ANGLOJĘZYCZNE[43]

[58] Rescorla E, Oku K, Sullivan N, Wood C (2019) Encrypted Server Name Indication for TLS 1.3. (Internet Engineering Task Force (IETF) Transport Layer Security Working Group), Internet-Draft draft-ietf-tls-esni-04. https://datatracker.ietf.org/doc/draft-ietf-tls-esni/

[59] Rescorla E, Ray M, Dispensa S, Oskov N (2010) Transport Layer Security (TLS) Renegotiation Indication Extension. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 5746. https://doi.org/10.17487/RFC5746

[60] Rizzo J, Duong T (2012) The CRIME Attack. EKOparty Security Conference 2012 (Buenos Aires, Argentina). Available at https://www.ekoparty.org/archivo/2012/eko8- CRIME.pdf

[61] Salowey J, Choudhury A, McGrew D (2008) AES Galois Counter Mode (GCM) Cipher Suites for TLS. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 5288. https://doi.org/10.17487/RFC5288

[62] Salowey J, Zhou H, Eronen P, Tschofenig H (2008) Transport Layer Security (TLS) Session Resumption without Server-Side State. (Internet Engineering Task Force (IETF) Network Working Group), IETF Request for Comments (RFC) 5077. ***https://doi.org/10.17487/RFC5077***

[63] Santesson S, Myers M, Ankney R, Malpani A, Galperin S, Adams C (2013) X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6960. ***https://doi.org/10.17487/RFC6960***

PUBLIKACJE ANGLOJĘZYCZNE[43]

[64] Seggelmann R, Tuexen M, Williams M (2012) Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6520. *https://doi.org/10.17487/RFC6520*

[65] Sheffer Y, Holz R, Saint-Andre P (2015) Summarizing Known Attacks on Transport Layer Security (TLS) and Datagram TLS (DTLS). (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 7457. https://doi.org/10.17487/RFC7457

[66] Sönmez Turan M, Barker EB, Kelsey JM, McKay KA, Baish ML, Boyle M (2018) Recommendation for the Entropy Sources Used for Random Bit Generation. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Special Publication (SP) 800-90B. https://doi.org/10.6028/NIST.SP.800-90B

[67] Springall D, Durumeric Z, Halderman JA (2016) Measuring the Security Harm of TLS Crypto Shortcuts. IMC'16 Proceedings of the 2016 Internet Measurement Conference (ACM, Santa Monica, California), pp 33-47. https://doi.org/10.1145/2987443.2987480

[68] Federal Bridge Certification Authority (2019) X.509 Certificate Policy For The Federal Bridge Certification Authority (FBCA). Available at https://www.idmanagement.goV/topics/fpki/#certificate-policies

[69] U.S. General Services Administration (2019) DAP: Digital Analytics Program. Available at https://digital.gov/dap

[70] US-CERT/NIST (2014) CVE-2014-0160 Detail. National Vulnerability Database. Available at https://nvd.nist.gov/vuln/detail/CVE-2014-0160

PUBLIKACJE ANGLOJĘZYCZNE[43]

[71] Wouters P, Tschofenig H, Gilmore J, Weiler S, Kivinen T (2014) Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 7250. https://doi.org/10.17487/RFC7250

[72] Yee P (2013) Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. (Internet Engineering Task Force (IETF)), IETF Request for Comments (RFC) 6818. https://doi.org/10.17487/RFC6818

ZAŁĄCZNIK H–HISTORIA ZMIAN[44]

H.1 WERSJA PIERWOTNA

Pierwotna wersja publikacji NIST SP 800-52 została wydana w czerwcu 2005 roku [18]. W tamtym okresie tylko wersja TLS 1.0 była ostateczna (protokół TLS 1.1 był jeszcze w fazie rozwoju). Protokół TLS 1.1 stał się standardem w kwietniu 2006 roku, a TLS 1.2 w sierpniu 2008 roku. Publikacja NIST SP 800-52 stała się nieaktualna, a wytyczne dotyczące kluczy i zestawów szyfrowania zostały włączone do części 3 publikacji NIST SP 800-57 [22]. W marcu 2013 roku publikacja NIST SP 800-52 została wycofana.

H.2 AKTUALIZACJA 1

Pierwsza aktualizacja publikacji NIST SP 800-52 została wydana w kwietniu 2014 roku [55]. Aktualizacja ta była nowym dokumentem, który niewiele przypominał pierwotną wersję. W tamtym okresie protokół TLS 1.2 nie był jeszcze powszechnie stosowany, a rządowa infrastruktura PKI opierała się głównie na certyfikatach RSA. Biorąc pod uwagę ten fakt, sformułowano zalecenia, aby organizacje państwowe mogły stosować się do wytycznych przy użyciu istniejącej lub opracowywanej technologii. Instytucjom rządowym zalecono opracowanie planu migracji do protokołu TLS 1.2.

Po opublikowaniu aktualizacji 1 wytyczne dotyczące kluczy i zestawów szyfrowania zostały usunięte z części 3 NIST SP 800-57.

H.3 AKTUALIZACJA 2

Od aktualizacji 1 wzrosła popularność TLS 1.2 i zestawów szyfrowania wykorzystujących efemeryczną wymianę kluczy, a także pojawiły się nowe ataki. W aktualizacji 2 (niniejszy dokument) znalazł się wymóg obsługi TLS 1.2 oraz kilka zmian w zaleceniach dotyczących certyfikatów i zestawów szyfrowania.

Aktualizacja 2 obejmuje zalecenia dotyczące protokołu TLS 1.3. Protokół TLS 1.3 oferuje wiele ulepszeń w stosunku do poprzednich wersji, dlatego w aktualizacji 2 zalecono instytucjom rządowym opracowanie planu migracji do TLS 1.3.

[44] Dla zainteresowanych.

W aktualizacji 2 znajduje się również obszerniejsze omówienie ataków na protokół TLS i wytyczne dotyczące zapobiegania im.

W niniejszej aktualizacji zmianie uległy również wymagania dotyczące certyfikatów. W szczególności wymaga się, aby informacje o statusie certyfikatów serwera TLS były udostępniane za pośrednictwem protokołu OCSP. W tej aktualizacji wytycznych dotyczących protokołu TLS obniżono wymagania dotyczące tego, które algorytmy podpisu mogą być wykorzystywane do podpisywania jakich typów kluczy w certyfikatach.