WYDZIAŁ FIZYKI
i INFORMATYKI STOSOWANEJ
Uniwersytet Łódzki

Adrian Ambroziak

Kierunek: informatyka
Specjalność: informatyka stosowana
Specjalizacja: bazy danych i aplikacje internetowe
Numer albumu: 353144

Utwardzanie serwera sieciowego opartego o system Linux

Praca inżynierska
wykonana pod kierunkiem
dr. hab. Kordiana Smolińskiego
w Katedrze Fizyki Teoretycznej
WFiIS UŁ

Łódź 2019

STRESZCZENIE

W pracy przedstawiono przegląd nowoczesnych mechanizmów bezpieczeństwa systemu Linux na przykładzie dystrybucji CentOS 7.6, Debian 9.8.0 oraz Fedora 29 ze szczególnym uwzględnieniem bezpieczeństwa lokalnego i sieciowego systemu. Przeanalizowano metody zabezpieczania serwerów, których konfiguracja występuje najczęściej, czyli web, poczty i serwera plików, a także omówiono istniejące rozwiązania dla systemu Linux. Zasadniczą częścią pracy jest opis procesu konfiguracji serwerów w celu „utwardzenia" systemów, który stanowił praktyczną część pracy dyplomowej. Celem pracy jest wdrożenie „utwardzonego" (hardened) serwera opartego o system Linux oraz stworzenie instrukcji wdrażania utwardzonego serwera (od etapu instalacji do chwili oddania do użytkowania) oraz dokument określający „dobre praktyki" administracyjne i politykę użytkowania.

Słowa kluczowe: Linux, bezpieczeństwo, utwardzanie serwerów

HARDENING OF A LINUX-BASED NETWORK SERVER

The following thesis presents an overview of modern security mechanisms of the Linux system on the example of Centos 7.6, Debian 9.8.0 and Fedora 29 distribution with particular emphasis on local and network security. The methods of securing the most frequently configured servers, i.e. web, mail and file server, as well as the existing solutions for Linux were analyzed. The main part of the paper is a description of the process of server configuration in order to "harden" the systems, which was a practical part of the thesis. The aim of this work is to implement a hardened server based on Linux and to create an implementation manual for the hardened server (from installation to commissioning), as well as a document setting out "good administrative practices" and usage policy.

Keywords: Linux, security, servers hardening

Spis treści

Wstęp

Bezpieczeństwo serwerów zawsze było ważnym tematem. W dzisiejszych czasach jest nawet bardziej istotne niż dawniej. Liczba naruszeń zabezpieczeń w postaci włamań, ataków DDoS czy kradzieży baz danych wzrasta w tempie geometrycznym. Nawet systemy GNU(Linux), które tradycyjnie były mniej podatne na problemy, są przedmiotem ataków. Żyjemy w czasach, o których można powiedzieć, że są złotą erą komputerowego hackingu. Wiele czynności w naszym codziennym życiu odbywa się w Internecie – używamy komunikatorów, tworzymy blogi, strony internetowe, tworzymy treści multimedialne, korzystamy z portali społecznościowych, konsumujemy treści, gramy w gry, czytamy wiadomości, czy też robimy zakupy. Każde z tych działań bazuje na serwerach umieszczonych w centrach danych, podłączonych do Internetu, a te serwery stale są przedmiotem ataków. Zagrożenia i ryzyko we współczesnej, globalnej sieci komputerowej oraz wpływ, jaki wywierają na przeciętnego człowieka, który ma do niej dostęp, są większe, niż kiedykolwiek wcześniej.

Biorąc pod uwagę historie osób takich jak chociażby Kevin Mitnick, należy zwrócić uwagę na to, że coraz częściej motywacją wielu hakerów jest zysk. Luki zero-day w popularnych aplikacjach, czy bazy danych „zhakowanych" popularnych serwisów, wspomagające kradzież danych osobowych, są sprzedawane za dziesiątki, a nawet setki tysięcy dolarów w Darknet na stronach poświęconych tzw. dark market. Coraz częściej czyta się o trojanach, które szyfrują ważne pliki, za których odszyfrowanie hakerzy żądają okupu w kryptowalutach. Zamiast czarnej bluzy z kapturem haker może nosić garnitur i pracować dla organizacji terrorystycznej, dla rządu, jak i dla wojska. Internet na przełomie XX i XXI wieku ulega zmianie, tak samo „(ang.) *hacking*" uległ przemianie i stał się istotną częścią działalności szpiegowskiej, a także działań wojennych. Nawet Polska w ramach obrony terytorialnej tworzy wojsko obrony cyberprzestrzeni, jako komponent, który umożliwi osobom kontynuację kariery informatycznej na rynku cywilnym i jednocześnie podjęcie służby na rzecz obronności Polski w cyberprzestrzeni.

W ostatnich latach działalność hakerów stawała się coraz bardziej wyraźna, do tego stopnia, że informacje o zorganizowanych atakach hakerskich na sieci energetyczne czy rządowe, a także ośrodki nuklearne, nie należą do rzadkości. Hakerzy coraz częściej zorganizowani w grupy są dobrze finansowani i znakomicie wyszkoleni. Dysponują własnymi, zaawansowanymi programami, które piszą sami, mają

opracowane metody działania. Często narzędzia napisane dla takich organizacji jak CIA, NSA, czy KGB trafiają w ręce hakerów i wykorzystywane są zaledwie po roku, lub dwóch od ich napisania. Zdarza się, że informacje o takich programach są ujawnianie przez byłych pracowników służb wywiadowczych, jak miało to miejsce w przypadku programu PRISM ujawnionego przez byłego pracownika CIA Edwarda Snowdena. Ile z takich programów dostało się w niepowołane ręce? Tego nie wiemy. Oznacza to, że model zagrożeń powinien uwzględniać możliwości ataków hakerskich z ostatniego roku.

Obecnie większość przeprowadzanych włamań jest w pełni zautomatyzowana. Cel jest najczęściej jeden – zbudowanie możliwie jak największego zbioru zainfekowanych maszyn tak, aby można je było wykorzystać do nowych ataków. Przetwarzanie danych w chmurze doprowadziło do jeszcze większego rozmycia pojęć sieci „zewnętrznej" i „wewnętrznej". W przeszłości prawidłowe rozdzielenie między tymi sieciami odbywało się poprzez „strefę zdemilitaryzowaną" (ang. *demilitarized zone* – DMZ), w której administrator konfigurował zaporę firewall na granicy pomiędzy obiema tymi sieciami. Wtedy hakerowi było trudniej kupić serwer i wmontować go do szafy serwerowej w serwerowni w firmie. Teraz dzięki usługom takim jak Azure, Amazon Cloud, czy hostingowi, jest to kwestia kilku kliknięć. Należy więc odrzucić założenie, że serwery w chmurze komunikują się ze sobą przez sieć prywatną i zachowywać się tak, jakby każdy pakiet był przesyłany przez sieć publiczną, która z zasady nie jest bezpieczna. Ochrona większości systemów informatycznych polega na zastosowaniu odpowiednich metod zabezpieczających poszczególne warstwy. Analizując bezpieczeństwo systemu, rozważa się zabezpieczenie fizycznego dostępu do urządzeń, zabezpieczenie aplikacji dostępowej, protokołu komunikacji itd. Oprogramowanie, które jest częścią każdego systemu komputerowego, to system operacyjny, pod którego nadzorem pracują aplikacje wykonawcze. Właśnie ze względu na to, że system operacyjny jest nadrzędnym składnikiem niezawodności systemu informatycznego, jego bezpieczeństwo jest bardzo istotne. Z tego powodu pierwsze działania związane z ochroną systemu komputerowego dotyczą zabezpieczenia warstwy systemu operacyjnego. W zależności od specyfiki projektu podjęte czynności mogą mieć różny wymiar i przykładowo mogą sprowadzać się wyłącznie do aktualizacji oprogramowania. Istnieje jednak bardzo wiele sytuacji, które wymagają zastosowania wyszukanych narzędzi i technik dla zapewnienia należytego poziomu bezpieczeństwa systemu operacyjnego.

Cel i układ pracy

W związku z powyższymi spostrzeżeniami, w niniejszej pracy podjęto temat mechanizmów bezpieczeństwa. Systemy operacyjne różnią się znacznie między sobą, z tego też powodu narzędzia, które zostały dla nich stworzone, są zróżnicowane. Ze względu na bardzo dużą liczbę systemów operacyjnych, które uniemożliwiają przeprowadzenie całościowej a przy tym wnikliwej analizy wszystkich dostępnych mechanizmów, w pracy skoncentrowano się wyłącznie na nowoczesnych środkach bezpieczeństwa przygotowanych dla systemu Linux. System ten jest powszechny w rozwiązaniach serwerowych, wbudowanych, a także na urządzeniach mobilnych. Wysokie bezpieczeństwo, które składa się na niezawodność Linux ma w wielu zastosowaniach istotne znaczenie przemysłowe, co czyni opisywaną tematykę szczególnie interesującą. Celem pracy jest wdrożenie „utwardzonego" (ang. *hardened*) serwera opartego o System Linux i zapewnienie bezpieczeństwa lokalnego i sieciowego systemu. Aby zrealizować sformułowane zadanie, w pierwszej kolejności przeanalizowano najnowsze dystrybucje Debian 9.8.0, CentOS 7.6, Fedora 29 pod kątem zastosowanych w nich środków ochrony. Podczas eksploracji mechanizmów bezpieczeństwa zidentyfikowano zapotrzebowanie na narzędzia służące do ochrony ww. systemów operacyjnych. W kolejnym kroku przeprowadzono analizę dostępnych rozwiązań powyższego problemu. Wnioski wyciągnięte na podstawie przeglądu doprowadziły do sformułowania wymagań i zaprojektowania na tej podstawie instrukcji wdrażania utwardzonego serwera od etapu instalacji do chwili oddania go do użytkowania oraz dokumentu określającego dobre praktyki administracyjne i politykę użytkowania. Układ rozdziałów niniejszej pracy w naturalny sposób odzwierciedla przebieg powyższego procesu. W rozdziale I, który jest częścią teoretyczną, zaprezentowano różne rodzaje zabezpieczeń związanych z bezpieczeństwem lokalnym systemu na przykładzie ww. dystrybucji Linux, poprzedzone określeniem celów i potrzeb, które doprowadziły do wdrożenia konkretnych rozwiązań. Kolejny rozdział koncentruje się na praktycznej stronie bezpieczeństwa w zależności od wybranej roli sieciowej serwera i przedstawia typowe zagadnienia oraz rozwiązania towarzyszące wybranej roli. Rozdział ten kończy się pełną instrukcją opisującą instalację i konfigurację poszczególnych elementów w celu zabezpieczenia serwera, a także dokumentem zawierającym informacje, które określają prawidłowe praktyki, którymi powinien na kierować się każdy administrator oraz politykę użytkowania serwera.

1. Aspekty bezpieczeństwa Linux – teoria.

1.1. System operacyjny Linux

Linux jest komputerowym systemem operacyjnym (OS) dość podobnym do Unix OS i w większości przypadków jest zgodny z POSIX-em i jest rozwijany w modelu wolnego i otwartego oprogramowania i dystrybucji. Linux został pierwotnie stworzony tak, aby był podobny do systemu Unix. Oba mają podobne narzędzia do współpracy z systemami, narzędzia programistyczne, układy systemów plików i inne kluczowe komponenty. Jednak Uniks nie jest wolny. Przez lata powstało wiele różnych systemów operacyjnych, które próbowały być "uniksopodobne" lub "kompatybilne z uniksem", ale Linux okazał się najbardziej udany, znacznie przewyższając popularność swoich poprzedników. Linux jest najbardziej znanym i najczęściej używanym systemem operacyjnym open source. Dla celów niniejszej pracy, używam terminu "Linux" w odniesieniu do jądra Linuksa, ale również zestawu programów, narzędzi i usług, które są zazwyczaj łączone razem z jądrem Linuksa, aby zapewnić wszystkie niezbędne komponenty w pełni funkcjonalnego systemu operacyjnego. Pod wieloma względami Linux jest podobny do innych systemów operacyjnych, takich jak Windows, OS X lub iOS. Podobnie jak inne systemy operacyjne, Linux posiada również graficzny interfejs i rodzaje oprogramowania, do korzystania, z których jesteśmy przyzwyczajeni na innych systemach operacyjnych, takich jak aplikacje do przetwarzania tekstu.

Niemniej jednak, Linux różni się również od innych systemów operacyjnych na wiele różnych sposobów. Przede wszystkim Linux jest oprogramowaniem typu open source; kod używany do tworzenia Linuksa jest darmowy i dostępny publicznie do przeglądania, edytowania i dla użytkowników z odpowiednimi umiejętnościami istnieje możliwość wzięcia udziału w jego tworzeniu. Wreszcie, Linux jest również inny w tym, że chociaż podstawowe elementy systemu operacyjnego Linux są na ogół powszechne, istnieje wiele dystrybucji Linuksa, które zawierają różne opcje oprogramowania. Oznacza to, że Linux jest niesamowicie konfigurowalny, ponieważ użytkownicy Linuksa mogą wybierać podstawowe komponenty, takie jak to, który system wyświetla grafikę, którą preferują, oraz decydują o tym, jakie aplikacje chcą mieć zainstalowane. Na ten system Linux istnieje też niewielka ilość wirusów. To jest powód, dla którego zdecydowałem się używać Linuksa do celów tej pracy.

1.2. Bezpieczeństwo lokalne

W tym rozdziale zaprezentuję ogólne zasady dotyczące bezpieczeństwa, które mogą zostać zastosowane podczas zabezpieczania serwera Linux. Następnie przejdę do omówienia jednego z największych problemów dotyczących zabezpieczeń, z którymi spotkać można się na co dzień, a mianowicie haseł. W kolejnych podrozdziałach skupię się na rozwinięciu zagadnień dotyczących zabezpieczeń z naciskiem na kontrolę integralności lokalnego systemu plików, politykę bezpieczeństwa kontroli dostępu i mechanizmy z nią związane zapewnianie przez moduły LSM (ang. *Linux Security Modules* – Moduły Bezpieczeństwa Linux), które stanowią część jądra systemu Linux. Następnie przejdę do kontroli antywirusowej realizowanej przez skaner antywirusowy ClamAV. Ostatnim etapem tego rozdziału będzie szczegółowe omówienie sposobów monitorowania, analizy i korelacji dzienników systemowych i dzienników zdarzeń.

1.2.1. Kontrola nad rozruchem systemu

W tym podrozdziale opiszę dokładniej mechanizmy działania systemu Linux, aby przekonać się, co się dzieje w momencie rozruchu (ang. *boot*) systemu. Przeanalizuję poszczególne etapy tego procesu i zaprezentuję sposoby uruchamiania systemu w różnych trybach, a także objaśnię, jak konfigurować i poprawnie modyfikować rozruch systemu.

Rozruch systemu tzw. ładowanie początkowe polega na działaniu trzech oddzielnych, lecz wzajemnie powiązanych ze sobą elementów: BIOS (ang. *Basic Input/Output System* – podstawowy układ wejść/wyjść) lub UEFI (ang. *Unified Extensible Firmware Interface* – ujednolicony rozszerzalny interfejs oprogramowania sprzętowego), programu rozruchowego (ang. *boot loader*) oraz wczytywania systemu operacyjnego. Te trzy elementy odpowiedzialne są za poszczególne etapy, które można podzielić w następujący sposób:

1. Środowisko BIOS/UEFI jest odpowiedzialne za uruchomienie i sprawdzenie podzespołów sprzętowych,
2. Program rozruchowy (GRUB2) pozwala wybrać zainstalowany system operacyjny,
3. Systemd odpowiada za wczytanie i inicjację systemu operacyjnego.

Nie jest to zjawisko specyficzne tylko i wyłącznie dla systemów z rodziny Linux. Wiele systemów operacyjnych wykorzystuje podobny zestaw funkcji i czynności. Nowe

podzespoły zawierają obsługę środowiska UEFI, które zastępuje starsze środowisko BIOS, wykorzystywane we wcześniejszych urządzeniach. Zarówno BIOS, jak i UEFI pełnią zbliżone funkcje i omówię obydwa, zaczynając od starszego, czyli BIOS.

1.2.1.1. BIOS/UEFI

Po wciśnięciu przycisku zasilania serwera następuje rozruch, za który odpowiada BIOS znajdujący się w niewielkim układzie scalonym na płycie głównej, zwany układem BIOS. Środowisko BIOS przeprowadza podczas rozruchu podstawowe testy systemowe, albo procedurę POST (ang. *power-on-self-test* – samodzielny test po włączeniu zasilania), aby sprawdzić dostępność poszczególnych elementów komputera takich jak pamięć operacyjną, dyski twarde, klawiaturę czy kartę graficzną. Domyślnie w pierwszej kolejności przeszukiwane są zamontowane dyski twarde, a w dalszej kolejności sprawdzane są pozostałe nośniki, takie jak napęd CD/DVD, czy napędy USB. Tutaj istotne dla bezpieczeństwa jest wyłączenie napędów płyt i USB, pozostawienie tylko dysków twardych w ustalonej wcześniej kolejności przeszukiwania i zabezpieczenie BIOS hasłem w celu uniemożliwienia wprowadzenia zmian osobom do tego nieupoważnionym. Jest to pierwsza linia obrony przed atakiem od wewnątrz. Przy czym dobrą praktyką jest, aby nie zabezpieczać dysku twardego hasłem w BIOS, lecz jedynie zmianę ustawień, co sprawi, że BIOS nie będzie pytać o hasło po każdym zaplanowanym restarcie serwera.

Nowsze płyty główne posiadają wbudowany interfejs UEFI. Różnica w rozruchu pomiędzy BIOS i UEFI jest niewielka. Mechanizmy działania obu środowisk są jednak zupełnie inne. Istnieją rodzaje złośliwego oprogramowania, które dostają się do OS (ang. *Operating System* – system operacyjny) poprzez program rozruchowy. W interfejsie UEFI dostępny jest mechanizm „bezpiecznego rozruchu”. Podpisane programy rozruchowe oraz podzespoły są przed wczytaniem weryfikowane za pomocą kluczy publicznych/prywatnych. W swoich rozwiązaniach na dwóch starszych modelach laptopów z Debian i CentOS używany jest BIOS, nowszy laptop z Fedora posiada UEFI.

1.2.1.2. Program rozruchowy GRUB2

Gdy zostaną zainicjowane wszystkie podzespoły dołączone do płyty głównej oraz zastosowane zostaną podstawowe ustawienia, serwer jest gotowy do rozruchu

systemu. Zarówno BIOS jak i UEFI potrafią uruchomić kod wgrany podczas instalacji dystrybucji. Kod ten jest tzw. programem rozruchowym i musi być umieszczony w odpowiednim miejscu na dysku. BIOS w tym celu używa MBR, który można nazwać głównym rekordem rozruchowym umieszczonym w specjalnym sektorze dysku twardego zwanym też sektorem rozruchowym. Po uruchomieniu GRUB2 znajduje, wczytuje i przekazuje system operacyjny do dyspozycji administratora systemu. Proces ten składa się z dwóch etapów. Jak już wspomniałem podczas omawiania rekordu MBR, istnieje wydzielony obszar przed tablicą partycji o rozmiarze 466 bajtów, w którym przechowywane są informacje o pierwszym etapie. W tym właśnie obszarze znajduje się plik o nazwie boot.img. Następnie po uruchomieniu boot.img BIOS wyszukuje i wykonuje kod zawarty w pliku core.img. Domyślnie plik core jest umieszczony w przerwie MBR (ang. *MBR gap*). Jest to przestrzeń pomiędzy MBR a pierwszą partycją. Rolą pliku core.img jest uzyskanie dostępu do katalogu /boot/grub i wczytanie zawartych tam modułów. Plik ten wywołuje menu rozruchowe i jest w stanie wczytać docelowy system operacyjny.

W przypadku UEFI proces uruchomienia systemu przebiega w inny sposób. UEFI potrafi odczytywać tablicę GPT, a także wyszukać oraz uruchomić kod rozruchowy zawarty w katalogu /EFI/ na partycji systemowej EFI. Interfejs UEFI, w przeciwieństwie do BIOS, ma własną partycję, na której podczas instalacji kopiowane są program rozruchowy oraz moduły rozruchowe.

W momencie, gdy interfejs UEFI jest gotowy, to szuka programu rozruchowego na wspomnianej partycji. W powyższym przypadku zostanie odnaleziony plik /EFI/fedora/grubx64.efi. Na tym etapie program rozruchowy wykrywa partycję /boot, na której przechowywane jest oprogramowanie GRUB2 i wczyta zawartość pliku core.efi, co spowoduje wyświetlenie menu GRUB2. Jeżeli odczekamy, aż skończy się odliczanie, to program rozruchowy wczyta domyślne jądro systemu. W kolejnym etapie GRUB2 wyszuka plik binarny jądra (vmlinux-<numer_wydania>), następnie wczyta do pamięci komputera plik initrd.img, w którym są przechowywane sterowniki wymagane przez jądro do obsługi podzespołów komputera. Wczytanie pliku initrd.img jest ostatnią czynnością wykonywaną przez program GRUB2, w następnej kolejności kontrolę nad serwerem przejmuje jądro, które odpowiedzialne jest za kontynuację procesu rozruchu poprzez inicjację urządzeń, w tym dysków twardych. Zostaje uruchomiony system operacyjny i następuje wywołanie specjalnego programu Systemd.

1.2.1.3. Zabezpieczenie programu rozruchowego GRUB2

GRUB2 umożliwia rozruch w trybie pojedynczego użytkownika (ang. *single-user mode*), który nazywany jest także trybem administracyjnym (ang. *maintenance mode*). Tego trybu używa się w przypadku awarii systemu operacyjnego serwera. W tej sytuacji dostęp ograniczony jest jedynie do konta root (konto administratora z najwyższymi uprawnieniami) i zazwyczaj poziomu konsoli systemowej. Umożliwia to pracę z dyskami i plikami bez obawy o występowanie konfliktów lub interferencje ze strony innych użytkowników. Uruchamianie systemu w trybie *user-single mode* przebiega nieco odmiennie w dystrybucjach CentOS/Fedora i Debian. We wszystkich przypadkach wyświetlane jest podobne menu GRUB2, w którym możliwe jest wybranie jądra systemu, które następnie zostanie wczytane. Dystrybucje Fedora oraz CentOS zawierają dwie opcje, czyli wybór właściwego jądra a także jądra trybu odzyskiwania (ang. *recovery mode kernel*). W przypadku Debian widoczny jest wpis podmenu nazwany Opcje zaawansowane dla systemu Debian *(*ang. *Advanced Options for Debian GNU/Linux*). W omawianym przykładzie wykorzystam standardowe jądro Debian 9.8.0. Aby edytować wybrany kernel trzeba wcisnąć klawisz E. Wyświetlony zostanie kod konfiguracji tego jądra, w tym ustawienia jego lokalizacji i parametrów. Identyczne informacje znajdują się w pliku grub.cfg. W tym konkretnym przypadku interesuje nas wiersz linux /vmlinuz-4.9.0-8-amd64… Za pomocą klawiszy strzałek przechodzę na koniec tego wiersza. Jeśli chcę przeprowadzić rozruch w trybie pojedynczego użytkownika, wpisuję tutaj wyraz single. Po naciśnięciu kombinacji ctrl+x można wejść w tryb pojedynczego użytkownika. W tym momencie warto zwrócić uwagę na fakt, że niezabezpieczony program rozruchowy może zostać zmodyfikowany w trakcie rozruchu przez osobę mającą wrogie zamiary. GRUB2 daje możliwość zdefiniowania hasła, co sprawia, że wszelkie zmiany w przygotowanym procesie rozruchowym będą wymagały jego wprowadzenia. Zabezpieczenie GRUB2 jest jedną z podstawowych zasad bezpieczeństwa i stało się standardem. Podczas każdego audytu zabezpieczeń ten punkt jest sprawdzany jako jeden z pierwszych. Zabezpieczenie programu rozruchowego GRUB opisane jest w części praktycznej, w instrukcji wdrażania utwardzonego serwera.

1.2.1.4. Aplikacja Systemd

W dalszym etapie startu systemu, gdy są uruchamiane usługi systemowe, działają serwisy Systemd, które są przenośne pomiędzy dystrybucjami. Dzięki temu

deweloperzy jakiejś usługi mogą dostarczyć razem z nią serwis Systemd, który zapewni bezproblemowe zarządzanie usługą na każdym systemie wspierającym Systemd. Nie trzeba już dla każdej dystrybucji pisać oddzielnych skryptów startowych.

Usługi (ang. *daemons*) odpowiadają za wiele istotnych dla działania serwera funkcji. Każda usługa stanowi co najmniej jeden proces działający na serwerze. Do znanych demonów zaliczają się: `master` (serwer pocztowy Postfix), `httpd` (serwer sieciowy Apache), czy `mysqld` (serwer bazodanowy MySQL). Niektóre procesy mogą być uruchamiane jako domyślne wraz z innymi procesami, które są odpowiedzialne za prawidłowe funkcjonowanie systemu i programów. Nazwy większości procesów utrzymujących demony kończą się na literę „d" (od wyrazu „daemon").

Najważniejszy z procesów to Systemd, który jest odpowiedzialny za uruchomienie wszystkich innych procesów na serwerze. Systemd jest nadrzędnym procesem z PID o numerze 1 i działa bez przerwy, by móc zagwarantować poprawne działanie systemu operacyjnego. Program inicjujący Systemd wyszukuje niezbędne usługi i wymagane przez nie zasoby poprzez odczytanie instrukcji z szeregu plików.

Systemd występuje natywnie w dystrybucjach, które wykorzystuję w mojej pracy CentOS 7.6 oraz Fedora 29. Natomiast w dystrybucji Debian 9.8.0 identyfikator PID 1 jest przydzielony do programu */sbin/init* (rysunek 1.1). Inicjator ten pochodzi z pakietu *systemd-sysv* i wiąże się z Systemd. W chwili obecnej Debian korzysta z trzech rodzajów inicjatorów podczas przechodzenia na natywną obsługę Systemd.

Wszystkie trzy wymienione dystrybucje posiadają zainstalowany pakiet *systemd-sysv*, w którym umieszczone są narzędzia, które pozwalają na kompatybilne uruchamianie skryptów *initd* przypominających implementację *SysVInit*. Systemd-sysv został zaprojektowany tak, by móc zastąpić implementację *SysVInit*, jednak nie są one ze sobą całkowicie kompatybilne.

Systemd staje się powszechny we współczesnych dystrybucjach Linux. Zastępuje on *SysV* oraz *Upstart*. Cechuje się wieloma zaletami, które wyróżniają go na tle poprzedników.

- Reaguje na zdarzenia systemowe np. podłączenie nowego urządzenia.
- Potrafi przywracać procesy.
- Jest odpowiedzialny za tworzenie raportów zdarzeń.
- Równolegle przetwarza elementy rozruchowe.
- Śledzi procesy za pomocą funkcji Cgroups jądra systemu.

Inicjator ten wykorzystuje do swojego działania koncepcję celów (ang. *targets*). Cele są odpowiedzialne za grupowanie zależności pomiędzy usługami. Powszechnie używane cele to *boot, rescue, multiuser*. Cel *multiuser* wykorzystywany jest w sytuacji, gdy chcemy mieć system, w którym wszyscy użytkownicy są w stanie zalogować się i korzystać z usług. W tym stanie powinniśmy udostępnić usługi sieciowe, protokoły ssh i logowania, które są „pożądanymi" elementami stanu *multiuser*.

1.2.2. Implementacja polityki kont i haseł

Hasła to jedna z podstawowych metod uwierzytelniania. Hasła powinny być znane tylko użytkownikowi i takie, aby nie można było ich odgadnąć. Mimo, że obecnie techniki łamania haseł są bardzo złożone i szybkie, to nadal wybierane są hasła, które są proste i łatwe do odgadnięcia. Ataki na hasła stały się wyrafinowane. Metody brute force korzystające ze słowników zawierających setki tysięcy słów lub też wykorzystujące ataki siłowe, polegające na podejmowaniu wielu prób odgadnięcia hasła jak to możliwe (w przypadku hasła, które jest 6-cyfrową liczbą mogą one rozpocząć od 000000 i próbować wszystkich kombinacji, aż do 999999 – oznacza to milion prób) są dość częste. Może się wydawać, że to zajęłoby dużo czasu, ale jeśli weźmiemy pod uwagę, że nowoczesne komputery są w stanie wykonać setki tysięcy lub miliony prób odgadnięcia hasła na sekundę, to okaże się, że hasła krótkie lub takie, które można znaleźć w słowniku, są dla napastnika trywialne do odgadnięcia. Niestety, w wielu firmach wciąż stosuje się przestarzałą politykę dotyczącą haseł i w rezultacie użytkownicy wybierają słabe hasła, łatwe do odgadnięcia dla cyberprzestępców. W dalszej części tego rozdziału omówię najważniejsze ograniczenia stosowane w polityce dotyczącej haseł i napiszę o tym, co w dzisiejszym świecie się sprawdza, a co nie. Na koniec opiszę dobrą politykę dotyczącą haseł.

1.2.2.1. Użytkownicy i grupy

Linux umożliwia wielu użytkownikom łączyć się jednocześnie z daną dystrybucją przy użyciu oddzielnych sesji interfejsów tekstowych lub graficznych. Dostęp do serwera i jego zasobów kontrolowany jest poprzez konta użytkowników i grupy. Konta użytkowników tworzone są pod kątem określonych składników systemowych oraz używane do uruchamiania i utrzymywania usług. Przykładowo podczas instalacji serwera pocztowego zostanie także utworzony użytkownik mail współdziałający z usługą pocztową. W Linux wykorzystuje się także koncepcję grup,

czyli zbiorów podobnych użytkowników. Użytkownicy mogą być członkami jednej lub większej liczby grup i zazwyczaj są umieszczani w określonej grupie, aby mogli otrzymać dostęp do wyznaczonych zasobów. Przykładowo wszyscy użytkownicy, którzy powinni mieć dostęp do systemu transakcji finansowych, mogą zostać dodani do wspólnej grupy nazwanej transakcje. Istotni są zarówno użytkownicy, jak i grupy, dlatego omówię politykę zakładania kont użytkowników i tworzenia grup oraz ich działanie.

W Linux każdy użytkownik posiada konto, które zazwyczaj jest chronione hasłem. Podczas tworzenia większości użytkowników powstaje także katalog domowy `/home/nazwa_użytkownika`, w którym użytkownicy mogą przechowywać swoje dane; jest to również domyślna lokalizacja dla plików konfiguracyjnych wielu aplikacji. Użytkownicy są przypisani do grup, dzięki czemu uzyskują dostęp do dodatkowych zasobów lub usług. Informacje o użytkownikach i grupach przechowywane są przede wszystkim w dwóch plikach: `/etc/passwd` zawiera informacje o użytkowniku, a `/etc/group` – o grupie. Istnieją standardowe grupy dające pełne uprawnienia (root) w systemach Linux takie, jak: admin, wheel i sudo. Każdy użytkownik serwera linuksowego musi należeć do co najmniej jednej grupy, zwanej grupą główną (ang. *primary group*), może jednak być również członkiem innych grup, tak zwanych grup dodatkowych (ang. *supplementary group*).

Prawo własności pliku dla użytkownika i grupy możemy zmienić przy pomocy polecenia chown. Jedynie konto root ma możliwość zmiany przynależności pliku (chociaż możliwe jest rozszerzenie tego prawa przy użyciu komendy sudo, której poświęcę uwagę w sekcji A.2). Dostępna jest także komenda chgrp. Używając jej użytkownik może zmieniać przynależność pliku do grupy, z tym że można przydzielić prawo własności jedynie grupie, do której przynależy użytkownik. Składnia tego polecenia wygląda następująco: `chgrp nazwa_grupy plik.`

Tworzenie użytkowników i grup jest bardzo proste: w pierwszym przypadku używane jest polecenie `useradd`, a w drugim – `groupadd`. Ponadto istnieje możliwość modyfikowania kont użytkowników i grup przy pomocy komend, odpowiednio: `usermod` i `groupmod`. Natomiast usunięcie konta użytkownika następuje przy użyciu polecenia `userdel`, a grupę usuniemy, korzystając z komendy `groupdel`. Hasło ustawiamy komendą `passwd nazwa_użytkownika`.

1.2.2.2. Polityka kont użytkowników i grup

Administratorem w Linux może być tylko użytkownik root. Administrator ma pełne prawa i nikt mu ich nie może odebrać. Każdy użytkownik powinien mieć swój katalog domowy. Domyślnie profile tworzone są w katalogu /home i nazwą jest nazwa użytkownika, np. /home/użytkownik, jednak nie istnieją żadne ograniczenia, które powodują brak możliwości zmiany lokalizacji i nazwy katalogu domowego. Wyjątkiem jest tutaj użytkownik root, który katalog domowy ma w folderze /root. Zalecanym rozwiązaniem podczas instalacji systemu jest, aby katalogi domowe znajdowały się na osobnej partycji przeznaczonej dla katalogu /home, na której uruchomiony jest system quota, czyli mechanizm sterowania przydziałem miejsca na dysku dla poszczególnych użytkowników i grup. Użytkownicy przynależą do jednej lub wielu grup i prawa dostępu do plików, których nie są właścicielami, zależą od tej przynależności. Pliki utworzone przez użytkownika należą do jego aktualnej grupy. Grupa ta jest inicjalizowana w momencie otwarcia sesji, dzięki plikowi /etc/passwd.

Nazwy kont użytkowników powinny być niepowtarzalne i zgodne z aktualnie przyjętymi regułami. Użytkownicy powinni mieć silne, losowo wygenerowane hasła oraz numery UID i GID przyporządkowane rosnąco dla każdego użytkownika. Nie należy stosować kont współdzielonych, niezależnie od tego, czy chodzi o konta powłoki, konta interfejsów webowych, czy też konta usługowe. Konta współdzielone oznaczają wspólne hasła w zespole, co samo w sobie jest złą praktyką. Często konta takie są uprzywilejowane, dlatego w przypadku, gdy jeden z członków zespołu go opuści, powstaje potrzeba ciągłego utrzymywania haseł wspólnych kont oraz bezpiecznej dystrybucji nowego hasła dla wszystkich członków zespołu. Indywidualne konta pozwalają ustalić, jaka osoba zalogowała się na konto i pozwala z łatwością wyśledzić, czy podjęte zostały jakieś destruktywne działania. Współdzielone konta nie są konieczne w żadnym nowoczesnym systemie, w którym zagadnienia związane z bezpieczeństwem mają istotne znaczenie. Zawsze powinniśmy zachować zdolność do tworzenia uprzywilejowanych ról i przypisywania tym rolom indywidualnych kont. W przypadku kont powłoki powinno to przyjąć formę kont grup lub ról, do których indywidualni użytkownicy logują się za pośrednictwem narzędzia sudo. Natomiast w przypadku innych usług administrator zobowiązany jest do zadbania o utworzenie uprzywilejowanej grupy, do której można dodawać indywidualnych użytkowników.

Tymczasowe blokowanie kont użytkowników powinno wykonywać się poprzez polecenie: `$ sudo pw lock nazwa_użytkownika`. Odblokowanie konta następuje poprzez wykonanie komendy: `$ sudo pw unlock nazwa_użytkownika`.

Możliwe jest też inne rozwiązanie - wystarczy dopisać gwiazdkę lub inny znak na początku zaszyfrowanego hasła użytkownika w pliku /etc/shadow albo w pliku /etc/master.passwd. Dzięki temu większość prób dostępu do hasła nie będzie możliwych, gdyż nie jest możliwe sensowne odszyfrowanie zmodyfikowanego w ten sposób hasła. Wszystkie dystrybucje Linux umożliwiają łatwe zablokowanie i odblokowanie hasła za pomocą poleceń `usermod –L` użytkownik oraz `usermod –U` użytkownik. Opcja L (ang. *lock*) dodaje znak ! na początku zaszyfrowanego hasła w pliku /etc/shadow, natomiast opcja –U (ang. *unlock*) go usuwa. Innym sposobem wyłączenia tymczasowo konta jest zastąpienie powłoki użytkownika programem, który wyświetli komunikat o blokadzie konta i wyjaśni, jakie czynności należy podjąć w takiej sytuacji.

1.2.2.3. Polityka haseł

Jedną z najbardziej istotnych kwestii w przypadku polityki dotyczącej haseł jest koncepcja wprowadzenia wymagań dotyczących długości hasła, co ma na celu zwiększenie liczby kombinacji w przypadku próby odgadnięcia hasła przez napastnika przeprowadzającego atak metodą siłową. Zwiększenie liczby znaków w haśle sprawia, że atak siłowy staje się dużo trudniejszy. Zwiększenie minimalnej długości hasła chociażby tylko o jeden znak znacznie zwiększa liczbę możliwych kombinacji. Jednak trzeba wziąć pod uwagę fakt, że napastnik posługujący się metodą siłową, który potrafi wykonać milion prób na sekundę, zdoła złamać 6-znakowe hasło w ciągu 5 minut, a 8-znakowe w 2,5 dnia. Natomiast sprawdzenie wszystkich możliwych kombinacji dla hasła 12-znakowego zajmie już 3026 lat. To właśnie z tego powodu napastnicy znacznie częściej decydują się na zastosowanie ataku słownikowego zamiast siłowego. Jeśli 8-znakowe hasło jest słowem ze słownika, to złamanie go za pomocą ataku siłowego zajmie wiele dni, natomiast złamanie go metodą słownikową potrwa kilka sekund. Atak siłowy można utrudnić poprzez wprowadzenie wymagania dotyczącego złożoności hasła. Należy wymagać, aby hasło zawierało litery i przynajmniej jedną cyfrę lub mieszankę wielkich liter z małymi oraz znakami specjalnymi. Co prawda zwiększanie złożoności haseł znacznie zwiększa całkowitą liczbę kombinacji haseł, jednakże ten wzrost jest wielokrotnie mniejszy w porównaniu ze zwiększeniem minimalnej długości

hasła. 12-znakowe hasło pisane wyłącznie małymi literami wciąż ma o rząd wielkości więcej kombinacji niż 8 znakowe hasło o pełnej złożoności.

Wymuszenie rotacji haseł co kwartał lub miesiąc i wprowadzenie obowiązkowego wymogu, aby nie można było wprowadzić dwa razy takiego samego hasła jest mało skuteczne. Koncepcja rotacji haseł wynika z pragmatycznego podejścia do faktu, że nawet jeśli czyjeś hasło zostanie złamane, to to napastnik będzie miał tylko ograniczony czas na to, aby skorzystać z uzyskanego dostępu, zanim hasło zostanie zmienione. Jednak, gdy napastnikowi uda się złamać hasło i uzyskać dostęp do konta, to przystępuje natychmiast do działania. Zanim hasło zostanie zmienione, napastnik dokona szkód i zniknie. Ponadto hakerzy zwykle wykorzystują uzyskany dostęp po to, aby utworzyć rodzaj „tylnego wejścia" (ang. *backdoor*), które da mu dostęp nawet wtedy, gdy hasło ulegnie zmianie. Wprowadzenie częstej rotacji haseł zwiększa prawdopodobieństwo, że użytkownicy będą wybierać nowe hasło, które będzie bardzo podobne do poprzedniego. W strategii dla użytkowników zaleca się metodę prostoty. Hasła muszą mieć co najmniej 12 znaków, bez wymagań co do złożoności. Dodatkowo należy zachęcać użytkowników do wybierania jako haseł fraz (ang. *passphrase*) zamiast słów (ang. *password*). Hasło w postaci frazy to połączenie kilku słów. Najlepiej, jeśli będą one losowe. Hasła root i administratorów należy zmieniać przynajmniej co 6 miesięcy, za każdym razem, gdy ktoś, kto miał do nich dostęp kończy stosunek pracy, lub zmienia stanowisko, gdy istnieje podejrzenie naruszenia bezpieczeństwa.

1.2.3. Kontrola integralności lokalnego systemu plików

Weryfikacja integralności systemu, często jest nazywana monitoringiem integralności plików (FIM, ang. *file integrity monitoring*), jest sprawdzaniem bieżącego stanu systemu w stosunku do znanego stanu wyjściowego. Zazwyczaj taka walidacja porównuje zawartość plików systemowych (zmiany w plikach konfiguracyjnych, jądro systemu, polecenia wykonywalne) z kryptograficzną sumą kontrolną, taką jak SHA-512. Gdy suma kontrolna pliku w uruchomionym systemie różni się od sumy kontrolnej wersji początkowej, zwanej wyjściową, administrator systemu Linux zostanie o tym powiadomiony. Warto tutaj zwrócić uwagę na regularnie wykonywane działania konserwacyjne, takie jak planowane zmiany, poprawki, aktualizacje, dlatego nie wszystkie zmiany są podejrzane.

1.2.3.1. Narzędzia do monitorowania zmian lokalnego systemu plików

API powiadomień systemu plików umożliwia aplikacjom oglądanie określonych plików i otrzymywanie powiadomień o ich otwarciu, modyfikacji, usunięciu lub zmianie nazwy. To bardzo pomaga aplikacjom, ponieważ zanim takie narzędzia do monitorowania zdarzeń w systemie plików istniały, takie aplikacje musiałyby wielokrotnie odczytywać dysk w celu wykrycia wszelkich zmian, co skutkowało wysokim zużyciem dysku i CPU. Rozwiązaniem jest kilka narzędzi, które można wykorzystać do monitorowania zdarzeń w systemie plików.

- FAM (File Alteration Monitor) - jest to jeden z najstarszych przenośnych monitorów zdarzeń. Wysyła zdarzenia do aplikacji, gdy wprowadzane są zmiany w plikach lub katalogach, które aplikacja zarejestrowała w celu monitorowania. Jest dość skomplikowany.
- Gamin - nowszy i prostszy niż FAM. Stara się być kompatybilny z FAM, nie implementując przy tym wielu niejasnych funkcji. Jest utrzymywany i szeroko dostępny w wielu dystrybucjach Linux. Jest przenośny, ale rozwój i testowanie skupia się na Linux. Posiada wsparcie BSD i można go znaleźć w Portach FreeBSD.
- dnotify - wprowadzony od wersji jądra Linux od wydania 2.4. Może oglądać tylko katalogi i wymaga utrzymywania otwartego deskryptora plików do katalogu, który użytkownik chce oglądać. Został zdezaktualizowany przez inotify.
- inotify - zastąpił dnotify. Jest częścią podsystemu jądra Linux i wchodzi w skład jądra Linux od wydania 2.6.13. Jest szybki i lekki i powinien być dostępny we wszystkich dystrybucjach Linux.

Jednymi z częściej stosowanych platform FIM są Tripwire i OSSEC. Standardowym rozwiązaniem FIM jest program mtree, który pozwala w prosty sposób monitorować stan plików i zmian ich zawartości. Jest łatwy do zintegrowania ze skryptami monitorującymi. Stworzenie platformy FIM a jej obsługa w czasie to dwie zupełnie różne sprawy. Do przechowywania danych i odpowiadania na powiadomienia FIM potrzeba zdefiniować odpowiedni proces.

1.2.4. Polityka bezpieczeństwa kontroli dostępu

Standardowy model kontroli dostępu od wielu lat pozostaje w dużej mierze niezmieniony. Pomimo tego, że wprowadzono do niego kilka udoskonaleń, to nadal jest

on domyślnym rozwiązaniem. Schemat tego modelu oparty jest na kilku podstawowych zasadach:

- Decyzje, które dotyczą kontroli dostępu zależą od tego, który użytkownik próbuje wykonać operację, a w niektórych przypadkach od członkostwa tego użytkownika w grupie.
- Obiekty (pliki i procesy) posiadają właścicieli. Właściciele mają kontrolę nad swoimi obiektami, jednak nie jest ona nieograniczona.
- Obiekty tworzone przez danego użytkownika są jego własnością.
- Specjalne konto użytkownika o nazwie root działa jak właściciel każdego obiektu.
- Tylko użytkownik root może przeprowadzać pewne wrażliwe operacje administracyjne.

W standardowym modelu każdy plik posiada równocześnie właściciela i grupę. Właściciel może określać uprawnienia do swoich plików i katalogów przy pomocy polecenia `chmod`. Wyróżniamy trzy podstawowe typy uprawnień:

- odczyt — reprezentowany przez literę r (ang. *read*),
- zapis — symbolizowany literą w (ang. *write*),
- wykonywalność — oznaczoną literą x (ang. *execute*).

Właściciel pliku określa, co może z nim zrobić właściciel grupowy, czyli osoby należące do grupy, która daje dostęp do pliku. Taki schemat pozwala na udostępnianie plików i katalogów np. członkom tego samego projektu.

Dostęp użytkownika root jest niezbędnym warunkiem administracji systemem oraz głównym elementem jego bezpieczeństwa. Zdalne logowanie na konto root powinno być domyślnie zabronione, ponieważ logowanie na to konto nie pozostawia żadnych zapisów operacji wykonywanych przez tego użytkownika. Zaleca się pozostawienie włączonych poleceń su (przełączenie się na konto root) oraz sudo. Polecenie su nie rejestruje poleceń wydanych jako użytkownik root, ale tworzy wpis dziennika, na podstawie którego można stwierdzić kto i kiedy pracował na jego koncie. Polecenie sudo jest programem, który powinien być zalecanym rozwiązaniem, ponieważ pozwala na podwyższenie uprawnień użytkownikom, którzy są dodani do pliku sudoers. Dzięki temu mogą oni wykonywać polecenia z uprawnieniami root. Polecenie sudo rejestruje informacje o wykonanych poleceniach, komputerach, na których zostały one wykonane, osobach, które je uruchomiły, katalogach, z których

zostały uruchomione oraz o czasach, w jakich zostały wywołane. Informacje te zazwyczaj zbierane są przez demon syslog, lecz mogą być zapisywane do innego, wybranego pliku. Warto przemyśleć konfigurację sudo dla całego ośrodka za pomocą dystrybucji pliku sudoers na wszystkie serwery. Warto wtedy zadbać o to, aby dystrybucja była przeprowadzona za pośrednictwem szerszego systemu zarządzania konfiguracją oraz śledzona za pomocą systemu monitorowania integralności plików. Warto też przemyśleć kwestię wyłączenia konta root i pozostanie tylko przy sudo. Rozwiązanie to niewątpliwie ma wiele zalet. Eliminuje możliwość naruszenia hasła użytkownika root a także brak zapisu informacji o tym, co zostało zmienione w systemie.

Model standardowy nie jest pozbawiony wad. Problemem jest to, że zapewnia zerową lub minimalną obsługę audytu i rejestrowania zdarzeń. W prosty sposób można sprawdzić, do jakiej grupy należy dany użytkownik, jednak niekoniecznie da się określić, jakie uprawnienia daje użytkownikom członkostwo w danej grupie. Oprócz tego nie ma żadnej realnej metody, aby śledzić, w jaki sposób są wykorzystywane rozszerzone uprawnienia, czy sprawdzić wykonywane w ten sposób operacje. Z tego powodu zalecane jest rozszerzenie standardowego modelu kontroli dostępu o rozwiązania, które wymienię poniżej.

- PAM (ang. *Pluggable Authentication Modules*) – interfejs obejmuje szereg bibliotek uwierzytelniania za pomocą określonych metod.
- Kerberos, czyli sieciowe uwierzytelnianie kryptograficzne w postaci zaufanego serwera wydającego poświadczenia kryptograficzne w celu uwierzytelnienia użytkowników oraz usług. Zaleca się stosowanie go razem z PAM.
- Listy Kontroli Dostępu (ang. ACL *Access Control List*).
- Docker – rozwiązanie do konteneryzacji oprogramowania, polegające na rozdzieleniu procesów na hierarchiczne przestrzenie nazw, co pozwala na zamknięcie procesu wewnątrz kontenera i odizolowanie go od innych procesów, co pozwala na uruchomienie go z uprawnieniami root bez obawy o to, że zagrozi to innym częściom systemu.
- MAC (ang. *Mandatory Access Control*) – obowiązkowa kontrola dostępu za pomocą zdefiniowanych reguł.

- Kontrola dostępu oparta na rolach określana w skrócie jako RBAC (ang. *Role Based Access Control*) za pomocą warstwy pośredniczącej pełniącej funkcję sprawdzania ról użytkowników.
- AppArmor
- SELinux

Te dwa ostatnie rozwiązania opiszę poniżej ze względu na to, że zostały zastosowane w mojej pracy, jako przykłady nowoczesnych mechanizmów kontroli dostępu.

1.2.4.1. AppArmor

AppArmor jest produktem firmy Canonical, Ltd., która wydaje dystrybucję Ubuntu. Jest wspierany w dystrybucjach Ubuntu oraz Debian, a także został przyjęty jako standard przez dystrybucję SUSE. Zaimplementowałem to rozwiązanie na serwerze web, na którym znajduje się Linux Debian 9.8.0 na zasadzie pewnego rodzaju MAC i jako uzupełnienie standardowego modelu kontroli dostępu. Mimo tego, że możliwa jest dowolna konfiguracja, to AppArmor nie został zaprojektowany jako system zorientowany na użytkownika. Głównym celem podsystemu AppArmor jest zabezpieczenie usług, poprzez ograniczenie szkód, które mogą wyrządzić programy, w sytuacji, gdy ich zabezpieczenia zostaną naruszone czy też będą działać nieprawidłowo. Chronione przez AppArmor programy nadal podlegają wszystkim ograniczeniom narzuconym przez model standardowy, jednakże dodatkowo jądro filtruje ich działania poprzez określony profil AppArmor, który domyślnie odrzuca wszystkie żądania, więc profil ten musi wyraźnie określać wszystko, co wolno danemu procesowi. Programy, które nie mają profili, takie jak powłoki użytkownika, nie posiadają żadnych ograniczeń, więc działają tak, jakby AppArmor nie był zainstalowany.

Profile AppArmor znajdują się w pliku /etc/apparmor.d i są klarowne nawet bez dokładnej znajomości systemu. Większość profili AppArmor korzysta z gotowych modułów, które wykonują określone działania przyznające dostęp. AppArmor odwołuje się do plików i programów przy pomocy ścieżek dostępu, dzięki czemu profile są czytelne i nie zależą od żadnej konkretnej implementacji systemu plików. Podejście to jednak jest pewnego rodzaju kompromisem. Przykładem może być tutaj fakt, że AppArmor nie rozpoznaje twardych dowiązań jako odwołań do tych samych jednostek, na które wskazują.

AppArmor jest efektywnym i łatwym w użyciu narzędziem, które aktywnie chroni system operacyjny poprzez monitorowanie i ograniczenie dostępu do plików, sieci i wykonywania zadań tzw. *linux capabilities (chown, setuid, itp.)*. Jednakże w żaden sposób nie zmienia on zasad DAC (ang. *Discretionary Access Control*), czyli metody zabezpieczania, która polega na przypisywaniu konkretnym podmiotom (właściciel, grupa, inni) konkretnych uprawnień. Dlatego, jeśli wszystkie pliki katalogu domowym danego użytkownika w systemie mają flagi 777, albo też używane jest konto root, AppArmor nie będzie spełniać swojej roli. Co prawda, AppArmor nie jest w stanie ochronić serwera przez wszystkimi możliwymi expolitami, to system jest bardziej bezpieczny, gdy AppArmor jest włączony, ponieważ do wielu usług (Apache, Postfix) istnieją gotowe profile, które można zaimplementować, co pozwala w prosty sposób zabezpieczyć serwer. Dodatkowo używając zaawansowanej analizy statystycznej i narzędzi opartych na uczeniu się, polityki AppArmor dla nawet bardzo złożonych aplikacji mogą zostać wdrożone w ciągu kilku godzin.

Podczas gdy jądro Linuksa zapewnia dobrą izolację użytkowników i silną kontrolę uprawnień do plików, MAC taki jak AppArmor zapewnia bardziej precyzyjne uprawnienia i ochronę przed wieloma nieznanymi zagrożeniami. Jeśli w jądrze Linuksa lub innym demonie systemowym zostanie znaleziony błąd bezpieczeństwa, dobrze skonfigurowany system AppArmor może uniemożliwić dostęp do ścieżek krytycznych, które mogą być podatne na ten problem.

AppArmor może działać w dwóch trybach:

- egzekwowania (ang. *enforce*),
- „narzekania” (ang. *complain*).

Egzekwowanie jest domyślnym statusem produkcyjnym AppArmor, natomiast „skarga” jest przydatna przy opracowywaniu zestawu reguł opartych na rzeczywistych wzorcach działania oraz przy naruszaniu zasad logowania. Jest on konfigurowany za pomocą plików tekstowych w stosunkowo przyjaznym formacie i ma krótszą krzywą uczenia się niż większość innych obowiązkowych systemów kontroli dostępu.

Ważną cechą AppArmor jest to, że profile AppArmor ograniczają procesy w oparciu o ścieżkę wykonywalną aplikacji, ponieważ jest to usługa MAC oparta na nazwach ścieżek. Tak więc AppArmor nie zapewnia żadnej ochrony przed nieuczciwym użytkownikiem lokalnym, który może skopiować plik wykonywalny do innej lokalizacji i uruchomić go pod inną ścieżką. Chroni to jednak przed scenariuszem, w którym zdalny atakujący może przejąć kontrolę nad aplikacją znajdującą się w sieci

i sprawić, że zrobi ona złe rzeczy dla systemu plików. Przykładowo, gdy złośliwa strona wykorzystuje jakiś błąd w Apache do przejęcia kontroli nad serwerem web, profil AppArmor może uniemożliwić Apache wykonanie akcji, która jest niebezpieczna.

W przypadku ssh, jeśli ssh zabrania dostępu użytkownikowi root, wtedy polityka AppArmor może uniemożliwić ssh nadanie uprawnień sysadmin, przyznając mu tylko domyślnie uprawnienia użytkownika i dostęp do powłoki. W ten sposób można zagwarantować, że atakujący nie uzyska dostępu root, a jedynie dostęp do kont typu non-root.

W sytuacji, gdy ustawiona zostanie ścieżka dla usługi w AppArmor, to usługa ta nadal podlega ograniczeniom systemu plików Linuksa (tzn. ustawień za pomocą chmod, chgrp i chown). AppArmor zapewni dodatkową warstwę ochrony, nawet jeśli mechanizmy te zostaną naruszone.

1.2.4.2. SELinux

SELinux to zestaw reguł, które są kompilowane w konfigurację załadowaną i zaimplementowaną w czasie pracy. To jedna z najstarszych implementacji mechanizmu MAC dla systemów Linux. Operacje na podmiotach są pośredniczone przez ten abstrakcyjny zbiór zasad opartych na etykietach dołączonych do tych podmiotów i użytkownika próbującego dokonać zmiany. Więc poza etapem kompilacji, nie różni się aż tak bardzo od uprawnień. Większość doświadczonych użytkowników Linuksa/Uniksa może stwierdzić, patrząc na uprawnienia ujawnione w 'ls -l' jak rozwiązać problem. Podczas gdy 'ls -Z' wyświetla etykiety SELinux na plikach i nie mówi to zbyt wiele o uprawnieniach, które one umożliwiają. W tym celu należy przyjrzeć się polityce.

Docelowa polityka SELinuksa z Fedory jest obecnie dystrybuowana jako 1271 plików zawierających 118815 linii konfiguracyjnych. Zalecaną praktyką nie jest zmiana tych plików, ale raczej dodanie większej ilości konfiguracji w celu zmiany zachowania SELinux. Jedną z konsekwencji jest to, że aktualizacje do konfiguracji, która przynajmniej nie zniszczy dodatkowej konfiguracji, która została dodana, często wymagają zmiany etykiet w systemie plików; prosta aktualizacja może niespodziewanie przekształcić się z krótkotrwałego przestoju w godziny lub dni, podczas gdy każdy plik na dyskach serwera jest ponownie etykietowany.

SELinux zmienia istniejące modele zabezpieczeń. Ukierunkowana polityka nie tylko unieważnia uprawnienia systemu plików, ale także mechanizmy kontroli dostępu

wbudowane w programy, na przykład 'at' nie jest w stanie odczytać z /etc/at.allow running as a system_u user. Z bitem setuid ustawionym na wykonywalny, można uruchomić go jako inny użytkownik, ale zachowuje oryginalny kontekst SELinux.

"Domyślnie użytkownicy Linuksa w domenach guest_t i xguest_t nie mogą wykonywać aplikacji w swoich katalogach domowych lub katalogu /tmp/, uniemożliwiając im wykonywanie aplikacji, które dziedziczą uprawnienia użytkowników, w katalogach, do których mają dostęp przy zapisywaniu. Pomaga to zapobiec modyfikowaniu plików użytkowników przez uszkodzone lub złośliwe aplikacje."[1]

W SELinux przyjęto podejście maksymalistyczne, czyli zaimplementowano każdą możliwą odmianę MAC i RBAC. Został on zaadaptowany jako domyślne rozwiązanie instalowane w dystrybucjach, których użyłem, czyli CentOS 7.6 i Fedora 29. Konfiguracja i obsługa SELinux są dość skomplikowane, jednakże pomimo tego faktu, jego wykorzystanie w systemach systematycznie rośnie, zwłaszcza w środowiskach o większych wymaganiach dotyczących bezpieczeństwa danych. Należy zaznaczyć, że SELinux jest także standardowym elementem systemu Android. Pomimo tego, że definiowanie polityki w przypadku SELinux jest szczególnie skomplikowane, ponieważ w celu zabezpieczenia określonego demona sieciowego polityka powinna szczegółowo wymieniać wszystkie obiekty, do których ma on dostęp, coraz częściej można znaleźć w internecie wiele uogólnionych konfiguracji, a większość dystrybucji wyposażonych w SELinux posiada rozsądne konfiguracje domyślne, które można łatwo zainstalować i dostosować do potrzeb danej instalacji. Na stronie *seedit.sourceforge.net* znajduje się rozbudowany edytor konfiguracji, który znacznie upraszcza definiowanie polityki SELinux. Jednak zalecam ostrożność, ponieważ na stronie widnieje informacja: „*Ten projekt nie jest obecnie utrzymywany. Ta strona i kod istnieją tylko w celach informacyjnych.*" Na szczęście nie brakuje dokumentacji i jedną z lepszych jest ta znajdująca się na stronie Red Hat[2].

Główna konfiguracja SELinux znajduje się w pliku /etc/selinux/config. Dwa wiersze są szczególnie istotne, a mianowicie:

```
SELINUX=enforcing
SELINUXTYPE=targeted
```

[1] *https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/selinux_users_and_administrators_guide/sect-security-enhanced_linux-targeted_policy-confined_and_unconfined_users*
[2] *https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/5/html/deployment_guide/rhlcommon-chapter-0017*

Pierwszy parametr przyjmuje jedną z trzech wartości: enforcing, permissive lub disabled. Opcja enforcing wymusza załadowanie polityki i blokuje próby nieautoryzowanego dostępu. Permissive pozwala na naruszanie polityki, lecz zapisuje próby tego typu za pośrednictwem syslog, co jest opcją przydatną szczególnie, gdy testowana jest konfiguracja serwera i rozwijana polityka SELinux. Ostatni parametr wyłącza SELinux. Parametr targeted jest domyślną polityką w dystrybucjach z rodziny Red Hat. Definiuje on dodatkowe zabezpieczenie dla kilku demonów, które są wyraźnie chronione, lecz nie ingeruje to w pozostałą część systemu.

Rozwijanie własnych polityk można wykonać za pomocą narzędzia audit2allow, które tworzy definicje polityk na podstawie dzienników naruszeń. Problem w tym przypadku polega na tym, że zezwala się na pobłażliwą ochronę systemu tak, by jej naruszenia zostały zarejestrowane, ale nie egzekwowane, a następnie testuje się system i buduje politykę zezwalającą na działania w oparciu o to, co podsystem rzeczywiście zrobił. Niestety, tego rodzaju podejście nie jest w stanie zagwarantować pełnego objęcia wszystkich ścieżek kodu, więc wygenerowane automatycznie profile nie będą idealne.

1.2.5. Kontrola antywirusowa

W mojej pracy postanowiłem omówić aplikację ClamAV i wyjaśnić, w jaki sposób można ją zintegrować z agentem Postfix w formie tzw. miltera, czyli filtra poczty (ang. *mail filter*). Program ten stanowi otwarty silnik antywirusowy, który w dużej mierze przypomina rozwiązania takich firm jak Symantec czy McAfee. Różnica jednak polega na tym, że w przeciwieństwie do produktów wspomnianych firm oprogramowanie i aktualizacje sygnatur w programie ClamAV są bezpłatne. Program ClamAV wykorzystuje specjalne reguły zwane sygnaturami do skanowania poczty pod kątem danych przypominających kod wirusa. Każda sygnatura zawiera informacje opisujące danego wirusa, np. ciąg znaków charakterystyczny dla tego wirusa, który pozwala na jego wykrycie.

1.2.5.1. ClamAV

ClamAV to zestaw narzędzi antywirusowych, które są przeznaczone dla systemów uniksowych. Kluczową rolę odgrywa tutaj potrzeba ciągłej aktualizacji zarówno kodu jak i baz sygnatur wirusów. Program ten jest rozwijany przez międzynarodową grupę kilkunastu developerów i dostępny na licencji GNU GPL v2. ClamAV stosuje zasadę reguły KISS (ang. *Keep it Simple, Single*). Dlatego też

w wachlarzu programów dostarczanych w pakiecie znajdują się proste w użyciu wyspecjalizowane programy, które pozwalają zbudować złożone systemy skanujące.

Clamscan jest prostym narzędziem umożliwiającym skanowanie plików oraz katalogów bezpośrednio z linii poleceń. Wystarczy wskazać poleceniem nazwę pliku lub katalogu, który powinien zostać przeskanowany.

Clamd to usługa, która została zaprojektowana do zadań, w których istotna jest wydajność. Program posiada budowę wielowątkową, dzięki czemu jest w stanie wykorzystać dodatkowe procesory w systemach SMP. Umożliwia ona także efektywne wykorzystanie baz sygnatur, które są ładowane podczas startu oraz po wykryciu nowej wersji, a następnie współdzielone przez wszystkie wątki.

Freshclam to program automatyzujący proces aktualizacji baz sygnatur wirusów. Posiada on szereg opcji, które pozwalają na uruchomienie go na żądanie, w tle jako usługę. Potrafi on współpracować z serwerami proxy, umożliwia uruchomienie zewnętrznych programów w zależności od zdarzenia (aktualizacja bazy sygnatur, błąd, wykrycie nowej wersji oprogramowania).

1.2.6. Monitorowanie dzienników systemowych

Jednym z zadań administratora jest przeglądanie plików dzienników systemowych. Zawierają one ważne podpowiedzi, które mogą wskazać drogę do rozwiązania rozmaitych problemów związanych z konfiguracją. Jeśli jakaś usługa nie chce się uruchomić lub przy uruchomieniu systemu ciągle pojawia się ten sam błąd warto przejrzeć co zapisuje się w dziennikach systemowych. W Linux dąży się do rejestrowania informacji w jednym miejscu. Ogólnie syslog jest usługą systemową odpowiadającą za logowanie wszystkich zdarzeń jakie miały miejsce w systemie operacyjnym. Po zainstalowaniu systemu operacyjnego syslog, który jest konfigurowany w pliku konfiguracyjnym /etc/rsyslog.conf. jest od razu aktywny i można to stwierdzić poleceniem :

```
$ sudo chkconfig rsyslog –list
```

1.2.6.1. Dzienniki zdarzeń

W sytuacji, gdy coś nieprzewidzianego dzieje się z aplikacją lub jądrem, zawsze warto zajrzeć do dziennika systemowego lub dziennika jądra. Administrator zazwyczaj zobligowany jest tzw. umową o gwarantowanym poziomie świadczenia usług SLA (ang. *Service Level Agreement*), aby jak najszybciej ponownie uruchomić zawieszoną

usługę, jednak zajrzenie do dzienników zdarzeń w poszukiwaniu przyczyny problemu pomaga zapobiec powtórzeniu się nieprzyjemnej sytuacji. Jeżeli sam w sobie demon dziennika zdarzeń nie odpowiada, administrator jest ciągle w stanie sprawdzić bufor dziennika za pomocą polecenia dmesg.

Linux wykonuje wiele procesów, które wykorzystują systemowe zasoby. Procesy tworzone są przez system i serwery usług. Informacje o ich działaniu, przekazywane przez komunikaty, zapisują się w dziennikach zdarzeń, które administrator powinien analizować.

Dzienniki zdarzeń są niezbędne w pracy każdego administratora. Ilość oraz różnorodność logowanych informacji jest ogromna. W systemie Linux zdarzenia są przetrzymywane w plikach tekstowych, co dodatkowo utrudnia ich analizę. Istnieją jednak narzędzia, które ułatwiają przeglądanie dzienników, np. GoAccess, który umożliwia analizę logów serwera Apache.

Dzienniki zdarzeń w systemie Linux to zarejestrowane zdarzenia systemu operacyjnego, aplikacji i usług, które są uporządkowane zgodnie z linią czasu. Przechowywane są w postaci plików tekstowych, zazwyczaj w katalogu /var/log. Folder ten zawiera znaczącą liczbę plików z informacjami o każdej aplikacji oraz katalogi dla szczegółowych dzienników zdarzeń wybranych aplikacji. Najłatwiej odczytać plik z dziennikami zdarzeń, wykonując jedno z poleceń (w zależności czy plik jest skompresowany lub nie):

```
# cat /var/log/nazwa_pliku_dziennika
# zcat /var/log/nazwa_pliku_dziennika.gz
```

Jeżeli chcielibyśmy przeglądać dzienniki zdarzeń w trybie rzeczywistym, należy wykonać polecenie tail -f /var/log/nazwa_pliku_dziennika. Natomiast wyszukiwanie w plikach zrealizujemy komendą:

```
# cat /var/log/nazwa_pliku_dziennika | grep wyszukiwana_fraza.
```

Logi można też analizować w czasie rzeczywistym za pomocą wyrażenia:

```
# tail -f /var/log/nazwa_pliku_\dziennika|grep wyszukiwana_fraza
```

Każdy dziennik zdarzeń zawiera datę, nazwę hosta, nazwę aplikacji lub usługi, wiadomość lub komunikat. Dzienniki zdarzeń podlegają rotacji zgodnie z ustawieniami w pliku konfiguracyjnym /etc/logrotate.conf. Definiujemy w nim, z jaką częstotliwością będą tworzone pliki z dziennikami zdarzeń oraz jak długo będą przechowywane. Pojemność dyskowa systemów z usługami nie pozwala przeważnie na długie przechowywanie dzienników zdarzeń, a przeszukiwanie takich plików nie jest ani

szybkie, ani elastyczne. Tu z pomocą przychodzą specjalne analizatory takie jak np.: LogAnalyzer – analizator dzienników dostępny przez przeglądarkę web, GoAccess - darmowy analizator dzienników zdarzeń serwera WWW, do przeglądania dzienników w terminalu: multitail, jako rozszerzenie tail, swatch, lnav, a także inne jak logsentry, epylog, fwlogwatch, czy logcheck, które po odpowiedniej konfiguracji wysyłają informacje na wskazany adres e-mail.

Dzienniki zdarzeń zazwyczaj są podzielone na wiele plików umieszczonych w katalogu /var/log. Tworzeniem wpisów dziennika zajmują się najczęściej dwa demony: journald i rsyslogd. Opiszę kolejno każdy z nich.

Demon journald

Wraz z usługą systemd wprowadzono nowy mechanizm tworzenia wpisów dziennika. Demon journald tworzy binarne pliki dzienników i może być używany wraz z tradycyjną usługą (r)syslog (lub ją zastępować).

Dzienniki tworzone przez demona journald cechują się poniższymi funkcjami:

- dzienniki są indeksowane, co przyśpiesza proces wyszukiwania danych;
- wykrywane jest manipulowanie wpisami dziennika i nie można w łatwy sposób ręcznie ich modyfikować;
- wpisy dziennika mają wyraźnie określony format, a poszczególne pola są dobrze zdefiniowane;
- usługa journald gromadzi dodatkowe metadane dzienników dla każdego wpisu;
- demon journald obsługuje formaty eksportowe (np. JSON).

Demon rsyslogd

Aplikacje przesyłają dane do usługi rsyslog w specjalnym formacie, który może być przetwarzany przez tego demona. Usługa ta pobiera następnie wpisy dzienników i może wykonać na nich różne operacje, jak choćby zapisać je w pliku.

1.2.6.2. Analizowanie i korelowanie dzienników

Skoro system Linux daje nam do dyspozycji pliki dzienników to, co można z nimi właściwie zrobić? Dzienniki zdarzeń mają dwa główne zastosowania:

- określanie czasu wystąpienia nieprawidłowości w działaniu systemu,
- pomoc w określeniu przyczyny problemu z działaniem systemu.

Aby móc spełnić pierwsze zadanie, potrzebne jest narzędzie, które identyfikuje określone wpisy dziennika oraz poinformuje administratora o ich obecności. Proces ten

nazywa się analizowaniem i korelowaniem dzienników zdarzeń. W ten sposób można w znaczący sposób usprawnić proces analizowania i korelowania danych w codziennej pracy administratora.

To, o czym należy pamiętać, to fakt, że analiza i korelacja są dwiema zupełnie odmiennymi czynnościami. Analizą nazywa się badanie elementów składowych oraz powiązań między nimi tworzących całość układu. Administratorzy uczą się wzorców działania poszczególnych serwerów i często są w stanie wykryć problem znacznie szybciej, niż zautomatyzowany system monitorujący. Jednak model ten ma dwie istotne wady. Pierwsza z nich polega na tym, że administrator nie jest w stanie być w dwóch miejscach jednocześnie. Natomiast drugą wadą jest to, że rosną rozmiary gromadzonych danych i stają się one w pewnym momencie tak duże, że pojedyncza osoba nie jest w stanie ich zanalizować.

W tym momencie pojawia się korelacja. Najprościej można zdefiniować ją jako wykrywanie związków pomiędzy danymi. Administrator konfiguruje narzędzia, które gromadzą dane, program je porządkuje i wyświetla tylko te, które są niezbędne, a następnie następuje korelacja pozostawionych danych tak, aby można je było zaprezentować w logiczny i uporządkowany sposób, co pozwala administratorowi przeprowadzić dokładną analizę.

Dobrze skonfigurowane i zarządzane programy są w stanie w sposób ciągły sortować strumień danych generowany przez codzienne operacje serwerów. Co więcej, są one w stanie wykrywać powiązania pomiędzy poszczególnymi typami danych i albo składają je w jedną całość, albo przekazują poszczególne elementy do analizy. Dość ważną kwestią jest dobór właściwych narzędzi, które wyszukają właściwe dane, co pozwoli poinformować administratora o jakichś nieprawidłowościach, na które należy zwrócić uwagę.

Pierwszy etap w tworzeniu takiego zautomatyzowanego systemu monitorującego powinien polegać na umieszczeniu zebranych danych w odpowiednim miejscu. Należy stworzyć listę wszystkich aplikacji i serwerów wraz z rozpiską, gdzie są zlokalizowane ich pliki dzienników. W następnym kroku należy zebrać te informacje i zdefiniować, o czym ma być informowany administrator. Należy stworzyć listę krytycznych komunikatów, które są istotne dla prawidłowej pracy serwerów. Listy powinny zostać pogrupowane pod względem priorytetów w taki sposób, aby część informacji była wysyłana na adres e-mail, niektóre z nich były wyświetlane na ekranie,

a jeszcze inne uruchamiały zautomatyzowane procesy lub próbowały samodzielnie rozwiązać problem np. poprzez ponowne uruchomienie usługi.

Trzeci etap polega na zaimplementowaniu procesu analizowania i korelowania dzienników, a także na skonfigurowaniu narzędzi korelacji i zaprojektowaniu oczekiwanych reakcji. Należy ze szczególną uwagą dokumentować każdą wiadomość, reakcję na komunikat oraz wszelkie dodatkowe informacje, które są powiązane z komunikatem.

1.3. Bezpieczeństwo sieciowe

System SSH, to protokół, który służy do zdalnego logowania i zabezpieczenia usług sieciowych w niezabezpieczonej sieci. Możliwości tego protokołu obejmują zdalne wykonywanie poleceń, dostęp do powłoki, transfer plików, przekierowywanie portów, usługi pośredniczenia w ruchu sieciowym oraz tunelowanie VPN. Jest to protokół klient-serwer, który wykorzystuje kryptografię w celu uwierzytelniania oraz zapewnienia poufności i integralności komunikacji pomiędzy dwoma hostami. „*Został zaprojektowany z myślą o elastyczności algorytmicznej, co umożliwia aktualizowanie i wycofywanie wykorzystywanych przez niego protokołów kryptograficznych w miarę rozwoju branży.*”[3]

Oprócz SSH mechanizmy bezpieczeństwa powinny być implementowane na poziomie sieci. Podstawowym narzędziem bezpieczeństwa sieci jest zapora sieciowa (ang. *firewall*), czyli urządzenie, lub oprogramowanie blokujące dostęp niepożądanych pakietów do sieci i systemów. W tym rozdziale omówię OpenSSH, implementację SSH na licencji open source, a także podstawy konfiguracji i używania zapory sieciowej oraz generowania kluczy RSA w celu zabezpieczenia zdalnego logowania.

1.3.1. Zabezpieczenie logowania sieciowego

Zabezpieczenie logowania sieciowego polega na zainstalowaniu i poprawnym skonfigurowaniu zapory filtrującej pakiety, która ogranicza ilość i rodzaj ruchu sieciowego, jaki może wejść do sieci przez bramę (ang. *gateway*) z internetu (lub też przez wewnętrzną bramę pomiędzy niezależnymi sieciami w ramach organizacji) w oparciu o nagłówek pakietu. Pakiet posiada adres źródłowy i docelowy, numery

[3] Evi Nemeth, Garth Snyder, Trent R. Hein, Ben Whaley, Dan Mackin, James Garnett, Fabrizio Branca, Adrian Mouat, Unix i Linux Przewodnik Administratora Systemów, Wydanie V, Gliwice 2018, HELION, s. 1046

portów oraz typ protokołu, który brama przepuszcza lub odrzuca, a w niektórych przypadkach zapisuje w dzienniku pakiety, które nie spełniają określonego zestawu reguł. Programy filtrujące pakiety są dostępne dla dystrybucji Linux w postaci różnych programów. Najczęściej spotyka się iptables, który jest dostępny praktycznie prawie w każdej dystrybucji. W dystrybucjach CentOS, Fedora, Red Hat zainstalowany jest domyślnie firewalld. Natomiast w dystrybucjach z rodziny Debian można zainstalować ufw (ang. *uncomplicated firewall*), który jest tak naprawdę nakładką na iptables, lecz pozwala zaoszczędzić czas, skracając do minimum tworzenie reguł, które w iptables są dość długie.

Większość zapór sieciowych wykorzystuje porty, które są zdefiniowane w pliku /etc/services (lub odpowiedniku, zależnie od dystrybucji systemu). Demony odpowiadające za te usługi przyłączają się do odpowiednich portów i oczekują na połączenia od zdalnych hostów. Większość portów, na których „nasłuchują połączeń" popularne usługi to tzw. porty uprzywilejowane, co oznacza, że mają przypisane numery z zakresu od 1 do 1023. Porty te mogą być używane wyłącznie przez konto root lub przez odpowiednie możliwości systemu Linux. Porty od 1024 wzwyż mogą być używane przez dowolne usługi i nazywane są portami nieuprzywilejowanymi. Filtrowanie pakietów specyficzne dla usług oparte jest na założeniu, że ruch wychodzący z serwera powinien być akceptowany niezależnie od portu i hosta docelowego, natomiast ruch przychodzący domyślnie jest blokowany dla wszystkich i zezwala się na ruch przychodzący tylko na określonych portach i wyłącznie dla określonych docelowych adresów IP.

Zapora sieciowa nie powinna być jedynym rodzajem zabezpieczenia przed zagrożeniami z sieci. Jest to tylko jeden z elementów wielopłaszczyznowej, dopracowanej strategii bezpieczeństwa. Administrator nie powinien lekceważyć innych narzędzi bezpieczeństwa. Jeśli tak postąpi, wpływ zapory sieciowej na bezpieczeństwo sieci może być wręcz negatywny. Każdy serwer w sieci firmowej, czy korporacyjnej powinien być odpowiednio zabezpieczony, zaktualizowany, wzmocniony i monitorowany za pomocą odpowiednich narzędzi takich jak: Nessus, Nmap, Snort, Bro, czy OSSEC. W celu zmaksymalizowania zabezpieczenia logowania sieciowego już na etapie projektowania bezpieczeństwa sieci administrator powinien skupić się na wygodzie i dostępności narzędzi. Ostatecznie to czujność i skuteczność administratora decyduje o bezpieczeństwie sieci, a nie profesjonalny wygląd sprzętowej zapory sieciowej.

1.3.1.1. Zdalny dostęp

W wielu przypadkach można zalogować się zdalnie do Linuksa. Jest to szczególnie istotne, gdy mamy do dyspozycji dystrybucje Linuksa, które pełnią funkcje serwerowe w centrach danych lub odległych regionach geograficznych, ewentualnie są schowane w szafie serwerowej lub innym w racku. Często takie serwery nie mają nawet podłączonych ekranów ani klawiatur, a dostęp do nich można uzyskać jedynie z poziomu sieci. W Linux bardzo prosto można nawiązać zdalne połączenie z serwerami, co pozwala administratorowi na administrowanie i zarządzanie nimi. Istnieje wiele różnych sposobów uzyskiwania zdalnego dostępu, wśród których można wymienić protokół współdzielenia pulpitu VNC (ang. *Virtual Network Computing* – przetwarzanie wirtualnej sieci), często używany do łączenia z komputerami działającymi w systemie Windows protokół RDP (ang. *Remote Desktop Protocol* – protokół zdalnego pulpitu) czy też bardzo popularny protokół SSH (ang. *Secure Shell* – bezpieczna powłoka). Na tym ostatnim protokole skupię moją uwagę, ponieważ jest on najczęściej wykorzystywany do łączenia się zdalnego z serwerami.

1.3.1.2. Korzystanie z protokołu SSH

SSH jest jednocześnie aplikacją i protokołem zabezpieczeń używanym na wiele sposobów, jednakże jego głównym przeznaczeniem jest zdalna administracja serwerami. Połączenie SSH następuję poprzez sieci TCP/IP (ang. *Transmission Control Protocol/Internet Protocol* – protokół kontroli transmisji/protokół internetowy) w modelu klient-serwer. Komputer, z którego łączy się administrator, pełni rolę klienta. Natomiast komputer, z którym następuje połączenie pełni rolę serwera, gdyż akceptuje przychodzące połączenie i zarządza nim.

Połączenia za pomocą protokołu SSH są szyfrowane i wymagają metody uwierzytelniania – hasła, lub klucza publicznego. By móc utworzyć zdalne połączenie, administrator musi znać adres IP lub nazwę serwera, do którego chce uzyskać dostęp. Następnie administrator inicjuje połączenie po stronie klienta, który komunikuje się z serwerem poprzez port 22 przy użyciu protokołu TCP/IP (można zmienić numer portu w pliku /etc/ssh/sshd_config, co jest zalecanym rozwiązaniem). Po nawiązaniu połączenia początkowego serwer wysyła prośbę o podanie nazwy użytkownika i poświadczeń bezpieczeństwa. Gdy wprowadzone dane są prawidłowe, klient nawiązuje zdalne połączenie z serwerem.

1.3.1.3. Konfigurowanie bezpiecznego połączenia SSH

Aby móc nawiązać połączenie z serwerem poprzez SSH wystarczy użyć terminala (w przypadku systemów Unix/Linux) lub odpowiedniego klienta (np. Putty w środowisku Microsoft Windows). Większość dystrybucji bazujących na Linuksie lub Uniksie zawiera wbudowany protokół SSH oraz dostępną komendę ssh. Aby z niej skorzystać należy wpisać nazwę użytkownika oraz adres IP serwera lub jego nazwę (przedzielone znakiem @), z którym administrator zamierza nawiązać połączenie, co zostało zaprezentowane na listingu 1.

Listing 1 *Połączenie SSH*

```
$ ssh deb_usr@sysadmin.info.pl lub ssh deb_usr@176.105.137.72
Password:
```

W listingu 1 następuje połączenie z serwerem sysadmin.info.pl jako użytkownik deb_usr. Serwer prosi o podanie hasła. Po wprowadzeniu właściwego hasła zostaniemy zalogowani do wiersza poleceń serwera.

Podczas pierwszego połączenia z danym serwerem istnieje konieczność akceptacji klucza RSA, który jest wygenerowany przez serwer SSH, z którym następuje połączenie. Po jego zaakceptowaniu jest on zapisywany wraz z nazwą serwera w pliku known_hosts. Podczas każdego kolejnego połączenia z tym serwerem protokół SSH sprawdza przydzielony klucz pod kątem wszelkich modyfikacji. Jeśli wykryje on jakąkolwiek zmianę, poprosi o oczyszczenie klucza przed nawiązaniem połączenia i odrzuci połączenie. W ten sposób administrator może sprawdzić, czy ktoś próbował ingerować w zdalne połączenie.

1.3.1.4. Konfiguracja i wykorzystanie kluczy RSA do połączenia SSH

Wcześniej opisałem użycie protokołu SSH jedynie do łączenia się z serwerem z poziomu klienta w celu wprowadzania zmian w konfiguracji serwera lub instalacji oprogramowania. W tym podrozdziale zademonstruję pełny potencjał powłoki SSH. Wprowadzę uwierzytelnianie za pomocą kluczy.

Gdy jako administrator łączę się zdalnie do serwera przez protokół SSH, zostaję za każdym razem poproszony o podanie hasła. Nie ma problemu, jeśli robię to raz dziennie, jednak jeśli często łączę się w ten sposób z serwerami, okazuje się, że jest to dość czasochłonny proces, zwłaszcza gdy korzystam z długich, skomplikowanych haseł.

Aby ułatwić sobie codzienną pracę, warto, skorzystać z uwierzytelniania za pomocą klucza, które umożliwia protokół SSH. Wystarczy wygenerować parę kluczy – prywatny i publiczny, a następnie skopiować klucz publiczny na serwery, z którymi potem będzie można się łączyć przy jego pomocy. Po nawiązaniu komunikacji zdalny serwer dokona sprawdzenia, czy używany przeze mnie klucz prywatny stanowi składową wspólnego klucza publicznego. Jeśli tak jest, proces uwierzytelniania zostaje zakończony powodzeniem. Warto zwrócić uwagę na fakt, że klucze prywatne i publiczne służą do uwierzytelniania administratora, natomiast szyfrowanie samego połączenia jest gwarantowane przez klucze serwera SSH generowane w trakcie instalowania usługi.

Bardzo ważną kwestią jest dołączenie hasła do klucza prywatnego, ponieważ bez niego każda osoba, która uzyska dostęp do tego klucza, będzie mogła się zalogować (bez konieczności podawania hasła) do każdego serwera przechowującego powiązany klucz publiczny.

Po otrzymaniu pary kluczy możliwe jest przesłanie jej składowej publicznej do zdalnego komputera. Aby administrator mógł korzystać z klucza publicznego, musi go przechowywać w pliku authorized_keys wewnątrz katalogu `.ssh` umieszczonego w folderze domowym. Może samodzielnie przepisać wartość tego klucza albo wykorzystać w tym celu narzędzie ssh-copy-id.

1.4. Bezpieczeństwo serwera web

1.4.1. Apache

Istnieją różne sposoby konfiguracji serwera Apache. Jeszcze do niedawna wykorzystywano protokół HTTP, który wciąż jest jeszcze używany, lecz chcąc sprostać nowoczesnym standardom i jednocześnie zadbać o bezpieczeństwo jako administrator powinniśmy zadbać o obsługę protokołu HTTPS (który chroni komunikację http za pośrednictwem protokołu TLS, wcześniej znanego jako SSL – ang. *Secure Socket* Layer) na swoim serwerze. Włączenie obsługi HTTPS w witrynach internetowych z wielu istotnych powodów jest ważnym elementem ich hartowania. Pierwszy i najważniejszy powód, to szyfrowanie ruchu pomiędzy klientem a serwerem. Jest to ważne o tyle, że jeśli używane jest podstawowe uwierzytelnianie HTTP np. podczas logowania do witryny internetowej banku, lub witryny opartej o wybrany system CMS, to uniemożliwiamy napastnikowi podsłuchanie komunikacji i zobaczenie hasła. Drugim

powodem możliwe, że nawet ważniejszym jest to, że dzięki stosowaniu protokołu HTTPS użytkownicy strony mogą mieć pewność, że odwiedzają autentyczny serwer, a nie „podrobioną" witrynę internetową, która przypomina wyglądem oryginalny serwis oraz, że cała komunikacja pomiędzy ich przeglądarkami WWW a serwerem jest zabezpieczona przed atakami Man in the Middle. W tego rodzaju ataku napastnik udaje, że jest serwerem, przechwytuje zaszyfrowany ruch, odszyfrowuje go, a następnie ponownie go szyfruje i wysyła do rzeczywistego serwera. Dodatkowo TLS można wykorzystać jako mechanizm uwierzytelniania pomiędzy serwerem i klientem.

Aby taka konfiguracja TLS mogła działać, trzeba uzyskać ważny certyfikat TLS z urzędu certyfikatów (CA). Jest wiele urzędów i firm CA, w których można zakupić certyfikat. W certyfikat można zaopatrzyć się też u rejestratora, u którego została zakupiona nazwa domeny, lub skorzystać z bezpłatnych usług. Wybrałem ostatnią metodę i zaopatrzyłem się w certyfikat od Let's Encrypt, który dostaje się na 90 dni z możliwością przedłużenia.

Jednym z dobrych podręczników, które opisują konfigurację TLS dla różnych usług, w tym Apache, jest strona wiki *Server Side TLS* firmy Mozilla, dostępna pod adresem https://wiki.mozilla.org/Security/Server_Side_TLS. Znajdują się na niej interaktywne sekcje, które pozwalają zbudować przykładowe bezpieczne konfiguracje serwisów WWW na podstawie kilku profili.

- Modern: używa tylko nowoczesnych i bezpiecznych zestawów szyfrów i ustawień, co oznacza, że witryna może być niedostępna w wielu starszych wersjach przeglądarek.
- Old: prawidłowa konfiguracja TLS z wykorzystaniem starszych zestawów szyfrów, co zapewnia maksymalną zgodność wstecz z przeglądarkami.
- Intermediate: dobre połączenie nowoczesnych szyfrów dostępnych w profilu Modern z lepszą zgodnością z przeglądarkami, włącznie z niektórymi starszymi, ale wciąż popularnymi.

W prostej konfiguracji HTTPS wykorzystywane są ustawienia domyślne zestawu szyfrów dla serwera. Zatem wszystko, co jest potrzebne, to lokalizacja certyfikatu i klucza prywatnego witryny. Oczywiście trzeba pamiętać o tym, że certyfikat powinien być publiczny, natomiast klucz prywatny powinien być niejawny i dostępny do odczytu i zapisu wyłącznie dla użytkownika root (chmod 600). Konfiguracja obsługi protokołu HTTPS oraz wymuszanie przekierowania ruchu z HTTP na HTTPS zostały opisane w części praktycznej.

Istnieje sposób, aby pokonać zabezpieczenia witryny za pomocą protokołu HTTPS, który polega na przeprowadzeniu ataku degradacji protokołu. Polega to na tym, że napastnik zajmuje miejsce pomiędzy serwerem WWW a klientem i informuje klienta, że protokół HTTPS nie jest dostępny. W takim przypadku klient prawdopodobnie skorzysta z wersji HTTP witryny, dzięki czemu napastnik będzie mógł przechwycić niezaszyfrowany ruch pomiędzy klientem a serwerem. Nawet, gdy administrator serwera zadba o konfigurację przekierowania 302 z HTTP na HTTPS, to napastnik może to w łatwy sposób obejść. Problem rozwiązuje zastosowanie protokołu HSTS (ang. *HTTP Strict Transport Security*). Dzięki niemu serwer Apache może przesłać do klientów specjalny nagłówek, który informuje ich, że w komunikacji z serwerem powinni korzystać wyłącznie z protokołu HTTPS. Jeśli administrator serwera Apache zastosuje protokół HSTS, a napastnik podejmie próbę przeprowadzenia ataku degradacji protokołu, to przeglądarka będzie miała zbuforowany nagłówek z poprzedniej wizyty w witrynie i wyśle do klienta komunikat o błędzie. Chociaż stosowanie HSTS wydaje się skomplikowane, dodanie jego obsługi do witryny HTTPS jest bardzo proste. Wystarczy dodać jeden wiersz na końcu konfiguracji HTTPS. W przypadku serwera Apache ten wiersz wygląda następująco:

```
Header always set Strict-Transport-Security "max-age=15768000"
```

Maksymalny czas (w sekundach) przechowywania nagłówka w pamięci podręcznej w tym przykładzie ustawiony jest na wartość 15 768 000, czyli sześć miesięcy. Należy jeszcze włączyć opcję obsługi nagłówków przy pomocy polecenia `a2enmod headers` i zrestartować serwer Apache.

W instrukcji wdrażania opisana jest też instalacja WAF (ang. *Web Application Firewall*). Zapora WAF analogicznie jak zapora tradycyjna w tym sensie, że jest zdolna do przechwytywania i blokowania ruchu na podstawie reguł, jednak w przeciwieństwie do zapór tradycyjnych, które biorą pod uwagę tylko źródłowe i docelowe adresy IP oraz porty, WAF sprawdza zawartość żądań webowych docierających do serwera WWW i może blokować potencjalnie niebezpieczne żądania webowe, zanim dotrą do serwera WWW i zostaną przez niego przetworzone. W części praktycznej przedstawiłem, w jaki sposób należy zainstalować i skonfigurować ModSecurity – najpopularniejszy moduł WAF dostępny dla serwerów Apache i Nginx. Mimo tego, że ModSecurity działa zarówno z Apache, jak i Nginx, to pierwotnie został zaprojektowany dla serwera Apache i jest stosowany z nim o wiele dłużej. W związku z tym znacznie łatwiej uruchomić ModSecurity z Apache niż Nginx. Ponieważ Nginx

jeszcze nie obsługuje ładowanych modułów, dołączenie modułu ModSecurity wymaga kompilacji głównej aplikacji Nginx, w związku z tym wybrałem instalację Mod Security na serwerze Apache. *„Serwer Apache obecnie zawiera do wyboru trzy moduły przetwarzania wieloprocesorowego (ang. multi-processing module — MDM): przedwstępny (ang. prefork), roboczy (ang. worker) i zdarzeniowy (ang. event).*”[4]

1.4.2. PHP

Wiele aplikacji internetowych jest napisanych w PHP – języku skryptowym, które stanowią trzon różnorodnych witryn sieciowych na całym świecie, w tym popularnego systemu zarządzania treścią (ang. *Content Management System* – CMS) WordPress. Ponad 60% stron zbudowanych na bazie CMS używa WordPress.[5] Z tego względu zdecydowałem się na wdrożenie WordPress na serwerze web.

Po wysłaniu żądania strony, która jest napisana w języku PHP serwer Apache przetwarza tworzący ją kod i wyświetla wynik w przeglądarce. Aby móc samodzielnie przechowywać tego typu aplikacje na serwerze, w/w serwer musi być w stanie rozumieć i wykonywać kod PHP. Instrukcje zapisane w języku PHP powinny być wykonywane w poniższy sposób. Trzeba dodać do silnika Apache obsługę środowiska PHP i włączyć przedwstępny moduł MPM (mpm_prefork), jak już napisałem wyżej ze względów bezpieczeństwa, a ponieważ środowisko PHP nie jest wątkowo bezpieczne, wykorzystam menedżer procesów FastGGI do powiązania serwera Apache ze środowiskiem PHP. Używanie FastCGI jest w tej chwili jedynym sposobem na posiadanie bezpiecznego serwera WWW. Korzystanie z mod_php z punktu widzenia bezpieczeństwa nie jest zalecane, a wręcz odchodzi się od tego rozwiązania, ponieważ moduł ten działa w kontekście serwera WWW i ma dostęp do wszystkich jego struktur danych, w tym do tablicy procesów dzieci. Błąd w PHP można wykorzystać do przeprowadzenia ataku DoS (ang. *Denial of Service*) na serwer Apache, a nawet na cały serwer - proces odpowiadający za „odradzanie” (ang. *spawn*) procesów pomocniczych działa na prawach roota, więc może „zabić” wszystkie procesy w systemie.

W mojej pracy wykorzystałem również serwer baz danych MariaDB, jeśli więc chcę, aby korzystały z niego aplikacje internetowe, muszę dodać jego obsługę w języku PHP. Ponadto zainstaluję obsługę powszechnie stosowanej biblioteki graficznej GD

[4] Dennis Matotek, James Turnbull, Peter Lieverdink, *Profesjonalne administrowanie systemem*, Wydanie II, Gliwice 2018, HELION, s. 454

[5] *https://w3techs.com/technologies/overview/content_management/all*

(służącej do przetwarzania i modyfikowania plików obrazów), biblioteki konwersji ciągów znaków mbstring (zapewniającej obsługę kodowania wielobajtowych łańcuchów znaków), a także protokołu pocztowego IMAP (uaktywnia funkcje protokołów pocztowych IMAP i POP3). Dzięki temu ostatniemu będzie można również zainstalować i wykorzystywać aplikacje pocztowe dostępne w WordPress bazujące na języku PHP.

1.4.3. MariaDB

Gdy mowa jest o serwerze usług WWW (ang. *web services*) jedną z najczęściej występujących obecnie konfiguracji usługi internetowej świadczonej przez serwer WWW jest wspomniany już model LAMP. Maria DB stanowi bezpośrednie zastępstwo bazy danych MySQL. Wiele poleceń środowiska MySQL jest używanych w bazie danych MariaDB, natomiast zmienne konfiguracyjne i środowiskowe są identyczne, co pozwala na łatwe korzystanie naprzemiennie z obydwu typów serwera. Serwer MariaDB został stworzony, ponieważ część twórców technologii MySQL niewykupiona przez korporację Oracle dąży do tego, aby pozostała bezpłatna i objęta licencją GNU GPL. Silnik MariaDB w dalszym ciągu wykorzystuje bazowy kod serwera MySQL i chociaż obydwie bazy danych różnią się od siebie, to ich kolejne wersje są ze sobą w pewien sposób powiązane.

W MariaDB tak samo jak w MySQL administrator zarządza uprawnieniami za pomocą instrukcji GRANT. Przyjmuje ona szereg parametrów definiujących działania, które może wykonywać użytkownik na danym komputerze oraz na określonym obiekcie. Zgodnie z przyjętymi zasadami bezpieczeństwa dotyczącymi baz danych, administrator powinien utworzyć konto użytkownika, który może wykonywać określone operacje na pojedynczej bazie danych. W praktyce oznacza to, że każda aplikacja wykorzystująca serwer baz danych MariaDB otrzymuje własne konto użytkownika oraz własną bazę danych. Jeśli przykładowo kod PHP strony internetowej będzie zawierał błąd dający dostęp do serwera bazodanowego, będą narażone jedynie dane w bazie danych wykorzystywanej przez tę stronę.

1.5. Bezpieczeństwo serwera poczty

Jednym z najczęstszych powodów wdrażania serwera linuksowego jest chęć zapewnienia usług pocztowych, w tym odbierania i wysyłania wiadomości e-mail, a także ich odzyskiwania za pomocą takich mechanizmów jak protokół IMAP (ang.

Internet Message Access Protocol definiowany jako internetowy protokół dostępu do wiadomości) czy POP3 (ang. *Post Office Protocol* definiowany jako protokół węzłów pocztowych). W tym rozdziale opiszę aplikacje pełniące powyższe funkcje:

- Postfix – serwer pocztowy, który bazuje na protokole SMTP (ang. *Simple Mail Transfer Protocol* – prostym protokole transportowym poczty),
- Dovecot – serwer, który wykorzystuje protokoły IMAP oraz POP3.

Zaprezentuję także podstawowe działanie aplikacji chroniących pocztę przed spamem - SpamAssassin, wirusami - ClamAV oraz objaśnię, w jaki sposób ochronić serwer przy pomocy Fail2ban.

1.5.1. Postfix

Istnieje kilka serwerów pocztowych w Linuksie – na przykład Postfix, Exim i Sendmail. Zasady zabezpieczania zaprezentowane w tej pracy można zastosować do dowolnych serwerów pocztowych, lecz na potrzeby przykładów konfiguracji wybrałem Postfix w roli serwera poczty e-mail. Postfix został napisany przez eksperta ds. zabezpieczeń Pana Wietse Venema specjalnie z myślą o bezpieczeństwie. Postfix „z pudełka" jest skonfigurowany z ustawieniami domyślnymi zapewniającymi bezpieczeństwo. Konfiguracja serwera Postfix jest dość prosta i jednoznaczna, dzięki czemu w razie potrzeby można łatwo coś zmienić, a przykłady zamieszczone w rozdziale są łatwe do przeanalizowania.

Istotną kwestią jest zrozumienie podstawowych kroków dotyczących hartowania przez administratora, który może łatwo wprowadzić zmianę w konfiguracji, która zrujnuje bezpieczeństwo serwera poczty. Jednym z najważniejszych obowiązków administratora serwera pocztowego w zakresie bezpieczeństwa jest niedopuszczenie do tego, aby serwer został wykorzystany w roli systemu open relay. Serwery pocztowe zasadniczo realizują dwie funkcje: odbierają wiadomości e-mail na adresy, za które są odpowiedzialne oraz umożliwiają komputerom w sieci przesyłanie za swoim pośrednictwem wiadomości (tzw. przekazywanie – ang. *relaying*). Prawidłowo skonfigurowany serwer odbiera pocztę e-mail tylko dla tych domen, które zostały dodane w jego konfiguracji jako docelowe i powinien zezwalać na wykorzystanie siebie w roli przekaźnika tylko uprawnionym komputerom. Natomiast open relay to taki serwer pocztowy, który przekazuje wiadomości e-mail w imieniu dowolnego nadawcy oraz do dowolnej domeny. Kiedy spamer zidentyfikuje serwer pocztowy jako open relay, można się spodziewać, że liczba wiadomości e-mail przetwarzanych przez serwer

znacznie wzrośnie. Spamer będzie bowiem wysyłał za pomocą takiego serwera tyle spamu, ile zdoła, aż serwer e-mail trafi na jedną z wielu w internecie „czarnych list" serwerów spamu. Więcej na ten temat znajduje się w podrozdziale SpamAssassin.

Głównym celem stosowania podstawowych czynności z zakresu utwardzania jest ograniczenie grona użytkowników, którzy mogą korzystać z wybranego serwera poczty e-mail.

1.5.2. Dovecot

Pełna konfiguracja systemu Dovecot jako serwera POP/IMAP wykracza poza zakres niniejszego rozdziału. W instrukcji wdrażania utwardzonego serwera poczty opisałem tworzenie wirtualnych kont użytkowników w katalogu Maildir, który znajduje się w katalogu domowym wraz z całą procedurą tworzenia kont w bazie danych MariaDB. Użytkownicy mogą się logować i korzystać z protokołów SMTP, IMAP oraz POP3.

Dovecot jest serwerem poczty elektronicznej typu open source IMAP i POP3 dla systemów Linux/UNIX, napisanym z myślą o bezpieczeństwie. Dovecot to doskonały wybór zarówno dla małych jak i dużych instalacji. Jest szybki, prosty w konfiguracji, nie wymaga specjalnej administracji i zużywa bardzo mało pamięci.

Dovecot jest jednym z najwydajniejszych serwerów IMAP, a jednocześnie obsługuje standardowe formaty mbox i Maildir. Skrzynki pocztowe są transparentnie indeksowane, co daje Dovecot dobrą wydajność przy zachowaniu pełnej kompatybilności z istniejącymi narzędziami do obsługi skrzynek pocztowych. Uwierzytelnianie użytkowników Dovecot jest niezwykle elastyczne i bogate w funkcje, obsługujące wiele różnych baz danych i mechanizmów uwierzytelniania.

1.5.3. SpamAssassin

Jednym ze sposobów walki ze spamem i wirusami jest integracja SpamAssassin z agentem Postfix. Program ten bazuje na naiwnym klasyfikatorze Bayesa[6]. Filtrowanie bayesowskie polega na przewidywaniu prawdopodobieństwa powiązania danego słowa, wyrażenia lub jakiegoś innego elementu w wiadomości e-mail ze spamem. Każda wiadomość jest sprawdzana, czy jest legalna lub SpamAssassin ma ją potraktować, jako spam na podstawie wartości numerycznej, która jest wyliczona za pomocą szeregu modyfikowalnych testów lub reguł. Domyślnie wiadomość e-mail z wynikiem

[6] *http://obfusc.at/ed/bayes_filtering_pl.html*

przekraczającym ocenę 5,0 zostaje oznaczona, jako spam. Wartość tę jednak można zmienić w razie potrzeby w konfiguracji SpamAssassin. W zależności od wagi reguły może ona dodawać lub odejmować wartość od ostatecznej oceny. Na przykład taka reguła może sprawdzać określony parametr w wiadomościach e-mail i jeżeli wynik operacji będzie pozytywny, do ostatecznej oceny zostanie dodane pół punktu. Filtry Bayesa są również w stanie uczyć się wzorców na podstawie wiadomości, które przychodzą na skrzynkę użytkownika i można je trenować w odróżnianiu spamu od prawidłowych wiadomości. Informacje uzyskiwane w wyniku takiego treningu są dodawane do bazy danych, a następnie wykorzystywane do przeprowadzania dokładniejszej analizy przychodzących wiadomości. Program SpamAssassin działa na serwerze jako usługa, to znaczy, że wiadomości e-mail są do niego przekazywane, analizowane, a następnie zwracane z odpowiednim oznaczeniem dodanym do nagłówka wiadomości. Aplikacja SpamAssassin dodaje sześć różnych nagłówków:

- X-Spam-Checker-Version – wersja programu SpamAssassin,
- X-Spam-Level – ocena symbolizowana przez liczbę gwiazdek,
- X-Spam-Flag – widoczna jedynie wtedy, gdy wiadomość e-mail zostanie sklasyfikowana jako spam,
- X-Spam-Status – stan spamu (Yes albo No), całkowity wynik testów oraz lista użytych testów,
- X-Spam-Report – objaśnienie poszczególnych składowych końcowego wyniku,
- X-Spam-Prev-Subject – poprzednia nazwa tematu wiadomości.

Pierwsze trzy nagłówki są dodawane do wszystkich wiadomości e-mail. Pozostałe są wstawiane wyłącznie do wiadomości uznanych za spam.

1.5.4. Filtrowanie poczty za pomocą aplikacji Sieve

Składnia, która służy do filtrowania wiadomości e-mail za pomocą programu Sieve jest prosta. Administrator może wprowadzić filtry globalne lub lokalne w katalogach domowych.

Program Sieve używa kilku prostych poleceń i instrukcji warunkowych. Ich spis można uzyskać po wpisaniu poniższego polecenia:

```
$ doveconf -n managesieve_sieve_capability
```

Listę potrzebnych komend umieszcza się na początku pliku (zazwyczaj ~/.dovecot.sieve), które powinny zostać umieszczone w katalogach Maildir użytkowników.

1.5.5. Filtrowanie antywirusowe poczty – ClamAV

Administrator może zintegrować ClamAV z agentem Postfix w formie tzw. miltera, czyli filtra poczty (ang. *mail filter*). Program ten w dużej mierze przypomina produkty takich firm jak Symantec czy McAfee. W przeciwieństwie jednak do produktów wspomnianych firm, oprogramowanie i aktualizacje sygnatur w programie ClamAV są bezpłatne. Agent Postfix pozwala na przesłanie wiadomości e-mail do filtra, a następnie jej ponowne zakolejkowanie. Istnieją dwa rodzaje milterów Postfix – te, które bazują wyłącznie na demonie smtpd oraz te, które nie wymagają protokołu SMTP. W instrukcji wdrażania zaprezentowałem technikę wdrożenia filtra bazującego na demonie smtpd.

```
sieć → smtpd → filtr → smtpd → dostarczenie
```

Tak w uproszczeniu wygląda proces.

W pierwszej kolejności administrator serwera musi zainstalować i skonfigurować skaner ClamAV wraz z jego demonem, nazwanym clamd. Powinien także zainstalować i zaktualizować narzędzie FreshClam, które automatycznie pobiera i instaluje sygnatury wirusów w programie ClamAV. Następnie postępując zgodnie z instrukcją musi zintegrować ClamAV z Postfix.

1.5.6. Zabezpieczenie serwera za pomocą Fail2ban

Większość serwerów Linuksa oferuje logowanie SSH przez Port 22 do celów zdalnej administracji. Port ten jest dobrze znanym portem, dlatego często jest atakowany przez brutalne ataki siły. Fail2ban jest oprogramowaniem, które skanuje pliki dziennika w poszukiwaniu prób logowania z użyciem brutalnej siły w czasie rzeczywistym i zakazuje napastników za pomocą firewalld lub iptables. Fail2ban rozpoznaje niepożądany dostęp lub próby naruszenia bezpieczeństwa do serwera w określonym przez administratora czasie i blokuje adresy IP, które wykazują oznaki ataków brutalnej siły lub ataków słownikowych. Program ten działa w tle i stale skanuje pliki dziennika w poszukiwaniu nietypowych wzorców logowania i prób naruszenia bezpieczeństwa. Fail2ban jest w stanie zablokować próby zgadnięcia hasła. Można użyć go także do banowania po trzech nieudanych próbach logowania podczas

uwierzytelniania klienta poczty. Opis instalacji i konfiguracji Fail2ban znajduje się w instrukcji utwardzania serwera.

1.6. Bezpieczeństwo serwera plików

Serwerem plików (ang. *file server*) udostępnia w sieci komputerowej określone zasoby plikowe. Udostępnianie plików może być zrealizowane na dwa sposoby: za pomocą protokołu komunikacyjnego i sieciowego systemu plików. Udostępnianie danych klientom Microsoft Windows lub Mac OS możliwe jest dzięki narzędziu o nazwie Samba, które opiszę poniżej. Udostępnianiem plików (ang. *file sharing*) nazywamy możliwość współdzielenia dokumentów pomiędzy użytkownikami.

W dzisiejszych czasach coraz rzadziej przechowuje się swoje dokumenty na własnym komputerze i przekazuje je np. za pomocą poczty elektronicznej. Obecnie są one gromadzone w jednym miejscu i pozwalamy pracownikom uzyskiwać do nich dostęp w kontrolowany i bezpieczny sposób. W ten sposób redukowane jest rozprzestrzenianie się różnych wersji jednego pliku wśród użytkowników, a ograniczanie dostępu i tworzenie kopii zapasowych dokumentów zostają znacznie ułatwione.

1.6.1. Samba

Samba jest oprogramowaniem umożliwiającym uruchomienie tak zwanego serwera plików, na systemie Linux. Jest ono kompatybilne z systemem Windows, dlatego jest bardzo powszechnie stosowane w firmach, uczelniach, instytucjach, zakładach produkcyjnych, czy nawet bankach. Charakteryzuje się bardzo dużym bezpieczeństwem i stabilnością pracy. W poniższym podrozdziale opiszę jak zainstalować i poprawnie skonfigurować Sambę na przykładzie dystrybucji Linux Fedora. Samba działa zgodnie z tradycyjnym modelem klient-serwer: demon akceptuje żądania pochodzące od klientów sieciowych.

Podstawą działania Samby są protokoły CIFS (ang. *Common Internet Filesystem* – powszechny internetowy system plików) i SMB (ang. *Server Message Block* – blok komunikatu serwera), które obsługują komunikację pomiędzy komputerami korzystającymi z systemu Windows, dzięki czemu program Samba jest kompatybilny z klientami i usługami domenowymi Microsoft Windows, co jest dość istotne w środowisku biurowym, gdzie są powszechnie używane przez pracowników.

Samba działa zgodnie z tradycyjnym modelem klient-serwer, który oparty jest na systemie zapytań generowanych przez klienta i odpowiedzi od serwera. Wyjątkiem od tej zasady jest mechanizm tzw. oplock (ang. *opportunistic lock*), gdzie serwer generuje sygnał, który informuje o zerwaniu wcześniej założonego oplock-a (blokady).

Na dzień dzisiejszy funkcjonują dwie obsługiwane wersje aplikacji Samba: 3.x i 4.x. Wybór odpowiedniej wersji zależy od tego, co chce osiągnąć administrator. Wersja 4.x naśladuje usługę Active Directory (AD), natomiast wersja 3.x bardziej przypomina środowisko NT 4. Ta druga obsługuje integrację z protokołem LDAP (ang. *Lightweight Directory Access Protocol* – lekki protokół dostępu do katalogu), z kolei w wersji 4.x ta funkcja jest ciągle eksperymentalna.

Samba pracuje w dwóch trybach:

1. Jako kontroler domeny NT. Samba od wersji 2.X potrafi funkcjonować jako samodzielny kontroler domeny (ang. *domain controller*) lub przyłączyć się do niej (ang. *domain member*). Samba 4 na chwilę obecną obsługuje systemy Windows do wersji Windows Server 2012 R2, jednak zanim serwer 2012 zostanie przyłączony do kontrolera Samba AD, wymaga obecności serwera Windows 2008.
2. Jako zwykły serwer plików i drukarek w grupie roboczej (WORKGROUP) - to najczęstszy i najprostszy sposób korzystania z serwera Samba. Oczywiście Samba obsługuje dla klientów Windows tzw. logon domain, czyli logowanie do domeny, polegające na tym, że na serwerze przechowywane są profile użytkowników, jest to wygodne i eleganckie rozwiązanie zarządzania użytkownikami.

Pakiet Samba zawiera:

- serwer SMB udostępniający pliki i drukarki (w stylu Windows NT),
- serwer nazw (nameserver) NetBIOS zgodny ze specyfikacją opisaną w dokumentach RFC 100
- narzędzia znakowe, które pozwalają na uzyskanie dostępu do innych serwerów SMB z poziomu systemu Linux/Unix,
- programy pomocnicze i administracyjne.

2. Instalacja, konfiguracja i zabezpieczenie wybranej roli sieciowej w praktyce.

Ochrona serwerów stanowi spore wyzwanie. Zarówno aplikacje internetowe, pocztowe, serwisu do obsługi plików jak i platformy serwerów, które je uruchamiają są dużym źródłem luk w zabezpieczeniach. Polityka ograniczania dostępu i konwencjonalne polityki kontroli dostępu, zapory ogniowe, a także systemy wykrywania i zapobiegania włamaniom są skuteczne w wykrywaniu większości ataków. Nie są one jednak w stanie wykryć chociażby ataków typu „hijack" (porywających dostęp) do aplikacji internetowych. W niniejszej pracy przedstawiono praktycznie podejście do osiągnięcia celów bezpieczeństwa, identyfikacji różnych zagrożeń i ich eliminacji oraz zabezpieczenia kluczowych elementów serwera.

W tym rozdziale skupię się natomiast na zabezpieczaniu poszczególnych typów serwerów, ze względu na zróżnicowane potrzeby dotyczące poszczególnych konfiguracji. Spośród szerokiej gamy dystrybucji Linux, wybrałem trzy dystrybucje: CentOS 7.6, Fedora 29 oraz Debian 9.8.0. Wykorzystałem Tripwire do monitorowania integralności lokalnego systemu plików we wszystkich trzech dystrybucjach, który został zainstalowany z dystrybucji. Na wszystkich serwerach włączona została `quota` sterująca przydziałem miejsca na dysku dla poszczególnych użytkowników i grup.

W dystrybucji CentOS 7.6 oraz Fedora 29 SELinux jest zainstalowany domyślnie w trakcie instalacji minimalnej. W pliku `/etc/selinux/config` włączony został w obu dystrybucjach tryb enforcing. Na obu serwerach została zainstalowana także usługa setroubleshoot za pomocą polecenia:

```
$ sudo yum install setroubleshoot
```

Usługa setroubleshoot ma na celu uczynienie SELinuksa bardziej przyjaznym. Gromadzi ona zdarzenia audytu SELinux z jądra i uruchamia serię wtyczek analitycznych w celu zbadania naruszenia dostępu wykrytego przez SELinux. Następnie zapisuje wyniki analizy i sygnalizuje wszystkim klientom, którzy zażądali powiadomienia o tych zdarzeniach. Wykorzystywanym narzędziem jest sealert, które prezentuje powiadomienia w terminalu na temat błędów i sposobów ich naprawy.

Sprawdzono status usługi poleceniem: `sestatus`. Usługa działa na obu serwerach w trybie `enforcing`. Wykonano audyt, który nie znalazł błędów przy pomocy polecenia: `$ sudo sealert -a /var/log/audit/audit.log`

2.1. Serwer web

Serwer WWW ma realizować obsługę stron internetowych wymaganych przez komputery klienckie. Na potrzeby serwera web wykorzystałem dystrybucję Debian 9.8.0, która posiada kernel oznaczony numerem 4.9.0.8.

W mojej pracy postanowiłem wykorzystać najbardziej powszechną konfigurację, którą często można spotkać pod nazwą LAMP – zestaw oprogramowania open source stanowiący popularną platformę serwerową dynamicznych stron WWW:

- Linux (system operacyjny)
- Apache (serwer www) wersja
- MariaDB (serwer bazy danych)
- PHP (parser języka skryptowego PHP)

Pomimo że żaden z tych elementów nie został stworzony z myślą do współdziałania z pozostałymi, to taki zestaw oprogramowania jest bardzo popularny ze względu na niski koszt i dostępność wszystkich komponentów, ponieważ są one dostępne dla większości dystrybucji Linuksa.

Na potrzeby serwera web wykorzystałem laptop marki Toshiba model Satellite A210-19B, który posiada pamięć operacyjną RAM DDR2 o pojemności 2048 MB, dysk twardy TOSHIBA MK2046GSX o pojemności 200GB, 5400 rpm Serial-ATA/300, procesor AMD Turion 64 X2 TL-60 2,0. BIOS marki Phoenix w wersji 2.0 został zabezpieczony hasłem przed modyfikacjami. Dysk twardy został ustawiony jako pierwszy napęd. Wyłączona została obsługa USB.

GRUB został zabezpieczony zgodnie z opisem w instrukcji wdrażania utwardzonego serwera.

2.1.1. Apache

Na potrzeby realizacji serwera web został zainstalowany Apache2 z dystrybucji Debian stretch w wersji 2.4.25 za pomocą polecenia:

```
$ sudo apt install apache2
```

Następnie ustawiono uruchamianie podczas startu systemu oraz włączono Apache przy pomocy poleceń:

```
$ sudo systemctl enable apache2 && sudo systemctl start apache2.
```

Po ponownym uruchomieniu serwera zaobserwowano za pomocą narzędzia netstat, że nasłuchuje on na porcie 443: `$ sudo netstat -lnpt | grep 443`

```
tcp6 0 0 :::443 :::* LISTEN 3515/apache2
```

Instalacja certyfikatu TLS.

Zainstalowany został certyfikat TLS przy pomocy narzędzia certbot, które zostało zainstalowane z dystrybucji Debian 9 przy pomocy polecenia:

```
$ sudo apt install certbot python-certbot-apache -t stretch-backports
```

Następnie wygenerowane zostały klucze publiczny i prywatny jak poniżej:

```
$ sudo certbot --apache
```

Klucze zapisane zostały w katalogu /etc/letsencrypt/live/sysadmin.info.pl i dołączone do pliku wirtualnego hosta <VirtualHost *:443></VirtualHost>.

Włączenie obsługi HTTPS.

Obsługa protokołu HTTPS została dodana do istniejącego wirtualnego hosta w Apache poprzez następującą konfigurację:

```
<VirtualHost *:443>
SSLEngine on
SSLCertificateFile /etc/letsencrypt/live/sysadmin.info.pl/fullchain.pem
SSLCertificateKeyFile /etc/letsencrypt/live/sysadmin.info.pl/privkey.pem
# Tutaj powinny się znaleźć wszystkie pozostałe ustawienia konfiguracji wirtualnego hosta
</VirtualHost>
```

Po restarcie serwera Apache przy pomocy polecenia `sudo systemctl restart apache2`, gdy przejdziemy pod adres https://sysadmin.info.pl w przeglądarce jest widoczna ikona kłódki. Po kliknięciu na kłódkę wyświetlone zostaną informacje o certyfikacie. Do przetestowania witryny można także użyć narzędzia s_client z OpenSSL:

```
$ openssl s_client -connect sysadmin.info.pl:443
```

Certyfikat został sprawdzony na stronie https://www.digicert.com/help/ i potwierdzona została w ten sposób ważność certyfikatu.

Przekierowanie http na https.

Ruch http jest przekierowany na https za pomocą mod_rewrite, który został włączony poleceniem:

```
$ sudo a2enmod rewrite
```

Przekierowanie jest realizowane poprzez wpis w pliku konfiguracyjnym wirtualnego hosta <VirtualHost *:80></VirtualHost>

```
RewriteEngine on
RewriteCond %{SERVER_NAME} = sysadmin.info.pl
RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]
```

Włączenie obsługi HSTS.

Włączona została również obsługa protokołu HSTS za pomocą poniższego wpisu w pliku wirtualnego hosta oraz poprzez włączenie modułu headers za pomocą polecenia:

```
$ sudo a2enmod headers
```

```
<VirtualHost *:443> Header always set Strict-Transport-Security "max-
age=15768000; includeSubdomains;"</VirtualHost>
```

Zabezpieczenie komunikacji pomiędzy Apache i przeglądarką.

Z generatora na stronie *Server Side TLS*, o której wspomniałem w paragrafie 1.4.1 skopiowana została sekcja profilu intermediate do konfiguracji wirtualnego hosta, w celu zabezpieczenia komunikacji pomiędzy serwerem Apache a przeglądarką klienta przed przechwyceniem komunikacji. Jest to tzw. utajnienie przekazywania (ang. *forward secrecy*) HTTPS, które polega na stworzeniu dla każdej sesji unikatowych, niedeterministycznych tajemnic. Dzięki temu nawet, jeśli napastnik zdoła złamać szyfr używany w jednej sesji, nie jest w stanie wykorzystać tych informacji do łatwiejszego złamania sesji w przyszłości.

Włączenie WAF mod_security oraz OWASP.

W celu zwiększenia poziomu bezpieczeństwa zainstalowano zaporę WAF w postaci modułu mod_security dostępnego w Apache za pomocą polecenia:

```
$ sudo apt install libapache2-mod-security2 modsecurity-crs
```

A następnie włączono moduł:

```
$ sudo a2enmod security2
```

Zasady OWASP znajdują się obecnie w `/usr/share/modsecurity-crs/rules` i są ładowane przez `/usr/share/modsecurity-crs/owasp-crs.load`, który jest włączony przez `security2.conf` znajdujący się w `/etc/apache2/mods-available/security2.conf`.

Ze względu na konflikt PHP w wersji 7.3 z pozostałymi dwoma trybami modułami Apache mpm_event oraz mpm_worker, o których wspomniałem w paragrafie 1.4.1., zmuszony zostałem wybrać moduł przedwstępny mpm_prefork. Po

wprowadzeniu tych zmian ponownie uruchomiono serwer WWW i sprawdzono, czy serwer jest osiągalny z popularnych przeglądarek używanych przez użytkowników.

2.1.2. PHP

PHP w wersji 7.3.3 zostało zainstalowane wraz z rozszerzeniami z dystrybucji Debian 9 za pomocą polecenia:

```
$ sudo apt install php7.3 libapache2-mod-php7.3 php7.3-bz2 php7.3-cgi php7.3-cli php7.3-common php7.3-curl php7.3-fpm php7.3-gd php7.3-gmp php7.3-imap php7.3-intl php7.3-json php7.3-mbstring php7.3-mysql php7.3-opcache php7.3-pspell php7.3-readline
```

Zaprezentuję w skrócony sposób użycie demona PHP-FPM wewnątrz wirtualnego serwera (pełny opis został umieszczony w instrukcji wdrażania utwardzonego serwera).

Najpierw sprawdzam, czy FPM (ang. *FastCGI Process Manager*) działa.

`sudo systemctl status php7.3-fpm`

W następnym kroku ustawiam, aby proces włączał się wraz ze startem serwera `sudo systemctl enable php7.3-fpm.service` i uruchamiam go przełącznikiem `start`. Wyłączam moduły od starszych wersji PHP oraz profile mpm event i worker. `sudo a2dismod php7.0 php7.1 php7.2 mpm_event mpm_worker`. Włączam moduł Mod PHP oraz profil mpm prefork poleceniami: `sudo a2enmod mpm_prefork` i `sudo a2enmod php7.3`. Restartuję serwer Apache: `sudo systemctl restart apache2`. W następnym kroku tworzę plik konfiguracyjny sysadmin.info.pl.conf mojej strony dla FPM w katalogu /etc/php/7.3/fpm/pool.d, który zaprezentowany jest poniżej:

```
[sysadmin.info.pl]
group = php
listen = 127.0.0.1:9000
pm = dynamic
pm.max_children = 40
pm.max_requests = 500
pm.process_idle_timeout = 10s
user = php
pm.start_servers = 15
pm.min_spare_servers = 15
pm.max_spare_servers = 25
```

W pliku konfiguracyjnym wirtualnego hosta <VirtualHost *:80></VirtualHost> dodano dyrektywę ProxyPassMatch:

```
ProxyPassMatch ^/(.*\.php(/.*)?)$
fcgi://127.0.0.1:9000/var/www/html/sysadmin.infopl/public_html
```

W pliku konfiguracyjnym wirtualnego hosta <VirtualHost *:443></VirtualHost> dodano dyrektywę SetHandler,

```
<FilesMatch ^/(.*\.php(/.*)?)$>
SetHandler
"fcgi://sysadmin.info.pl:9000/var/www/html/sysadmin.info.pl/public_html"
</FilesMatch>
```

Te dwie dyrektywy sprawiają, że dopuszczane są wszystkie pliki php, które następnie są przekazywane demonowi PHP FPM.

Set Handler wymusza przetwarzanie przez FastCGI plików PHP, które wykorzystuje moduł Mod Rewrite do przepisywania adresów, co jest jednym z wymogów instalacji Wordpress. Dyrektywa ProxyPassMatch zezwala na użycie wyrażenia regularnego odwzorowującego adres URL (ang. *Uniform Resource Locator* – ujednolicony lokalizator zasobów), który jest przekazywany serwerowi pośredniczącemu. Podajemy parametry nasłuchiwania tego serwera oraz ścieżkę katalogu, w którym dany program PHP ma zostać zainstalowany. Po skonfigurowaniu FastCGI i restarcie serwer Apache jest gotowy do korzystania z witryn internetowych oraz do obsługi aplikacji napisanych w języku PHP.

2.1.3. AppArmor

W celu zapewnienia dodatkowej warstwy ochrony serwera web podjęta została decyzja o instalacji i włączeniu AppArmor, jako jednego z ważniejszych czynników zabezpieczających serwer przed zagrożeniami.

AppArmor został zainstalowany z dystrybucji Debian 9 za pomocą poniższego polecenia:

```
$ sudo apt install apparmor apparmor-utils auditd
```

Następnie został zainstalowany starter oraz dodatkowe moduły:

```
$ sudo apt install apparmor-profiles apparmor-profiles-extra
```

Ponieważ AppArmor jest modułem jądra Linuksa, należy go włączyć za pomocą następujących poleceń:

```
$ sudo mkdir -p /etc/default/grub.d
```

Powyższe polecenie utworzyło katalog grub.d, w którym stworzono plik apparmor.cfg za pomocą polecenia:

```
$ sudo -e /etc/default/grub.d/apparmor.cfg
```

Do pliku wklejono następującą zawartość:

```
GRUB_CMDLINE_LINUX_DEFAULT="$GRUB_CMDLINE_LINUX_DEFAULT
apparmor=1 security=apparmor"
```

Zapisano zmiany i wykonano aktualizację grub za pomocą polecenia:

```
$ sudo update-grub
```

Następnie zrestartowano serwer poleceniem:

```
$ sudo systemctl reboot
```

Po ponownym uruchomieniu, sprawdzono, czy AppArmor jest włączony przez uruchomienie komendy `aa-enabled`. Na ekranie został wyświetlony komunikat `Yes`. W następnej kolejności zostały sprawdzone polityki AppArmor poleceniem:

```
$ sudo apt policy apparmor
```

Wynik wyświetlony w terminalu został zaprezentowany poniżej:

```
apparmor:
 Installed: 2.11.0-3+deb9u2
 Candidate: 2.11.0-3+deb9u2
 Version table:
 *** 2.11.0-3+deb9u2 500
 500 http://ftp.pl.debian.org/debian stretch/main amd64 Packages
 500 http://deb.debian.org/debian stretch/main amd64 Packages
 100 /var/lib/dpkg/status
```

Włączono AppArmor jako usługę podczas startu systemu poleceniem:

```
$ sudo systemctl enable apparmor
```

A następnie sprawdzono jej status poprzez:

```
$ sudo systemctl status apparmor
```

Usługa jest aktywna po restarcie systemu. W celu ochrony serwera Apache zainstalowano profil poleceniem:

```
$ sudo apt install libapache2-mod-apparmor
```

Następnie włączono ochronę profilu w trybie enforce:

```
$ sudo aa-enforce /etc/apparmor.d/usr.sbin.apache2
```

W kolejnym kroku należy włączyć moduł przedwstępny Apache, który już wcześniej został włączony za pomocą polecenia:

```
$ sudo a2enmod mpm_prefork
```

Włączono moduł AppArmor dla Apache i zrestartowano serwer poleceniami:

```
$ sudo a2enmod apparmor
$ sudo service apache2 restart
```

Polecenie `aa-status` pokazało załadowanych 47 profili, z czego 14 w trybie `enforce` i 33 w trybie `complain`. Dodatkowo wyświetlona została informacja o tym, że 36 procesów ma zdefiniowane profile, które działają w trybie `enforce`. Są to procesy Apache. Serwer został zabezpieczony przy pomocy AppArmor.

Przeglądanie logów możliwe jest dzięki zainstalowanemu auditd poleceniem:

```
tail -f /var/log/auditd/auditd.log | grep 'DENIED'
```

2.1.4. Serwer baz danych – MariaDB

Na potrzeby serwera baz danych oraz serwera poczty wykorzystałem laptop marki eMachines model E725, który posiada pamięć operacyjną RAM DDR2 o pojemności 4096 MB, dysk twardy Western Digital WD2500BEVT o pojemności 250GB, 5400 rpm Serial-ATA/300, procesor Pentium Dual-Core T4200 2,0 GHz 64-bit. BIOS marki InsydeH2O w wersji 1.03 został zabezpieczony hasłem przed modyfikacjami. Dysk twardy został ustawiony jako pierwszy napęd.

GRUB2 został zabezpieczony zgodnie z opisem w instrukcji wdrażania utwardzonego serwera.

Serwer baz danych ma realizować obsługę bazy danych wymaganej przez serwer web. W tym celu została zainstalowana dystrybucja CentOS 7.6, która posiada kernel oznaczony numerem 3.10.0-957.10.1.el7.x86_64.

W tym paragrafie opiszę, jak można skonfigurować serwer bazodanowy MariaDB w podstawowym zakresie. Natomiast, w jaki sposób zabezpieczyć komunikację z serwerem WWW przy pomocy aplikacji openssl, która pozwala na tworzenie certyfikatów SSL (ang. *Secure Sockets Layer* – warstwa bezpiecznych gniazd) lub TLS (ang. *Transport Layer Security* – zabezpieczenie warstwy transportowej) oraz sposób instalacji systemu zarządzania treścią WordPress, który wybrałem ze względu na to, że jest to jeden z najczęściej atakowanych systemów zarządzania treścią (ang. *Content Management System* – CMS), opisałem w instrukcji wdrażania utwardzonego serwera MariaDB.

W dystrybucji CentOS zainstalowałem serwer MariaDB za pomocą menedżera pakietów yum poleceniem:

```
$ sudo yum install mariadb mariadb-server
```

Po instalacji serwera wprowadzono kilka podstawowych zmian w ustawieniach MariaDB. W dystrybucji CentOS serwer MariaDB domyślnie nasłuchuje połączeń na wszystkich skonfigurowanych interfejsach sieciowych i adresach. W tym przypadku

baza danych znajduje się na laptopie z systemem CentOS 7.6, natomiast serwer internetowy znajduje się na laptopie z systemem Debian 9.8.0, w związku z tym ograniczono nasłuchiwanie bazy danych do jednego interfejsu, jest to rozwiązanie praktyczne, które stosuje się w dzisiejszym świecie ze względu na bezpieczeństwo. W pliku konfiguracyjnym `/etc/my.cnf.d/server.cnf` wstawiono dyrektywę bind-address w sekcji [mysqld], która wygląda następująco: `bind-address=150.10.0.11`. Serwer bazodanowy jest w ten sposób poinformowany o tym, że ma nasłuchiwać jedynie na adresie 150.10.0.11, dzięki czemu inne komputery nie są w stanie nawiązać połączenia z bazą danych. Do pliku zostały dodane jeszcze następujące elementy, które pokazane są na listingu 2.1.

Listing 2.1 *Plik /etc/my.cnf.d/server.cnf w dystrybucji CentOS*

```
[mysqld]
bind-address = 150.10.0.11
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
symbolic-links=0
[mysqld_safe]
log-error=/var/log/mariadb/mariadb.log
pid-file=/var/run/mariadb/mariadb.pid
```

Zaleca się wyłączenie łączy symbolicznych (symbolic-links=0), aby zapobiec różnorodnym zagrożeniom bezpieczeństwa.
Uruchomiłem serwer MariaDB za pomocą polecenia:

```
$ sudo systemctl start mariadb.
```

Po uruchomieniu serwera istnieje konieczność zdefiniowania hasła administratora i wyczyszczenia domyślnych tabel za pomocą narzędzia `mysql_secure_installation`, które opisane jest w instrukcji wdrażania. Po wprowadzeniu polecenia `mysql_secure_installation` i konfiguracji serwer MariaDB jest bezpieczny. Nie będzie akceptował połączeń nawiązywanych przez zdalne komputery oraz uniemożliwi łączenie się użytkownikom, którzy nie mają konta MariaDB. Aby sprawić, żeby serwer MariaDB był uruchamiany w trakcie rozruchu systemu, użyto poniższej komendy: `$ sudo systemctl enable mariadb`.

Połączyłem się z serwerem MariaDB jako administrator i stworzyłem konto użytkownika `wp_uzytk`; nadałem mu uprawnienia dostępu do bazy danych WordPress i tabel. Widoczne na listingu 2.2 polecenie tworzy użytkownika `wp_uzytk`, który może łączyć się z serwerem MariaDB jedynie z poziomu komputera lokalnego za pomocą hasła `ElF@9GoSvshO0Fn4P&MN`. Słowa kluczowe `UPDATE, INSERT, CREATE, DELETE, SELECT` decydują o przydzieleniu użytkownikowi prawidłowych uprawnień, które są wymagane przez WordPress. Natomiast skrót `baza_wp.*` odpowiedzialny jest za wyznaczenie wszystkich tabel we wskazanej bazie danych o nazwie baza_wp.

Listing 2.2 *Przydzielanie uprawnień przy pomocy komendy GRANT*

```
MariaDB [(none)]> CREATE USER 'wp_uzytk’@’localhost' IDENTIFIED BY 'password';
MariaDB [(none)]> GRANT SELECT, DELETE, CREATE, INSERT, UPDATE ON baza_wp.* TO 'wp_uzytk’@’localhost';
```

Zalecaną opcją jest ograniczenie dostępu do tablic w pojedynczej bazie danych, korzystając z wyrażenia `nazwabazy.*` oraz ustawienie uprawnień jak powyżej.

2.2. Serwer poczty

Jak już wspomniałem wcześniej na potrzeby serwera poczty wykorzystałem laptop marki eMachines model E725, którego opis znajduje się w paragrafie 2.4.1.

2.2.1. SELinux

Na serwerze poczty zainstalowałem narzędzia do zarządzania portami i usługami w SELinux poleceniem: `$ sudo yum install policycoreutils-python policycoreutils-python-utils policycoreutils-devel`. Następnie dodałem port 2244 do zestawu reguł dla SSH w SELinux: `$ sudo semanage port -a -t ssh_port_t -p tcp 2244`. Konfiguracja SELinux dla ClamAV została opisana w paragrafie 2.2.6.

2.2.2. Postfix

W dystrybucji CentOS 7.6 zainstalowałem Postfix za pomocą menedżera pakietów yum poleceniem:

```
$ sudo yum install postfix
```

Pierwszym ograniczeniem, które zostało zastosowane było wybranie zakresu adresów IP, które mogą korzystać z serwera do przekazywania poczty. Jest to szczególnie istotne, gdyż serwer jest podłączony do internetu. W konfiguracji ograniczono listę dozwolonych adresów IP tylko i wyłącznie do komputerów należących do sieci lokalnej. Listę zaufanych sieci serwera Postfix ustawiono przy pomocy parametru `mynetworks` w pliku /etc/postfix/main.cf.

```
mynetworks = 150.10.0.0/16, 127.0.0.0/8
```

Wartość tego parametru ma postać rozdzielonej przecinkami listy podsieci. Zadbać o to, aby ta wartość została ustawiona co najmniej na localhost (127.0.0.0/8) tak, aby wiadomości e-mail mogły być wysyłane z serwera poczty. Oprócz tego należy wprowadzić na tę listę co najmniej jedną dodatkową sieć wewnętrzną. Należy zachować jednak szczególną ostrożność podczas dodawania do tej listy adresów IP osiągalnych z internetu. Jeśli haker zdobędzie kontrolę nad takim hostem, to będzie mógł użyć serwera pocztowego do przekazywania spamu.

Kolejnym ważnym krokiem, który ma zapobiec temu, aby serwer Postfix stał się serwerem open relay, było zdefiniowanie konkretnych ograniczeń przekazywania przy pomocy polecenia `smtpd_relay_restrictions`. Postfix po instalacji domyślnie ma ustawione restrykcje, które gwarantują bezpieczeństwo. Można je sprawdzić za pomocą polcenia:

```
$ sudo postconf | grep smtpd_relay_restrictions
```

Na ekranie terminala powinny zostać wyświetlone poniższe ustawienia:

```
smtpd_relay_restrictions = reject_unauth_destination,
permit_sasl_authenticated, permit_mynetworks
```

Połączenia przychodzące SMTP zostały ograniczone przy pomocy dyrektywy `smtpd_recipient_restrictions`. Domyślnie to ustawienie nie ma żadnych opcji ze względu na ustawienie `smtpd_relay_restrictions`, które opisałem wcześniej, co ma na celu uniemożliwienie przekształcenia serwera poczty w `open relay`. Zdecydowałem się jednak na rozszerzenie tych ograniczeń o dodatkowe przeszukiwanie czarnych list spamerów za pomocą poniższych ustawień.

```
smtpd_recipient_restrictions = reject_rbl_client
dul.dnsbl.sorbs.net, reject_rbl_client cbl.abuseat.org,
reject_rbl_client sbl.spamhaus.org, reject_rbl_client
```

```
zen.spamhaus.org, reject_unauth_destination,
permit_sasl_authenticated, permit_mynetworks
```

Oczywiście tę konfigurację w późniejszym etapie rozszerzono o kolejne elementy. Pełną konfigurację wszystkich zabezpieczeń przedstawiłem w instrukcji wdrażania utwardzonego serwera.

Postanowiłem dodatkowo zabezpieczyć ustawienia serwera Postfix. W tym celu wstawiłem pewne opcje do pliku main.cf. Większość z nich służy do odrzucania wiadomości e-mail, które nie są zgodne ze standardem SMTP RFC, na przykład ignorują wiadomości pochodzące z nieprawidłowych adresów pocztowych. Przede wszystkim zaktualizowałem listy ograniczeń, określane podczas definiowania procesu uwierzytelniania. Część z tych ustawień opisałem już wyżej, skupię się wobec tego na tych, które nie zostały wymienione wcześniej. Dzięki komendzie postconf z flagą -e możliwe jest dołączanie dyrektyw do pliku main.cf. Dodałem więc następujące opcje:

```
$ sudo postconf –e 'smtpd_helo_required = yes'
$ sudo postconf –e 'smtpd_helo_restrictions =
reject_unknown_helo_hostname'
```

Dyrektywa `smtpd_helo_restrictions` odpowiedzialna jest za odrzucanie połączenia z serwerem poczty, w sytuacji, gdy nie zawiera on rekordów A i MX w systemie DNS. Natomiast opcja `smtpd_helo_required` odpowiedzialna jest za przesyłanie komunikatu `EHLO` od każdego agenta MTA, który łączy się z serwerem Postfix. Jeśli klient nie prześle prawidłowego powitania EHLO i nie ogłosi swojej nazwy, Postfix automatycznie odrzuci tego rodzaju połączenie. Pierwsza dyrektywa wymusza istnienie drugiej, bez której nie jest w stanie spełniać swojej funkcji. Opcjonalnie dołączyłem następujące ustawienia do list ograniczeń nadawcy, odbiorcy i danych. Do pliku main.cf wstawiłem opcje w `smtpd_sender_restrictions`.

```
$ sudo postconf –e 'smtpd_sender_restrictions =
reject_unknown_client_hostname, reject_unknown_sender_domain,
reject_non_fqdn_sender, reject_unknown_reverse_client_hostname'
```

Parametr `reject_non_fqdn_sender` powoduje odrzucanie wiadomości e-mail przez Postfix, w których adres nadawcy ma nieprawidłowy format. To znaczy, że wartość przekazywana przez polecenie MAIL FROM musi mieć postać prawidłowego adresu e-mail, np. info@sysadmin.info.pl. Adres, na jaki serwer Postfix nie byłby w

stanie odpowiedzieć, np. Info, czy info, nie zostanie zaakceptowany, więc wiadomość przez niego nadana zostanie odrzucona.

Opcja `reject_unknown_sender_domain` odpowiada za ignorowanie wiadomości pochodzących od nadawcy, który nie ma zdefiniowanych rekordów A lub MX w systemie DNS. Taka sytuacja występuje często, gdy wiadomości są wysyłane z fałszywych domen; jest to jedna z cech spamu.

Ustawienie `reject_unknown_reverse_client_hostname` jest odpowiedzialne za odrzucanie wiadomości, których nadawca nie ma zdefiniowanego adresu zwrotnego w swojej konfiguracji DNS.

Parametr `reject_unknown_client_hostname` powoduje odrzucenie wiadomości, które nie posiadają prawidłowej informacji o adresie klienta serwera poczty, z którego została wysłana wiadomość.

Dodałem także kolejne kryterium odrzucania do ustawienia `smtpd_recipient_restrictions`.

```
$ sudo postconf –e 'smtpd_recipient_restrictions =
reject_rhsbl_reverse_client dbl.spamhaus.org,
reject_rhsbl_helo dbl.spamhaus.org,
reject_rhsbl_sender dbl.spamhaus.org,
reject_unauth_pipelining’
```

Opcja `reject_rhsbl_reverse_client` sprawdza adres dbl.spamhaus.org pod względem klienta, jakim jest lista blokowanych domen (ang. *domain block list* – DBL). Rozwiązanie to pomaga odrzucać spam na podstawie kilku elementów (polecenie HELO, IP/DNS, nadawca i adresat, itd.).

Następne dwie listy blokowania domen koncentrują się na komendzie HELO (`reject_rhsbl_helo`) i nadawcy (`reject_rhsbl_sender`).

Ograniczenie `reject_unauth_pipelining` odrzuca wiadomości e-mail przesyłane specyficzną techniką zwaną potokowaniem (ang. *pipelining*) bez sprawdzania, czy mechanizm ten jest obsługiwany. Jest to metoda, którą powszechnie stosują spamerzy do rozprowadzania niechcianej poczty. Dodałem tę samą opcję do dyrektywy `smtpd_data_restrictions`, aby umożliwić wychwytywanie spamerów używających potokowania na etapie przesyłania komendy DATA.

```
$ sudo postconf –e ‘smtpd_data_restrictions =
reject_unauth_pipelining’
```

Na koniec dodałem parametr, który nie jest związany z listami ograniczeń, lecz blokuje część spamu.

```
$ sudo postconf –e 'disable_vrfy_command = yes'
```

Parametr `disable_vrfy_command` wyłącza obsługę polecenia VRFY „języka” SMTP. Komenda VRFY umożliwia nadawcy sprawdzić serwer poczty i upewnić się, że dany adres istnieje. Spamerzy używają tej komendy do wyszukiwania adresów, a hakerom od czasu do czasu przydaje się przed atakiem do zdobycia nazw użytkowników. Po wprowadzeniu zmian zrestartowałem demona Postfix.

Kolejną metodą zabezpieczenia jest zastosowanie popularnego podejścia w postaci uwierzytelniania na podstawie nazwy użytkownika i hasła. W celu określenia sposobu, w jaki użytkownicy powinni być uwierzytelniani na serwerach SMTP, utworzono protokół SASL. W przypadku Postfix protokół SASL nie jest obsługiwany bezpośrednio. Zadanie uwierzytelniania jest realizowane za pośrednictwem systemu Dovecot SASL, którego konfigurację przedstawiłem poniżej.

2.2.3. Dovecot

Instalacji Dovecot w dystrybucji CentOS 7.6 dokonałem przy użyciu menedżera pakietów yum za pomocą polecenia:

```
$ sudo yum install dovecot-mysql dovecot
```

Konfiguracja Dovecot została zmodyfikowana w taki sposób, aby zostało utworzone gniazdo UNIX, z którego serwer Postfix może korzystać, aby komunikować się z systemem Dovecot w przypadku, gdy zajdzie potrzeba przeprowadzenia uwierzytelniania użytkownika. Dokonałem edycji pliku `conf.d/10-master.conf` i dodałem następującą konfigurację w sekcji `auth{}`:

```
unix_listener /var/spool/postfix/private/auth {
      user = postfix
      group = postfix
      mode = 0660
}
```

Dzięki temu tworzone jest gniazdo `/var/spool/postfix/private/auth`, które należy do użytkownika `postfix` i grupy `postfix`. W następnym kroku włączyłem w pliku `conf.d/10-auth.conf` w konfiguracji systemu Dovecot następującą opcję:

```
auth_mechanisms = plain login
```

Po wprowadzeniu tych ustawień zrestartowano system Dovecot:

```
$ sudo systemctl restart dovecot
```

Po restarcie został utworzony plik `var/spool/postfix/private/auth`.

W następnym etapie po skonfigurowaniu systemu Dovecot trzeba poinformować serwer Postfix o tym, z jakiego pliku gniazda UNIX ma skorzystać w celu przeprowadzenia uwierzytelniania za pomocą systemu Dovecot. Dodałem następujące wiersza do pliku konfiguracyjnego Postfix:

```
smtpd_sasl_type = dovecot
smtpd_sasl_auth_enable = yes
smtpd_sasl_path = private/auth
smtpd_sasl_local_domain = $myhostname
smtpd_sasl_authenticated_header = yes
smtpd_sasl_security_options = noanonymous
broken_sasl_auth_clients = yes
```

Warto zwrócić uwagę na to, że opcję `smtpd_sasl_type` ustawiłem na `dovecot`, włączyłem uwierzytelnianie SASL za pomocą opcji `smtpd_sasl_auth_enable` oraz ustawiłem opcję `smtpd_sasl_path`, informującą Postfix o tym, gdzie znaleźć plik gniazda UNIX, który stworzyłem wcześniej. Ta konfiguracja jest dobra na początek, jednak pozostaje jeszcze kwestia włączenia na serwerze Postfix obsługi TLS, dzięki której można jeszcze bardziej ograniczyć uwierzytelnianie SASL – wykluczyć uwierzytelnianie za pomocą komunikacji jawnym tekstem dzięki szyfrowaniu za pośrednictwem TLS. Aby to zrobić, zmieniono ustawienie `smtpd_sasl_security_options` w następujący sposób:

```
smtpd_sasl_security_options = noanonymous, noplaintext
smtpd_sasl_tls_security_options = noanonymous
```

Poszedłem nawet o krok dalej i wyłączyłem wszystkie próby uwierzytelniania, jeśli nie są realizowane za pośrednictwem TLS:

```
smtpd_tls_auth_only = yes
```

Na koniec zaktualizowałem opcję `smtpd_relay_restrictions` w taki sposób, aby umożliwić uwierzytelnionym użytkownikom przekazywanie wiadomości e-mail:

```
smtpd_relay_restrictions = permit_mynetworks,
permit_sasl_authenticated, reject_unauth_destination
```

Po wprowadzeniu tych zmian ponownie uruchomiłem serwer Postfix.

```
$ sudo systemctl restart postfix
```

Skonfigurowałem klienta poczty e-mail w taki sposób, aby podczas przesyłania wiadomości e-mail było możliwe przeprowadzenie uwierzytelniania na serwerze Postfix.

Aby skonfigurować protokół SMTPS dla serwera pocztowego, trzeba uzyskać ważny certyfikat. W tym celu wykonałem poniższe polecenia:

```
$ sudo yum install certbot python2-certbot-apache
$ certbot certonly --standalone -d mail.sysadmin.info.pl
```

Po wygenerowaniu certyfikatu następnym krokiem jest poinformowanie o nim serwera Postfix. Dokonałem tego za pośrednictwem pliku konfiguracyjnego Postfix main.cf:

```
smtpd_tls_cert_file =
/etc/letsencrypt/live/mail.sysadmin.info.pl/fullchain.pem
smtpd_tls_key_file =
/etc/letsencrypt/live/mail.sysadmin.info.pl/privkey.pem
```

Let’s Encrypt podczas generowania pary kluczy ustawia prawidłowe wartości dla kluczy.

Kolejnym krokiem, który wykonano po prawidłowym skonfigurowaniu certyfikatów używanych przez serwer Postfix było ustawienie listy wykorzystywanych protokołów TLS i szyfrów:

```
smtpd_tls_mandatory_protocols = !SSLv2, !SSLv3
smtpd_tls_mandatory_ciphers = medium
smtpd_tls_received_header = yes
```

W pierwszej linijce wyłączyłem przestarzałe i niezabezpieczone protokoły `SSLv2 i SSLv3`. Dobrą praktyką jest wymaganie szyfrów o średniej sile (ang. *medium*), dzięki czemu można uzyskać przyzwoitą mieszankę zgodności ze zdalnymi serwerami pocztowymi bez dopuszczania niektórych słabszych szyfrów. Opcja `smtpd_tls_received_header` pozwala wprowadzić informacje na temat używanych protokołów i szyfrów, co pozwala na uzyskanie lepszych informacji diagnostycznych dla połączenia, gdyby takie dane okazały się potrzebne – ta opcja nie jest wymagana, jednak zaleca się jej ustawienie. Ostatnim krokiem jest ustawienie Postfix w taki sposób, aby używał zarówno połączeń dla przychodzących wiadomości e-mail, jak i wychodzących połączeń SMTP:

```
smtpd_use_tls = yes
smtp_use_tls = yes
```

Po restarcie serwera Postfix jest on gotowy do tego, aby wykorzystywać protokół TLS, jeśli zdalny serwer pocztowy będzie go obsługiwał (tak jest w przypadku produktów wielu głównych dostawców usług pocztowych).

2.2.4. SpamAssassin

Uzupełnieniem powyższych modyfikacji jest instalacja i konfiguracja programu SpamAssassin. Zacznę od zainstalowania potrzebnych pakietów. W dystrybucji CentOS 7.6 zainstalowałem pakiet SpamAssassin i skonfigurowałem demona SpamAssassin do automatycznego uruchamiania w czasie rozruchu.

```
$ sudo yum install spamassassin spamass-milter-postfix spamass-milter
```

W momencie uruchomienia aplikacja SpamAssassin włącza demona spamd. Stworzyłem więc użytkownika systemowego, który będzie obsługiwał wspomnianą usługę za pomocą polecenia:

```
$ sudo useradd –r –m –s /sbin/nologin spamd
```

Za pomocą flagi -r wyznacza się użytkownika systemowego (identyfikator UID jest mniejszy od wartości 1000), dzięki opcji -m tworzy katalog domowy, natomiast parametr -s definiuje powłokę. Powyższe polecenie jest stosowane w dystrybucji CentOS. Ustawiłem uruchamianie demona SpamAssassin podczas startu systemu przy użyciu następującego polecenia:

```
$ sudo systemctl enable spamassassin
```

Wymieniona w mojej pracy dystrybucja CentOS 7.6 przechowuje pliki konfiguracyjne aplikacji SpamAssassin w katalogu /etc/mail/spamassassin. W pliku local.cf wyznaczyłem wartość progową klasyfikowania wiadomości e-mail jako spamu (required_score) w następujący sposób:

```
$ sudo -e /etc/mail/spamassassin/local.cf
```

Dokonałem edytycji pliku i ustawiłem wg poniższego wzoru:

```
required_hits 5.0
report_safe 0
required_score 5
#rewrite_header Subject [SPAM]
remove_header ham Status
remove_header ham Level
```

Następnie wyedytowałem plik `/etc/sysconfig/spamassassin` w dystrybucji CentOS, w którym zawarte są opcje przekazywane demonowi (SPAMDOPTION). Umieściłem tu konto użytkownika i grupę, które obsługują usługę spamd.

```
SPAMDOPTIONS="-d -c -m5 -H -u spamd -g spamd"
```

Flagi -u i -g są dołączone do domyślnych ustawień. Pozostałe opcje służą do „demonizowania” (-d), tworzenia plików preferencji użytkownika (-c), określenia liczby potomnych procesów (-m), a także wyznaczenia innego katalogu pomocy (-H) – ta ostatnia przydaje się zewnętrznym usługom.

W pliku /etc/sysconfig/spamass-milter ustawiłem flagi:

```
EXTRA_FLAGS="-i 127.0.0.1 -m -r -1 -I "
```

Dokonałem edycji pliku /etc/sysconfig/spamass-milter-postfix i ustawiłem dwa parametry tak, jak poniżej:

```
SOCKET="/run/spamass-milter/postfix/sock"
SOCKET_OPTIONS="-g postfix"
```

Następnie trzeba uruchomić demona spamd. Użyłem w tym celu komendy systemctl.

```
$ sudo systemctl start spamassassin
```

Najpierw dokonałem edycji pliku `master.cf` i włączyłem opcję wysyłania wiadomości e-mail do programu SpamAssassin. Zmodyfikowałem w tym celu usługę smtp.

```
smtp inet n - n - - smtpd     -o content_filter=spamassassin
```

Wstawiłem do w/w pliku wiersz -o content_filter=spamassassin (flaga -o powinna być poprzedzona kilkoma spacjami, aby wskazać w ten sposób, że mam do czynienia z kontynuacją poprzedniego wiersza). Przy pomocy opcji `content_filter` informuję agenta Postfix, że chcę, aby wszystkie wiadomości e-mail dostarczane poprzez usługę smtp były przesyłane do filtra SpamAssassin. W kolejnym kroku trzeba zdefiniować ten filtr. W tym celu umieściłem jego opis na końcu pliku `master.cf` definiując go w następujący sposób:

```
spamassassin unix - n n - - pipe user=spamd argv=/usr/bin/spamc
-e /usr/sbin/sendmail -oi -f ${sender} ${recipient}
```

Tworzona jest w ten sposób w pliku master.cf nowa usługa typu unix (czyli gniazdo uniksowe), która wywołuje innego demona serwera Postfix o nazwie pipe. Służy on do przekazywania wiadomości e-mail do zewnętrznego polecenia. W kolejnym wierszu (także zaczynającym się odstępami) wyznaczam tę zewnętrzną komendę.

Ustalam użytkownika obsługującego tę komendę, samo polecenie oraz jego argumenty. Wywołuję polecenie spamc – plik binarny łączący się z demonem SpamAssassin, przesyłający mu nasze wiadomości e-mail, a następnie odczytujący wyniki. Przekazuję komendzie spamc argument -e. Opcja ta musi być zdefiniowana jako ostatnia w wierszu poleceń. Wskazuje ona poleceniu spamc, co należy zrobić z wiadomością e-mail po jej przeskanowaniu. W tym przypadku przekazywana jest ona instrukcji `/usr/sbin/sendmail`, która zwraca agentowi Postfix sprawdzoną wiadomość przekazaną do dostarczenia użytkownikowi (program sendmail zawiera dowiązanie symboliczne do pliku /usr/sbin/sendmail.postfix).

Sprecyzowałem także ustawienia polecenia sendmail. Opcja -io służy do tego, aby ignorować samotne kropki podczas skanowania wiadomości e-mail, ponieważ wiadomości e-mail mogą zawierać wiersze z kropkami, które nie oznaczają zakończenia treści. Parametr -f ${sender} gwarantuje, że dane nadawcy wiadomości e-mail zostaną przekazane agentowi Postfix, natomiast w ustawieniu ${recipient} zostaje określony jej adresat, przez co serwer Postfix wie, komu ją przekazać.

Możliwe jest zwiększenie szczegółowości komunikatów demona spamd poprzez dodanie flagi -D do opisanych wcześniej opcji usługi SpamAssassin. Uzyskać w ten sposób można znacznie więcej informacji i zaobserwować dokładniej mechanizmy wykonywane przez tego demona.

Jeżeli program SpamAssassin uzna wiadomość e-mail za spam, do jej tematu zostanie dołączone wyrażenie [SPAM] zgodnie z konfiguracją zapisaną w pliku `/etc/mail/spamassassin/local.cf`. W tym przypadku wiadomość spam jest standardowo dostarczana do skrzynki pocztowej adresata. Zazwyczaj ludzie przenoszą spam do osobnego folderu, przeglądają go i usuwają. Niektóre osoby z góry odrzucają lub usuwają tak oznaczone wiadomości, co czasem prowadzi do wykasowania istotnej poczty.

SpamAssassin nie jest nieomylny i bywają sytuacje, gdy administrator serwera pocztowego ma do czynienia z fałszywym alarmem, czyli zwykła wiadomość e-mail może zostać oznaczona jako spam. Wraz z wykasowaniem spamu traci się również tę wiadomość. Przechowywanie spamu przez jakiś czas w osobnym folderze zwiększa szansę na odzyskanie prawidłowych wiadomości. Do dobrych praktyk należy przeniesienie wiadomości oznaczonych jako spam do specjalnego folderu w katalogu Maildir, gdzie użytkownicy serwera poczty będą mogli go później na spokojnie przejrzeć. W tym celu można zastosować techniki wykorzystujące wymienione

wcześniej nagłówki do przenoszenia oznaczonych wiadomości. Wśród tych metod występują takie jak:

- użycie agenta LTMP (ang. *Local Mail Transfer Protocol* – lokalny protokół transportowy poczty), np. Dovecot,
- użycie agentów MDA, takich jak procmail lub maildrop,
- użycie reguł/filtrowania klienta pocztowego.

W mojej pracy wykorzystałem Dovecot do przesyłania poczty od agenta MTA do katalogów domowych użytkowników. Filtrowaniem wiadomości e-mail zajmie się kompatybilna z serwerem Dovecot aplikacja Sieve[7]. Aby móc skorzystać z serwera Dovecot, skonfigurowałem protokół LMTP – długoterminowy proces uruchamiany przez główny proces Dovecot, który uruchomiony jest cały czas w trakcie działania usługi Dovecot. Jest to główna różnica pomiędzy serwerem Dovecot a programami procmail i maildrop, które są uruchamiane przez każdą dostarczaną wiadomość e-mail. Poza tym protokół LMTP potrafi działać na osobnym serwerze, gdyż odczytuje gniazda uniksowe i TCP.

2.2.5. Filtrowanie poczty za pomocą aplikacji Sieve

Pierwszym etapem jest zainstalowanie dodatkowych pakietów, które zawierają aplikację Sieve. W przypadku dystrybucji CentOS użyłem komendy:

```
$ sudo yum install dovecot-pigeonhole
```

W pliku /etc/dovecot/dovecot.conf wstawiłem poniższy wiersz, aby Dovecot akceptował połączenia z protokołem LMTP.

```
protocols = lmtp
```

Następnie zmodyfikowałem plik `/etc/dovecot/conf.d/20-lmtp.conf`:

```
protocol lmtp {postmaster_address = info@sysadmin.info.pl mail_plugins
= quota sieve }
```

W następnym kroku dodałem do głównego procesu Dovecot proces LMTP/gniazdo uniksowe, co umożliwia serwerowi Postfix przesyłanie wiadomości. Dokonałem edycji pliku `/etc/dovecot/conf.d/10-master.conf` w poniższy sposób:

```
service lmtp {unix_listener /var/spool/postfix/private/dovecot-lmtp{
group = postfix   user = postfix    mode = 0600   }  }
```

Na koniec skonfigurowałem serwer Dovecot tak, aby nie odrzucał nazwy domeny z adresu e-mail podczas dostarczania poczty. Proces LMTP wydobywa nazwę

[7] *http://sieve.info/*

użytkownika z parametru RCPT TO:. Gdybym usunął fragment definiujący domenę, to serwer próbowałby przesłać wiadomość e-mail na adres admin zamiast do użytkownika admin@sysadmin.info.pl i skończyłoby się to wyświetleniem komunikatu błędu 550 5.1.1 User doesn't exist (użytkownik nie istnieje). Odpowiednich zmian dokonałem w pliku `/etc/dovecot/conf.d/10-auth.conf`.

```
auth_username_format = %Lu
```

Pozostało jeszcze tylko włączenie funkcji obsługi gniazda LMTP przez serwer Postfix. Użyłem do tego poniższego polecenia:

```
$ postconf -e 'mailbox_transport = lmtp:unix:private/dovecot-lmtp'
```

W pliku

```
/home/wpoczta/sysadmin.info.pl/admin/Maildir/.dovecot.sieve
```

wstawiłem poniższe dyrektywy:

```
require ["fileinto"];
if header :contains "X-Spam-Flag" "YES" {fileinto "Junk";}
```

W pliku `/etc/dovecot/conf.d/10-mail.conf` stworzyłem strukturę folderów, którą Dovecot utworzył dynamicznie w folderze użytkownika, którą sprawdziłem komendą: `$ sudo doveadm mailbox list`. W instrukcji wdrażania utwardzonego serwera poczty umieściłem przykładową strukturę. W powyższym przykładzie poczty oznaczona jako spam jest przesyłana do folderu `Junk`. Aby zmiany zostały zastosowane, trzeba uruchomić ponownie serwery Postfix i Dovecot.

```
$ sudo systemctl restart dovecot
$ sudo systemctl restart postfix
```

Wiadomości docierają do skrzynek pocztowych i spam jest oddzielany od zwykłej poczty e-mail.

2.2.6. Filtrowanie antywirusowe poczty – ClamAV

Agent Postfix umożliwia przesłanie wiadomości e-mail do filtra, a następnie jej ponowne zakolejkowanie. Istnieją dwa rodzaje milterów Postfix — te, które bazują wyłącznie na demonie smtpd oraz te, które nie wymagają protokołu SMTP. Poniej zaprezentowano technikę wdrożenia filtra bazującego na demonie smtpd, która została skonfigurowana na serwerze pocztowym.

sieć → smtpd → filtr → smtpd → dostarczenie

Tak w uproszczeniu wygląda proces, który jest opisany poniżej.

Instalacja skanera ClamAV w dystrybucji CentOS na serwerze poczty

Instalację ClamAV wykonano używając poniższego polecenia:

```
$ sudo yum install -y clamav-scanner clamav-update clamav-server-systemd clamav-milter-systemd sendmail-milter clamav-milter
```

Usunięto wyraz Example i wprowadzono następujące parametry wewnątrz pliku `/etc/mail/clamav-milter.conf`:

```
MilterSocket /var/run/clamav-milter/clamav-milter.socket
MilterSocketGroup mail
MilterSocketMode 660
ClamdSocket unix:/var/run/clamd.scan/clamd.sock
OnInfected Accept
AddHeader Add
ReportHostname poczta.przyklad.com
```

W powyższym fragmencie definiowane są gniazda aplikacji clamav-milter, a także gniazdo służące do komunikacji ze skanerem (clamd). Zwrócić należy uwagę na opcję **OnInfected**. Administrator powinien umieścić tu wartość **Quarantine** (kwarantanna). Opcja OnInfected Quarantine wysyła podejrzane pliki do kwarantanny.

Wykonano następujące kroki, aby clamav-milter mógł zapisywać do folderu /var/run/clamav-milter swój socket

```
$ sudo chmod +x /etc/rc.d/rc.local
$ sudo nano /etc/rc.d/rc.local
```

Dodano wpis na końcu pliku:

```
chmod u=rwx,g=rx,o=rx /var/run/clamav-milter/
```

Powoduje to zmianę uprawnień po restarcie serwera, ponieważ zmieniane są one na takie, które uniemożliwiaja zapis do tego folderu.

Dokonano edycji pliku konfiguracyjnego skanera /etc/clamd.d/scan.conf i również tutaj usunięto wyraz Example oraz wstawiono następujący wiersz:

```
LocalSocket /var/run/clamd.scan/clamd.sock
```

W następnej kolejności dodano użytkowników postfix i clamilt do grupy mail.

```
$ sudo usermod –aG mail postfix && sudo usermod –aG mail clamilt
```

Umieszczono informacje na temat gniazda smtpd_milter w pliku /etc/postfix/main.cf, przepisując do niego następujący fragment:

```
milter_default_action = accept
smtpd_milters = unix:/var/run/clamav-milter/clamav-milter.socket
```

Na koniec dokonano korekty zasady SELinux. Aby poniższy fragment kodu zadziałał, zainstalowano pakiet `policycoreutils-python` za pomocą polecenia:

```
$ sudo yum install policycoreutils-python
```

W tym przykładzie wykorzystano plik docelowego wpisu do wygenerowania pliku pakietu zasad, a następnie wczytano go do mechanizmu SELinux.

Stworzony został plik `clamav-write.te` w katalogu domowym `/home/user`

```
$ sudo -e clamav-write.te
```

Wstawiono do niego poniższą zawartość:

```
module clamav-write 1.0;
require {
type unconfined_t;
type var_run_t;
type postfix_smtpd_t;
type init_t;
class sock_file write;
class unix_stream_socket connectto;
}
#============= postfix_smtpd_t ==============
allow postfix_smtpd_t init_t:unix_stream_socket connectto;
allow postfix_smtpd_t unconfined_t:unix_stream_socket connectto;
allow postfix_smtpd_t var_run_t:sock_file write;
```

Pierwszy wiersz zawiera nazwę modułu i jego wersję. Sekcja require przechowuje różne wymagane przez nas typy i klasy. Na końcu umieszczono informacje umożliwiające demonowi smtpd łączenie się z gniazdami uniksowymi, takimi jak /run/clamav-milter/clamav-milter.socket, i zapisywanie w nich danych.

W następnej kolejności skompilowano pakiet zasad, które wczytane zostały do konfiguracji reguł SELinux. Najpierw skompilowano plik modułu zasad z poniższymi komendami:

```
$ sudo checkmodule –M -m clamav-write.mod -o clamav-write.te
```

Wraz z poleceniem checkmodule stosowane są następujące argumenty: -M włącza obsługę modułów zabezpieczeń Linuksa (ang. *Linux security modules* – LSM), -m generuje plik binarny modułu, natomiast -o definiuje nazwę wynikowego pliku.

Utworzono pakiet zasad przy użyciu następującej komendy:

```
semodule_package –o clamav-write.pp -m clamav-write.mod
```

Stworzony został pakiet zasad, przekazujący moduł (-m) poleceniu semodule_package, po czym wynik operacji został zapisany (-o) do pliku clamav.write.pp.

Można teraz wczytać ten moduł do środowiska SELinux. Wystarczy wydać instrukcję:

```
$ sudo semodule –i clamav-write.pp
```

W ten sposób zainstalowano (-i) zasadę clamav-write.pp do zabezpieczeń SELinux. Czas uruchomić usługi ClamAV.

Aby móc wystartować program antywirusowy, najpierw pobrano bazę wirusów poleceniem:

```
$ sudo freshclam
```

Po zainstalowaniu wszystkich potrzebnych pakietów aktywowano i uruchomiono demona ClamAV:

```
$ sudo systemctl enable clamd@scan && sudo systemctl start clamd@scan
```

Następnie włączono usługi clamav-milter i freshclamd (odpowiadają za aktualizowanie sygnatur wirusów):

```
$ sudo systemctl enable clamav-milter
$ sudo systemctl start clamav-milter
```

Testowanie serwera Postfix z zainstalowanym skanerem ClamAV

Po uaktywnieniu miltera ClamAV musisz sprawdzić, czy przychodzące wiadomości e-mail są analizowane pod względem obecności wirusów. Użyj do tego celu programu swaks oraz stwórz plik eicar.txt. Plik eicar.txt zawiera fragment kodu, który powinien zaalarmować skaner ClamAV.

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*
```

Przekład:

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARDOWY-PLIK-TESTOWY-ANTYWIRUSA!$H+H*
```

Wykonanie poniższego polecenia pozwoli wysłać zainfekowaną wiadomość.

```
swaks -tls -a -au admin@sysadmin.info.pl -t abuse@sysadmin.info.pl -f admin@sysadmin.info.pl -body /home/centos/eicar.txt
```

Wiadomość ta zostanie najpierw odebrana i przetworzona przez agenta MTA, a następnie przekazana milterowi ClamAV, który będzie komunikował się ze skanerem, po czym, zależnie od konfiguracji, może zostać przesłany do aplikacji SpamAssassin. Poczta zostanie przekazana użytkownikowi admin@sysadmin.info.pl, i będzie można sprawdzić zawartość jej nagłówków. W listingu poniżej zaprezentowane są nagłówki otrzymanej wiadomości e-mail.

```
X-Spam-Checker-Version: SpamAssassin 3.4.0 (2014-02-07) on
mail.sysadmin.info.pl
X-Spam-Level: *
X-Spam-Status: No, score=-0.9 required=5.0 tests=ALL_TRUSTED,MISSING_MID
X-Mailer: swaks v20170101.0 jetmore.org/john/code/swaks/
X-Virus-Scanned: clamav-milter 0.101.1 at mail.sysadmin.info.pl
```

```
X-Virus-Status: Infected (Eicar-Test-Signature)
```

Przekład:

```
X przeskanowano w poszukiwaniu wirusów: clamav-milter 0.101.21 at
mail.sysadmin.info.pl
X Stan: Zainfekowana (sygnatura testowa Eicar)
```

Widać na powyższym listingu, że do wiadomości e-mail zostały dodane nagłówki X-Virus-Scanned i X-Virus-Status. Skaner wykrył wirusa (Infected), gdyż rozpoznał sygnaturę zgodną z wartością umieszczoną w bazie danych. Gdyby wirus nie został wykryty, nagłówek X-Virus-Status zawierałby wartość Clean (czysta).

Filtrowanie zainfekowanej poczty

Podobnie jak w przypadku nagłówka X-Spam-Status, możliwe jest użycie wiadomości zawartej w nagłówku X-Virus-Status do innego traktowania wiadomości zawierających wykryty kod wirusa. Przykładowo mogą być one przesyłane do osobnego folderu nazwanego Virus. Dokonać tego można przy pomocy używanej już wcześniej wtyczki Sieve podczas oddzielania spamu od prawidłowych wiadomości e-mail. Wystarczy wstawić poniższy fragment do pliku ~/.dovecot.sieve:

```
require ["fileinto"];
if header :contains "X-Virus-Status" "" {
fileinto "Junk";
}
if not header :contains "X-Virus-Status" "Clean" {
fileinto "Virus";
}
```

Nie należy dopuszczać zainfekowanych plików do użytkowników. Powinny być one umieszczane na serwerze pocztowym w kwarantannie. Należy w tym celu wstawić parametr OnInfected Quarantine wewnątrz pliku clamav-milter.conf. Wartość ta została zmodyfikowana wcześniej, aby testowa wiadomość dotarła na skrzynkę. Rozważniej jednak blokować tego typu wiadomości, aby zminimalizować ryzyko przypadkowego otwarcia zainfekowanego pliku.

2.2.7. Instalacja i konfiguracja Fail2ban

Poniżej opiszę, w jaki sposób zainstalować i skonfigurować poprawnie aplikację Fail2ban, którą wdrożyłem na wszystkich serwerach w mojej pracy.

Aby zainstalować Fail2Ban na CentOS 7.6, w pierwszej kolejności trzeba będzie zainstalować repozytorium EPEL (ang. *Extra Packages for Enterprise Linux*). EPEL

zawiera dodatkowe pakiety dla wszystkich wersji CentOS, jednym z tych dodatkowych pakietów jest Fail2Ban.

```
$ sudo yum install epel-release
$ sudo yum install fail2ban fail2ban-systemd
```

W następnym kroku należy zaktualizować zasady SELinux.

```
$ sudo yum update -y selinux-policy*
```

Po zainstalowaniu, będziemy musieli skonfigurować i dostosować oprogramowanie za pomocą pliku konfiguracyjnego jail.local. Plik jail.local zastępuje plik jail.conf i jest używany w celu zapewnienia bezpieczeństwa aktualizacji konfiguracji użytkownika.

Zrób kopię pliku jail.conf i zapisz go pod nazwą jail.local: zaktualizuj politykę SELinux:

```
cp -pf /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

Otwórz plik jail.local do edycji w Vim za pomocą następującego polecenia.

```
$ sudo -e /etc/fail2ban/jail.local
```

Kod pliku może składać się z wielu linii kodów, które wykonują się, aby zapobiec zablokowaniu jednego lub wielu adresów IP, ustawić czas trwania bantime, itp. Typowy plik konfiguracyjny więzienia zawiera następujące linie:

```
[DEFAULT]
ignoreip = 127.0.0.1/8
ignorecommand =
bantime = 600
findtime = 600
maxretry = 5
```

- IgnoreIP służy do ustawienia listy adresów IP, które nie będą zakazane. Lista adresów IP powinna być podana z separatorem spacji. Ten parametr jest używany do ustawienia osobistego adresu IP (jeśli istnieje dostęp do serwera ze stałego adresu IP).
- Parametr Bantime służy do ustawienia czasu trwania sekund, na które host ma zostać zbanowany.
- Findtime jest parametrem, który służy do sprawdzenia, czy host musi zostać zbanowany czy nie. Gdy host generuje maksimum w ostatnim findtime, jest on banowany.
- Maxretry jest parametrem używanym do ustawienia limitu liczby prób przez hosta, po przekroczeniu tego limitu, host jest banowany.

Dodawanie pliku więzienia (ang. *jail*), w celu ochrony SSH.

Utwórz nowy plik za pomocą edytora Vim.

```
$ sudo vi /etc/fail2ban/jail.d/sshd.local
```

Do powyższego pliku należy dodać następujące wiersze kodu.

```
[sshd]
bantime = 86400
maxretry = 5
enabled = true
port = ssh
action = firewallcmd-ipset
logpath = %(sshd_log)s
```

- Parametr enable jest ustawiony na wartość true, w celu zapewnienia ochrony, aby wyłączyć ochronę, jest ustawiony na false. Parametr filtra sprawdza plik konfiguracyjny sshd, znajdujący się w ścieżce /etc/fail2ban/filter.d/sshd.conf.
- Parametr action służy do wyprowadzenia adresu IP, który musi być zakazany za pomocą filtra dostępnego w pliku /etc/fail2ban/action.d/firewallcmd-ipset.conf.
- Parametr port można zmienić na nową wartość, np. port=2244, jak to ma miejsce w tym przypadku. W przypadku korzystania z portu 22, nie ma potrzeby zmiany tego parametru.
- Ścieżka logowania podaje ścieżkę, na której zapisany jest plik logu. Ten plik dziennika jest skanowany przez Fail2Ban.
- Maxretry służy do ustawienia maksymalnego limitu nieudanych wpisów logowania.
- Parametr Bantime służy do ustawienia czasu trwania sekund, na który host musi zostać zablokowany.

Uruchomienie usługi Fail2Ban

Jeśli jeszcze nie używasz zapory sieciowej CentOS, uruchom ją:

```
$ sudo systemctl enable firewalld
$ sudo systemctl start firewalld
```

Wykonaj poniższe plecenia, aby uruchomić Fail2Ban na serwerze.

```
$ sudo systemctl enable fail2ban
$ sudo systemctl start fail2ban
```

Śledzenie wpisów logowania

Poniższe polecenie służy do sprawdzenia, które próby zalogowania się do serwera przez post ssh nie powiodły się.

```
cat /var/log/secure | grep 'Failed password'
```

Wykonanie powyższej komendy spowoduje wyświetlenie listy nieudanych prób wprowadzenia hasła głównego z różnych adresów IP. Format wyników będzie podobny do pokazanego poniżej:

```
Feb 12 19:27:12 centos sshd[25729]: Failed password for root from 150.10.0.107 port 9074 ssh2
Feb 13 17:25:47 centos sshd[9408]: Failed password for ambroziak from 150.10.0.10 port 38118 ssh2
Feb 13 15:05:35 deb_usr sshd[1617]: Failed password for invalid user pi from 42.236.138.215 port 58182 ssh2
Feb 14 08:07:52 centos sshd[4916]: Failed password for invalid user ambroziak from 37.47.31.38 port 18135 ssh2
```

Sprawdzanie zbanowanych adresów IP przez Fail2Ban

Poniższe polecenie służy do uzyskania listy zablokowanych adresów IP, które zostały rozpoznane jako zagrożenia metodą brute force.

```
iptables -L –n
```

Sprawdzanie statusu Fail2Ban

Użyj następującej komendy, aby sprawdzić status plików jail w Fail2Ban:

```
fail2ban-client status
```

Wynik powinien być podobny do tego:

```
[root@htf ]# fail2ban-client status
Status
|- Number of jail: 1
`- Jail list: sshd
```

Usunięcie zbanowanego adresu IP

W celu usunięcia adresu IP z zablokowanej listy, parametr IPADDRESS jest ustawiony na odpowiedni adres IP, który wymaga odbanowania. Nazwa "sshd" jest nazwą więzienia, w tym przypadku jest to więzienie "sshd", które skonfigurowaliśmy powyżej. Poniższe polecenie pozwala usunąć adres IP.

```
$ sudo fail2ban-client set sshd unbanip IPADDRESS
```

Dodawanie własnego filtra w celu zwiększenia ochrony

Fail2ban umożliwia tworzenie własnych filtrów. Poniżej krótki opis konfiguracji jednego z nich.

1. Należy przejść do katalogu filter.d Fail2ban:

   ```
   $ sudo cd /etc/fail2ban/filter.d
   ```

2. Utworzyć plik wordpress.conf i dodać do niego wyrażenie regularne.

   `$ sudo -e wordpress.conf`

```
# Fail2Ban filter for WordPress
[Definition]
failregex = <HOST> - - \[(\d{2})/\w{3}/\d{4}:\1:\1:\1 -\d{4}\] "POST
/wp-login.php HTTP/1.1" 200
ignoreregex =
```

   Zapisać i zamknąć plik.

3. Dodaj sekcję WordPress na końcu pliku jail.local:

   `$ sudo -e /etc/fail2ban/jail.local`

```
[wordpress]
enabled = true
filter = wordpress
logpath = /var/log/apache2/access.log
port = 80,443
```

   W tym celu użyty zostanie domyślny ban i akcja e-mail. Inne akcje mogą być zdefiniowane przez dodanie akcji = linia.

   Zapisz i wyjdź, a następnie uruchom ponownie Fail2ban poleceniem:

   `$ sudo systemctl restart fail2ban`

4. Sprawdź również, czy Twój regex działa:

   `fail2ban-regex /var/log/apache2/access.log /etc/fail2ban/filter.d/wordpress.conf`

5. W celu ochrony poczty dodaj wykorzystaj powyższy mechanizm i utwórz pliki `dovecot-pop3imap.conf` oraz `postfix.auth.conf` w katalogu `/etc/fail2ban/filter.d`

6. Do pliku `dovecot-pop3imap.conf` wstaw poniższą konfigurację:

```
[Definition]
failregex    =    (?:    pop3-login|imap-login):    (?:Authentication
failure|Aborted  login  \(auth  failed|Aborted  login  \(tried  to  use
disabled|Disconnected \(auth failed).*rip=(?P<host>\S*),.*
ignoreregex =
```

7. Do pliku postfix.auth.conf wstaw poniższą konfigurację:

```
[Definition]
failregex = connection from unknown<HOST>
ignoreregex =
```

8. Do pliku /etc/fail2ban/jail.local dodaj poniższe wpisy

```
[postfix-auth]
Enabled = true
filter = postfix.auth
```

```
action = iptables-multiport[name=postfix,
port="http,https,smtp,ssmtp,submission,pop3,pop3s,imap,imaps,sieve",
protocol=tcp]
logpath = /var/log/maillog
maxentry = 10
findtime = 60
bantime = 3600
[dovecot-pop3imap]
enabled = true
filter = dovecot-pop3imap
port = pop3,pop3s,imap,imaps
port="pop3,pop3s,imap,imaps", protocol=tcp]
logpath = /var/log/maillog
maxretry = 10
findtime = 60
bantime = 3600
```

9. Zrestartuj fail2ban poleceniem:

```
$ sudo systemctl restart fail2ban
```

2.3. Serwer plików

Na potrzeby serwera plików wykorzystałem laptop marki DELL model Inspiron 17 5767, który posiada pamięć operacyjną RAM DDR4 o pojemności 8192 MB, dysk twardy o pojemności 2 TB, z czego 1 TB zostało wydzielone dla potrzeb serwera plików, 5400 rpm, procesor Pentium Intel Core i7 7500-U 2 rdzenie po 2,7 GHz 64-bit. BIOS UEFI marki DELL w wersji 1.2.6 został zabezpieczony hasłem przed modyfikacjami. Dysk twardy został ustawiony jako pierwszy napęd. Wyłączono bootowanie z napędów USB.

GRUB2 został zabezpieczony zgodnie z opisem w instrukcji wdrażania utwardzonego serwera.

Serwer plików ma realizować obsługę udostępniania plików i katalogów w sieci LAN oraz w razie potrzeby w Internecie. W tym celu została zainstalowana dystrybucja Fedora 29, która posiada kernel oznaczony numerem 5.0.6-200.fc29.x86_64.

2.3.1. Samba

W mojej pracy wykorzystałem Samba w wersji 4.9.5, a także ustawiłem protokół SMB w wersji 3, ponieważ protokół SMB 1 został wycofany ze względów bezpieczeństwa. Natomiast ze względu na to, że SMBv3 jest obsługiwane w Windows 10 a także na fakt, że Windows 7, w którym jest używany protokół SMBv2, traci wsparcie Microsoft, podjąłem decyzję o jego użyciu.

W dystrybucji Fedora 29 zainstalowałem pakiet Samba i skonfigurowałem demona Samba oraz NMB, aby były automatycznie uruchamiane w czasie rozruchu serwera plików.

```
$ sudo yum install samba samba-client samba-common
$ sudo systemctl enable smb.service nmb.service
```

Następnie uruchomiłem usługę Samba oraz NMB - Serwer nazw NetBIOS w celu świadczenia klientom usług nazewnictwa IP w systemie NetBIOS:

```
$ sudo systemctl start smb.service nmb.service
```

Zainstalowałem zaporę firewald. Do zapory dodałem regułę, która zezwala na działanie Samby:

```
$ sudo firewall-cmd --permanent --zone=public --add-service=samba
```

Skonfigurowałem usługę SSH i demona sshd na porcie 50687 i odblokowałem powyższy port w zaporze:

```
$ sudo firewall-cmd --zone=public --add-port=50687/tcp –permanent
```

Zainstalowałem narzędzia do zarządzania portami i usługami w SELinux poleceniem:

```
$ sudo yum install policycoreutils-python policycoreutils-
python-utils
```

Następnie dodałem port 50687 do zestawu reguł dla SSH w SELinux:

```
$ sudo semanage port -a -t ssh_port_t -p tcp 50687
```

Zabezpieczyłem logowanie SSH para kluczy RSA, wyłączyłem logowanie dla root oraz za pomocą hasła.

Zdecydowałem się na instalację serwera w trybie zwykłego serwera plików. Instalację oraz konfigurację zabezpieczeń Samba opisałem w instrukcji wdrażania utwardzonego serwera.

Aspekty związane z bezpieczeństwem, np. stosowanie Samby na serwerze na stałe przyłączonym do Internetu bez maskowania adresów IP jest niemal zaproszeniem dla hakerów. Administrator powinien mieć świadomość takich zagrożeń i postępować z rozwagą, zwłaszcza tam, gdzie chodzi o wartościowe dane.

2.4. Instrukcja wdrażania utwardzonego serwera (od etapu instalacji do chwili oddania do użytkowania).

Instalacja systemu

Instalacja systemu sprowadza się do pobrania z obrazu systemu ze strony twórcy danej dystrybucji i nagranie obrazu na płytę lub pendrive. W internecie jest bardzo dużo

poradników na ten temat, które w przystępny sposób opisują, w jaki sposób wykonać powyższą procedurę. Należy podłączyć pendrive do portu USB lub włożyć płytę do napędu z instalacją systemu.

Podczas uruchomienia serwera wystarczy przytrzymać klawisz F12 i wybrać odpowiednie źródło, na którym znajduje się nagrany obraz dystrybucji, którą trzeba zainstalować. Następnie trzeba wybrać opcję: trubleshooting, a potem wybrać opcję install system in basic graphic mode (zainstaluj system w podstawowym trybie graficznym) i wcisnąć enter. Trzeba poczekać, aż wystartuje instalator. Jako język i kraj wybierać angielski oraz USA (ważne ze względu na układ klawiatury QWERTY). Ustawić strefę czasową na swój kraj, w tym przypadku Europa, Warszawa. Podłączyć sieć i skonfigurować ją, jeśli jest taka możliwość, najlepiej ręcznie ustawić statyczny adres IP, maskę, bramę i DNS-y. Powinno się wykonać ręczne partycjonowanie systemu. Dobrze jest wybrać system zarządzania pamięcią LVM, ale można inny. Wybór systemu do zarządzania przestrzenią pamięci masowej zależy od tego, jaka jest wymagana konfiguracja dysku twardego. Zalecane jest ustawienie osobno partycji dla `/, /home, /boot, /var, /tmp, /usr, pliku wymiany swap`. System plików `xfs` zajmuje mniej miejsca niż `ext4` i jest o ponad 50% szybszy od `ext4`, czy `btrfs`.
Zalecany przykładowy rozmiar partycji ustawić można w ten sposób:
/boot - 512 MB,
/biosboot - 1024 KB
swap - 6GB (dwukrotność lub trzykrotność RAM),
/ - 30 GB
/usr - 30 GB
/tmp - 30 GB
/var - 30 GB
/home - 100GB (tutaj warto po prostu ustawić taki rozmiar, jaki pozostał do dyspozycji, można wpisać słowo max, ale bezpieczniej jest wpisać rozmiar). Minimalne rozmiary partycji są opisane na poniższej stronie: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/installation_guide/s2-diskpartrecommend-x86. Po uruchomieniu instalatora z płyty lub pendrive postępować zgodnie ze wskazówkami. Należy utworzyć hasło dla użytkownika root oraz konto zwykłego użytkownika, którego można od razu dodać do grupy administratorów (sudoers). Należy wybrać instalację minimalną, co pozwala w pełni dostosować w późniejszym etapie serwer do wymaganych potrzeb. Po zakończonej instalacji należy zrestartować serwer.

Konfiguracja SSH
Należy Zalogować się na serwerze jako root lub użytkownik z uprawnieniami sudo.
W przypadku dystrybucji z rodziny Debian skonfigurować AppArmor.

W przypadku dystrybucji z rodziny Red Hat skonfigurować SELinux
```
$ sudo -e /etc/selinux/config
```
Sprawdzić, czy jest w trybie enforcing
Uruchomić aktualizację systemu:
```
$ sudo yum update
```
lub
```
$ sudo apt update && sudo apt upgrade
```
Należy edytować plik /etc/ssh/sshd_config i zmienić port z 22 na np. 2244. Ustawić PermitRootLogin prohibit-password, aby zablokować logowanie root.-a
```
$ sudo systemctl enable firewalld
```

```
$ sudo systemctl start firewalld
$ sudo firewall-cmd --state
$ sudo $ sudo firewall-cmd --zone=public --add-port=2244/tcp --permanent
$ sudo firewall-cmd --reload
$ sudo firewall-cmd --list-all
$ sudo yum install policycoreutils-python policycoreutils-python-utils
$ sudo semanage port -a -t ssh_port_t -p tcp 2244
$ sudo systemctl enable sshd.service
$ sudo systemctl start sshd.service
$ sudo systemctl status sshd.service
```

Należy wygenerować klucze rsa na serwerze za pomocą polecenia:

```
$ sudo ssh-keygen -t rsa -b 4096 -C "user@przyklad.pl"
```

(nazwa użytkownika @ serwer, na którym generowane są klucze)

Następnie trzeba przekopiować klucz do docelowej maszyny np.

```
$ sudo ssh-copy-id user@przyklad.pl -p 2244
```

Program poprosi następnie o hasło logowania użytkownika user, co wynika to z faktu, że klucz jeszcze nie został umieszczony we właściwym pliku na zdalnym serwerze. Po prawidłowym uwierzytelnieniu narzędzie ssh-copy-id przeniesie klucz publiczny do odpowiedniej lokacji.

Należy wyłączyć w sshd_config uwierzytelnianie hasłem i możliwość logowania bez hasła, a włączyć uwierzytelnianie kluczem.

```
PubkeyAuthentication yes
PasswordAuthentication no
PermitEmptyPasswords no
```

Od tej pory można łączyć się w sposób bezpieczny zdalnie z serwerem.

Konfiguracja hasła dla GRUB2

Aby zdefiniować hasło dla GRUB2 należy wpisać następującą komendę:

```
sudo grub2-mkpasswd-pbkdf2 (Centos/Fedora) lub
sudo grub-mkpasswd-pbkdf2 (Debian)
```

Zostaniemy poproszeni o wpisanie hasła użytkownika, a następnie zdefiniowanie hasła jedynie dla konta root. Dobrą praktyką, którą powinno się stosować jest dbałość o to, aby nie używać zwykłych nazw kont administracyjnych dla administratora superużytkownika GRUB2. Staramy się unikać używania popularnych nazw kont administracyjnych, takich jak root, administrator lub administrator dla konta superużytkownika GRUB2.Zaleca się, aby hasło konta administratora programu ładującego było inne niż główne dane uwierzytelniające.

Mamy teraz zaszyfrowane hasło, które musimy ustawić w głównym pliku konfiguracyjnym GRUB2 Bootloader, czyli grub.cfg. Pamiętaj o tym, by nie dodawać ręcznie konta administratora i hasła do pliku grub.cfg, ponieważ polecenie grub2-mkconfig nadpisuje ten plik. Trzeba w tym przypadku skopiować zaszyfrowane hasło do niestandardowego menu GRUB2, tj. 40_custom, które znajduje się w /etc/grub.d/.

Przed edycją pliku menu 40_custom zalecamy wykonanie najpierw kopii zapasowej tego pliku. W dalszej części użyjemy nazwy konta superużytkownika.

```
sudo cp 40_custom 40_custom.old
sudo -e 40_custom
```

Dodajemy użytkownika user oraz nasz wygenerowany hash w poniższy sposób:

```
set superusers="user"
password_pbkdf2 user
grub.pbkdf2.sha512.10000.4B141EF28E27B77E9BE3416B6303F5A0DF4CFA9010AC1A357EC9
25F62C542DF742733C21881FEE9BB1A506B3D682E0941343EC1631A0277F4B798539F6F2F90E.
2D4F33E211D91D299CD732F808FBF17EE259DD27CC4C55B686A0FE5C5CEAC244AB6208D6C6295
1A7A0DAE2824926D975CFD2DC21772FB0DB627762BD7AFA6AF1
```

Wchodzimy do katalogu /boot/grub (Debian) lub /boot/grub2 (CentOS). Wykonujemy kopię zapasową pliku grub.cfg a następnie aktualizujemy GRUB grub2-mkconfig (CentOS/Fedora) lub grub-mkconfig (Debian), pamiętając o ścieżce /boot/grub2 lub /boot/grub.

```
sudo cd /boot/grub2/
sudo cp grub.cfg grub.cfg.old
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-957.5.1.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-957.5.1.el7.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-957.el7.x86_64
Found initrd image: /boot/initramfs-3.10.0-957.el7.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-50571b3c544a458b90c7055fea34bc1f
Found initrd image: /boot/initramfs-0-rescue-
50571b3c544a458b90c7055fea34bc1f.img
done
```

W sytuacji, gdy mamy do czynienia z UEFI plik `grub.cfg` znajduje się w lokalizacji: `/boot/efi/EFI/nazwa_systemu/`, w tym przypadku `/boot/efi/EFI/fedora/`.

Pamiętać należy o tym, że program grub2-mkconfig nie ma jeszcze wbudowanej obsługi generowania plików konfiguracyjnych z uwierzytelnianiem. W przypadku dystrybucji Debian oraz CentOS przy każdym starcie system prosi o podanie loginu i hasła do GRUB. Rozwiązaniem jest edycja pliku `/etc/grub.d/10_linux` i dokonanie w nim zmian poprzez dodanie opcji `unrestricted` w poniższych paramtrach:

```
CLASS="--class gnu-linux --class gnu --class os --unrestricted
echo "submenu '$(gettext_printf "Advanced options for %s" "${OS}" |
grub_quote)' --unrestricted \$menuentry_id_option 'gnuLinuxdvanced-
$boot_device_id' {"
```

Następnie należy wykonać polecenie:

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

lub

```
sudo grub-mkconfig -o /boot/grub/grub.cfg
```

Przy próbie edycji menu proszeni jesteśmy o podanie loginu i hasła.

Konfiguracja serwera poczty w systemie CentOS 7.6 od podstaw

POSTFIX

1. Ustaw SELinux w trybie Permissive na czas konfiguracji
   ```
   sudo setenforce 0
   ```
2. Zainstaluj Postfix, Dovecot, Apache, MariaDB oraz PHP
   ```
   sudo yum update && sudo yum install postfix dovecot dovecot-mysql httpd
   httpd-devel mariadb mariadb-server php
   ```
3. Włącz uruchamianie przy starcie Apache oraz MariaDB
   ```
   sudo systemctl enable httpd mariadb
   ```
4. Uruchom Apache oraz MariaDB
   ```
   sudo systemctl start httpd mariadb
   ```
5. Uruchom polecenie: mysql_secure_installation
6. Ustaw hasło root i zatwierdź każdą opcję literą Y.
7. Zaloguj się do bazy danych MariaDB
   ```
   sudo mysql -u root -p
   ```
8. Utwórz bazę danych o nazwie serwerpoczty poleceniem:
   ```
   CREATE DATABASE serwerpoczty;
   ```

9. Wykonaj polecenia poniżej. Tworzymy specjalnego usera o nazwie dbadm z pełnymi uprawnieniami, aby sobie z niego korzystać zamiast root do połączeń z bazą danych serwerpoczty.
```
CREATE USER 'dbadm'@'localhost' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON serwerpoczty.* TO 'dbadm'@'localhost';
FLUSH PRIVILEGES;
```
10. Przełącz się na tę bazę danych, aby móc jej używać:
```
use serwerpoczty;
```
11. Utwórz poniższe tabele o nazwie Domeny, Userzy oraz Aliasy.
    Dla domen, które będą odbierać pocztę na serwerze
```
CREATE TABLE `serwerpoczty`.`Domeny` ( `IdDomeny` INT NOT NULL AUTO_INCREMENT , `NazwaDomeny` VARCHAR(50) NOT NULL , PRIMARY KEY (`IdDomeny`)) ENGINE = InnoDB DEFAULT CHARSET=utf8;
```
    Dla wszystkich kont e-mail
```
CREATE TABLE `Userzy` (
 `IdUsera` INT NOT NULL AUTO_INCREMENT,
 `IdDomeny` INT NOT NULL,
 `haslo` VARCHAR(200) NOT NULL,
 `Email` VARCHAR(200) NOT NULL,
 PRIMARY KEY (`IdUsera`),
 UNIQUE KEY `Email` (`Email`),
 FOREIGN KEY (IdDomeny) REFERENCES Domeny(IdDomeny) ON DELETE CASCADE
) ENGINE = InnoDB DEFAULT CHARSET=utf8;
```
    Dla wszystkich aliasów
```
CREATE TABLE `Aliasy` (
 `AliasId` INT NOT NULL AUTO_INCREMENT,
 `IdDomeny` INT NOT NULL,
 `Zrodlo` varchar(200) NOT NULL,
 `Cel` varchar(200) NOT NULL,
 PRIMARY KEY (`AliasId`),
 FOREIGN KEY (IdDomeny) REFERENCES Domeny(IdDomeny) ON DELETE CASCADE
) ENGINE = InnoDB DEFAULT CHARSET=utf8;
```
12. Dodaj domenę do tabeli z domenami
```
INSERT INTO Domeny (NazwaDomeny) VALUES ('sysadmin.info.pl');
```
13. Dodaj konta mail do tabeli z kontami e-mail
```
INSERT INTO Userzy (IdDomeny, haslo, Email) VALUES (1, ENCRYPT('password', CONCAT('$6$', SUBSTRING(SHA(RAND()), -16))), 'admin@sysadmin.info.pl');

INSERT INTO Userzy (IdDomeny, haslo, Email) VALUES (1, ENCRYPT('password', CONCAT('$6$', SUBSTRING(SHA(RAND()), -16))), 'info@sysadmin.info.pl');

INSERT INTO Userzy (IdDomeny, haslo, Email) VALUES (1, ENCRYPT('password', CONCAT('$6$', SUBSTRING(SHA(RAND()), -16))), 'abuse@sysadmin.info.pl');
```
14. Dodaj alias root dla konta admin
```
INSERT INTO Aliasy (IdDomeny, Zrodlo, Cel) VALUES (1, 'root@sysadmin.info.pl', 'admin@sysadmin.info.pl');
```
15. Sprawdź wyniki w tabeli z domenami, użytkownikami i aliasami
```
SELECT * FROM serwerpoczty.Domeny;
SELECT * FROM serwerpoczty.Userzy;
SELECT * FROM serwerpoczty.Aliasy;
```
16. Wyjdź poleceniem `exit` lub `quit;` z obsługi bazy danych.
17. Wykonaj kopię pliku konfiguracyjnego postfix
```
sudo cp /etc/postfix/main.cf /etc/postfix/main.cf.kopia
```

18. Edytuj główny plik konfiguracyjny postfix
    `sudo -e /etc/postfix/main.cf`
    Porównaj plik kopii z tym, co jest poniżej i sysadmin.info.pl zastąp domeną, której będziesz używać dla serwera poczty.

```
unknown_local_recipient_reject_code = 550
queue_directory = /var/spool/postfix
command_directory = /usr/sbin
daemon_directory = /usr/libexec/postfix
data_directory = /var/lib/postfix
mail_owner = postfix
myhostname = mail.sysadmin.info.pl
mydomain = sysadmin.info.pl
myorigin = $mydomain
inet_interfaces = all
inet_protocols = all
mydestination = $myhostname, localhost.$mydomain, localhost
unknown_local_recipient_reject_code = 550
mynetworks = 150.10.0.0/16, 127.0.0.0/8
relay_domains = $mydestination
relayhost =
alias_maps = hash:/etc/aliases
alias_database = hash:/etc/aliases
recipient_delimiter = +
home_mailbox = Maildir/
mailbox_transport = lmtp:unix:private/dovecot-lmtp
smtpd_banner = $myhostname ESMTP $mail_name (CentOS)
biff = no
append_dot_mydomain = no
delay_warning_time = 4h
debug_peer_level = 2
debugger_command =
 PATH=/bin:/usr/bin:/usr/local/bin:/usr/X11R6/bin
 ddd $daemon_directory/$process_name $process_id & sleep 5
sendmail_path = /usr/sbin/sendmail.postfix
newaliases_path = /usr/bin/newaliases.postfix
mailq_path = /usr/bin/mailq.postfix
setgid_group = postdrop
html_directory = no
manpage_directory = /usr/share/man
sample_directory = /usr/share/doc/postfix-2.10.1/samples
readme_directory = no
# limit an email size for 10M
#message_size_limit = 10485760
# No limit in message size
message_size_limit = 0
# limit a mailbox for 1G
#mailbox_size_limit = 1073741824
#No Limit in mailbox size
mailbox_size_limit = 0
# for SMTP-Auth
smtpd_sasl_type = dovecot
smtpd_sasl_path = private/auth
smtpd_sasl_auth_enable = yes
smtpd_sasl_local_domain = $myhostname
# TLS parameters
smtpd_tls_cert_file=/etc/letsencrypt/live/mail.sysadmin.info.pl/fullchain.
pem
smtpd_tls_key_file=/etc/letsencrypt/live/mail.sysadmin.info.pl/privkey.pem
smtpd_use_tls = yes
```

```
smtp_use_tls = yes
smtpd_tls_auth_only = yes
smtp_tls_security_level = may
smtpd_tls_security_level = may
smtpd_sasl_security_options = noanonymous, noplaintext
smtpd_sasl_tls_security_options = noanonymous
smtpd_sasl_authenticated_header = yes
broken_sasl_auth_clients = yes
virtual_transport = lmtp:unix:private/dovecot-lmtp
smtpd_sender_restrictions = permit_mynetworks, permit_sasl_authenticated,
reject_non_fqdn_sender, reject_unknown_sender_domain,
reject_unknown_reverse_client_hostname, reject_unknown_client_hostname
smtpd_recipient_restrictions = permit_mynetworks,
permit_sasl_authenticated, check_policy_service unix:postgrey/socket,
check_policy_service unix:private/policyd-spf, check_client_access
hash:/etc/postfix/rbl_override, reject_unknown_helo_hostname,
reject_non_fqdn_sender, reject_unknown_sender_domain,
reject_invalid_hostname, reject_non_fqdn_hostname,
reject_unauth_destination, reject_rbl_client zen.spamhaus.org,
reject_rbl_client sbl.spamhaus.org, reject_rbl_client cbl.abuseat.org,
reject_rbl_client dul.dnsbl.sorbs.net, reject_rhsbl_reverse_client
dbl.spamhaus.org, reject_rhsbl_helo dbl.spamhaus.org, reject_rhsbl_sender
dbl.spamhaus.org, reject_unauth_pipelining, permit
smtpd_helo_required = yes
smtpd_data_restrictions = reject_unauth_pipelining
disable_vrfy_command = yes
smtpd_delay_reject = yes
#OpenDKIM and OpenDMARC
smtpd_milters      =      unix:/var/run/clamav-milter/clamav-milter.socket,
inet:127.0.0.1:8891, inet:127.0.0.1:8893
non_smtpd_milters = $smtpd_milters
milter_default_action = accept
smtpd_tls_mandatory_protocols = !SSLv2, !SSLv3
smtpd_tls_mandatory_ciphers = medium
smtpd_tls_received_header = yes
config_directory = /etc/postfix
smtpd_error_sleep_time = 1s
smtpd_hard_error_limit = 20
smtpd_soft_error_limit = 10
# Virtual domains, users, and aliases
virtual_mailbox_domains = mysql:/etc/postfix/mariadb-wirtualne-domeny.cf
virtual_mailbox_maps = mysql:/etc/postfix/mariadb-wirtualne-mapy.cf
virtual_alias_maps = mysql:/etc/postfix/mariadb-wirtualne-
aliasy.cf,mysql:/etc/postfix/mariadb-wirtualne-maile.cf
```

Uwaga! Poniższe trzy ustawienia mają szczególne znaczenie. W plikach wskazanych po znaku = skonfigurujesz dostęp Postfix do tabel Domeny, Userzy i Aliasy:

19. Utwórz dla wirtualnych domen plik

sudo -e /etc/postfix/mariadb-wirtualne-domeny.cf

wklej:

```
user = dbadm
password = uM9kZqqabRz4FRmPhhqvfdg1
hosts = localhost
dbname = serwerpoczty
query = SELECT 1 FROM Domeny WHERE NazwaDomeny='%s'
```

```
sudo -e /etc/postfix/mariadb-wirtualne-mapy.cf
```

wklej:

```
user = dbadm
password = uM9kZqqabRz4FRmPhhqvfdg1
hosts = localhost
dbname = serwerpoczty
query = SELECT 1 FROM Userzy WHERE Email='%s'

sudo -e /etc/postfix/mariadb-wirtualne-aliasy.cf
```

wklej:

```
user = dbadm
password = uM9kZqqabRz4FRmPhhqvfdg1
hosts = localhost
dbname = serwerpoczty
query = SELECT Cel FROM Aliasy WHERE Zrodlo='%s'

sudo -e /etc/postfix/mariadb-wirtualne-maile.cf
wklej:
user = dbadm
password = uM9kZqqabRz4FRmPhhqvfdg1
hosts = localhost
dbname = serwerpoczty
query = SELECT Email FROM Userzy WHERE Email='%s'
```

20. Ustaw uprawnienia dla plików

```
sudo chmod 640 /etc/postfix/mariadb-wirtualne-domeny.cf
sudo chmod 640 /etc/postfix/mariadb-wirtualne-mapy.cf
sudo chmod 640 /etc/postfix/mariadb-wirtualne-aliasy.cf
sudo chmod 640 /etc/postfix/mariadb-wirtualne-maile.cf
```

lub

```
chmod -R o-rwx /etc/postfix/mariadb-wirtualne* (szybsza opcja)
```

21. Zmień uprawnienia na użytkownika root i grupę postfix, aby ten ostatni mógł mieć do tych czterech plików dostęp. Wykonaj polecenia z sudo.

```
chown root:postfix /etc/postfix/mariadb-wirtualne-domeny.cf
chown root:postfix /etc/postfix/mariadb-wirtualne-mapy.cf
chown root:postfix /etc/postfix/mariadb-wirtualne-aliasy.cf
chown root:postfix /etc/postfix/mariadb-wirtualne-maile.cf
```

lub

```
chown root:postfix /etc/postfix/mariadb-wirtualne* (szybsza opcja)
```

22. Zrestartuj usługi postfix, dovecot oraz mariadb

```
sudo systemctl restart mariadb.service
sudo systemctl restart dovecot.service
sudo systemctl restart postfix.service
```

23. Przetestuj i wykonaj po każdej zmianie w konfiguracji dla pewności
    Użyj polecenia sudo, aby wykonać poniższe komendy. Dla każdej wartości powinno podać cyfrę 1

```
postmap -q sysadmin.info.pl mysql:/etc/postfix/mariadb-wirtualne-domeny.cf
postmap -q admin@sysadmin.info.pl mysql:/etc/postfix/mariadb-wirtualne-mapy.cf
postmap -q info@sysadmin.info.pl mysql:/etc/postfix/mariadb-wirtualne-mapy.cf
postmap -q abuse@sysadmin.info.pl mysql:/etc/postfix/mariadb-wirtualne-mapy.cf
postmap -q admin@sysadmin.info.pl mysql:/etc/postfix/mariadb-wirtualne-maile.cf
postmap -q root@sysadmin.info.pl mysql:/etc/postfix/mariadb-wirtualne-aliasy.cf
```

24. Wykonaj kopię pliku master dla postfix

```
sudo cp /etc/postfix/master.cf /etc/postfix/master.cf.kopia
```

25. Edytuj:

```
sudo -e /etc/postfix/master.cf
```

    Porównaj plik z tym, co jest poniżej (w żadnym wypadku nie kasuj i nie podmieniaj, jedynie sprawdź wartości i odkomentuj i dodaj brakujące wartości tak, jak jest poniżej).

```
================================================================
```

```
# service type private unpriv chroot wakeup maxproc command + args
# (yes) (yes) (yes) (never) (100)
==========================================================
smtp inet n - n - - smtpd
#smtp inet n - - - 1 postscreen
#smtpd pass - - - - - smtpd
#dnsblog unix - - - - 0 dnsblog
#tlsproxy unix - - - - 0 tlsproxy
submission inet n - - - - smtpd
 -o syslog_name=postfix/submission
 -o smtpd_tls_security_level=encrypt
 -o smtpd_sasl_auth_enable=yes
 -o smtpd_sasl_type=dovecot
 -o smtpd_sasl_path=private/auth
 -o smtpd_reject_unlisted_recipient=no
 -o smtpd_client_restrictions=permit_sasl_authenticated,reject
 -o milter_macro_daemon_name=ORIGINATING
smtps inet n - - - - smtpd
 -o syslog_name=postfix/smtps
 -o smtpd_tls_wrappermode=yes
 -o smtpd_sasl_auth_enable=yes
 -o smtpd_sasl_type=dovecot
 -o smtpd_sasl_path=private/auth
 -o smtpd_client_restrictions=permit_sasl_authenticated,reject
 -o milter_macro_daemon_name=ORIGINATING
 ...
```

pod scache w sekcji maildrop dodaj: (na razie to zakomentowałem) jest to powiązane z opcją dovecot_destination_recipient_limit = 1 w main.cf postfixa, co powoduje, że mail wysłany do wielu jest rozdzielany i dostarczany po jednym naraz, co może powodować problemy.

```
dovecot unix - n n - - pipe
flags=DRhu   user=wpoczta:wpoczta   argv=/usr/libexec/dovecot/deliver   -f
${sender} -d ${recipient}
sudo systemctl restart postfix
```

DOVECOT

1. Zrób kopię zapasową dla dovecot.

```
sudo cp /etc/dovecot/dovecot.conf /etc/dovecot/dovecot.conf.kopia
sudo cp /etc/dovecot/conf.d/10-mail.conf /etc/dovecot/conf.d/10-
mail.conf.kopia
sudo cp /etc/dovecot/conf.d/10-auth.conf /etc/dovecot/conf.d/10-
auth.conf.kopia
sudo cp /etc/dovecot/conf.d/auth-sql.conf.ext /etc/dovecot/conf.d/auth-
sql.conf.ext.kopia
sudo cp /etc/dovecot/conf.d/10-master.conf /etc/dovecot/conf.d/10-
master.conf.kopia
sudo cp /etc/dovecot/conf.d/10-ssl.conf /etc/dovecot/conf.d/10-
ssl.conf.kopia
```

2. Edytuj plik konfiguracyjny dovecot

```
sudo -e /etc/dovecot/dovecot.conf
```

Zwróć uwagę na to, czy te linie są odkomentowane. (Usuń znak # przed nimi) Podpowiedź: użyj / aby wyszukać te pozycje jedna po drugiej.

```
protocols = imap pop3 lmtp
listen = * (gwiazdka tylko dla ipv4)
!include conf.d/*.conf
!include_try local.conf
#!include_try   /usr/share/dovecot/protocols.d/*.protocol   (specific   for
Debian)
```

3. Edytuj plik konfiguracyjny dovecot
   ```
   sudo -e /etc/dovecot/conf.d/10-mail.conf
   ```
   Ustaw element w pliku tak, jak wg poniższego wzorca:
   ```
   mail_location = maildir:/home/wpoczta/%d/%n/Maildir
   namespace inbox {
    inbox = yes
    mailbox Drafts {
    special_use = \Drafts
    auto = subscribe
    }
    mailbox Junk {
    special_use = \Junk
    auto = create
    }
    mailbox Virus {
    special_use = \Junk
    auto = no
    }
    mailbox Spam {
    special_use = \Junk
    auto = no
    }
    mailbox Trash {
    special_use = \Trash
    auto = subscribe
    }
    mailbox Sent {
    special_use = \Sent
    auto = subscribe
    }
    mailbox "Sent Mail" {
    special_use = \Sent
    auto = no
    }
    mailbox "Sent Messages" {
    special_use = \Sent
    auto = no
    }
    mailbox Archive {
    special_use = \Archive
    auto = create
    }
    mailbox "Archives" {
    special_use = \Archive
    auto = no
    }
   }
   mail_privileged_group = mail
   mbox_write_locks = fcntl
   ```
4. Utwórz katalog
   ```
   sudo mkdir -p /home/wpoczta/sysadmin.info.pl
   ```
5. Dodaj grupę wpoczta
   ```
   sudo groupadd -g 5000 wpoczta
   ```
6. Dodaj użytkownika wpoczta
   ```
   sudo useradd -g wpoczta -u 5000 wpoczta -d /home/mail/
   ```
7. Nadaj uprawnienia dla użytkownika i grupy wpoczta
   ```
   sudo chown -R wpoczta:wpoczta /home/wpoczta
   sudo chown -R wpoczta:wpoczta /home/wpoczta/*
   ```

```
sudo chmod -R g+w /home/wpoczta
```

8. W /etc/dovecot/conf.d/10-auth.conf włącz tylko uwierzytelnianie za pomocą SQL i pozostaw inne metody uwierzytelniania w spokoju.
```
sudo -e /etc/dovecot/conf.d/10-auth.conf
```
Wyszukaj i w razie potrzeby odkomentuj usuwając znak #
```
disable_plaintext_auth = yes
auth_mechanisms = plain login (tutaj dodaj opcję login po plain)
!include auth-system.conf.ext
!include auth-sql.conf.ext
```
9. Edytuj plik auth-sql.conf.ext i ustaw tak, jak poniżej.
```
sudo -e /etc/dovecot/conf.d/auth-sql.conf.ext
passdb {
 driver = sql
 args = /etc/dovecot/dovecot-sql.conf.ext
}
userdb {
 driver = static
 args = uid=wpoczta gid=wpoczta home=/home/wpoczta/%d/%n/Maildir
}
```
Uwaga! Zakomentuj całą sekcję z driver = sql jak poniżej.
```
#userdb {
# driver = sql
# args = /etc/dovecot/dovecot-sql.conf.ext
#}
```
10. Edytuj plik dovecot-sql.conf.ext i wklej poniższą zawartość
```
sudo -e /etc/dovecot/dovecot-sql.conf.ext
driver = mysql
connect = host=localhost dbname=serwerpoczty user=dbadm
password=uM9kZqqabRz4FRmPhhqvfdg1
default_pass_scheme = SHA512-CRYPT
password_query = SELECT Email as User, haslo FROM Userzy WHERE Email='%u';
```
11. Zmień uprawnienia do katalogu dovecot dla użytkownika wpoczta i grupy dovecot.
```
sudo chown -R wpoczta:dovecot /etc/dovecot
sudo chmod -R o-rwx /etc/dovecot
```
12. Edytuj plik 10-master.conf i ustaw jak poniżej na przykładzie.
```
sudo -e /etc/dovecot/conf.d/10-master.conf
service imap-login {
 inet_listener imap {
 port = 0
 #port = 143
 }
 inet_listener imaps {
 port = 993
 ssl = yes
 }
}
service pop3-login {
 inet_listener pop3 {
 port = 0
 #port = 110
 }
 inet_listener pop3s {
 port = 995
 ssl = yes
 }
}
service lmtp {
 unix_listener /var/spool/postfix/private/dovecot-lmtp {
```

```
 #mode = 0666i
 mode = 0600
 user = postfix
 group = postfix
 }
}
service auth {
 unix_listener /var/spool/postfix/private/auth {
 mode = 0660
 user = postfix
 group = postfix
 }
 unix_listener auth-userdb {
 mode = 0600
 user = wpoczta
 }
 user = dovecot
}
service auth-worker {
 user = wpoczta
}
```

13. Edytuj plik 10-ssl.conf odkomentuj ssl = required i zastąp klucze tym, co poniżej
```
sudo -e /etc/dovecot/conf.d/10-ssl.conf
ssl = required
ssl_cert = </etc/letsencrypt/live/mail.sysadmin.info.pl/fullchain.pem
ssl_key = </etc/letsencrypt/live/mail.sysadmin.info.pl/privkey.pem
```
14. Edytuj plik 10- logging.conf
```
sudo -e /etc/dovecot/conf.d/10-logging.conf
log_path = /var/log/dovecot.log
```
15. Następnie po zapisaniu zmiany zmień dostęp do pliku log dla dovecot.
```
sudo touch /var/log/dovecot.log
sudo chown wpoczta:dovecot /var/log/dovecot.log
sudo chmod 660 /var/log/dovecot.log
```
16. Zainstaluj i wygeneruj klucze letsencrypt
```
sudo yum install certbot python2-certbot-apache
certbot certonly --standalone -d mail.sysadmin.info.pl
crontab -e
```
dodaj to do crontab:
```
0   0,12   *   *   *   python   -c   'import   random;   import   time;
time.sleep(random.random() * 3600)' && certbot renew
```
To powoduje odświeżanie certyfikatu co 12 godzin, czyli zgodnie z zasadami let’s encrypt. https://letsencrypt.org/docs/rate-limits/ .
Nie zrobi niczego, dopóki certyfikat nie zostanie odnowiony lub unieważniony, ale regularne uruchamianie go da szansę korzystania z OnApp CP w trybie online w przypadku, gdy z jakiegoś powodu nastąpi odwołanie zainicjowane przez szyfrowanie.
17. Zainstaluj mailx do wysłania maila testowego, a potem usuń mailx
```
sudo yum install mailx
sudo mail admin@sysadmin.info.pl
sudo mail abuse@sysadmin.info.pl
Wpisz temat
```
Wciśnij ctrl+d
```
sudo yum remove mailx
```
18. Sprawdź w logach informacje o prawidłowym przesłaniu maila.
```
sudo tail -f /var/log/maillog /var/log/dovecot.log
```
19. Dodaj reguły dla portów w usłudze firewalld
```
sudo firewall-cmd --permanent --add-port=587/tcp
```

```
sudo firewall-cmd --permanent --add-service=smtp
sudo firewall-cmd --permanent --add-service=smtps
sudo firewall-cmd --permanent --add-service=imap
sudo firewall-cmd --permanent --add-service=imaps
sudo firewall-cmd --permanent --add-service=pop3
sudo firewall-cmd --permanent --add-service=pop3s
sudo firewall-cmd --state - czy jest włączony
sudo firewall-cmd –start – start/stop
sudo firewall-cmd –reload – przeładowuje ustawienia firewalld
sudo firewall-cmd --list-all - listowanie portów i serwisów
```

20. Zainstaluj sieve poleceniem:
    ```
    $ sudo yum install dovecot-pigeonhole
    ```
    Wyedytuj plik
    ```
    sudo -e /etc/dovecot/conf.d/10-auth.conf
    ```
    Ustaw wartość jak poniżej:
    ```
    auth_username_format = %Lu
    ```
21. Wyedytuj plik
    ```
    sudo -e /etc/dovecot/conf.d/20-lmtp.conf
    ```
    Dodaj poniższe ustawienia:
    ```
    protocol lmtp {
          postmaster_address = info@sysadmin.info.pl
          mail_plugins = quota sieve
    }
    ```
22. Sprawdź czy SELinux jest w trybie permissive
    ```
    sudo getenforce
    ```
23. Włącz tryb enforcing
    ```
    sudo setenforce 1
    ```
24. Na sam koniec zmień ustawienia SELinux i daj dostęp do katalogu domowego, ponieważ inaczej postfix ani dovecot nie będą mieć do niego dostępu i nie będą w stanie zapisywać i odczytywać z niego maili.
    ```
    sudo restorecon -r /home
    ```

Konfiguracja serwera WWW w systemie Debian 9.8.0

Serwer bazy danych – serwer z CentOS 7.6, na którym jest zainstalowana baza danych.
Server www: Serwer z Debian 9.8.0, na którym jest zainstalowany WordPress.
baza_wp: Nazwa bazy danych.
wp_uzytk: Użytkownik – klient bazy danych WordPress
ElF@9GoSvshO0Fn4P&MN: hasło użytkownika bazy danych SQL – baza_wp.
150.10.0.11: Prywatny adres IP serwera bazy danych.
150.10.0.10: Prywatny adres IP serwera www.
deb_usr: Lokalny użytkownik z prawami sudo, który nie jest rootem.
176.105.137.72/sysadmin.info.pl: Publiczny adres serwera , lub nazwa domeny (FQDN).

1. Zainstaluj serwer bazy danych MariaDB 10.3 w CentOS za pomocą polecenia:
   ```
   sudo -e /etc/yum.repos.d/MariaDB.repo
   ```
   wklej tę zawartość:

```
# MariaDB 10.3 CentOS repository list - created 2019-03-02 11:00 UTC
# http://downloads.mariadb.org/mariadb/repositories/
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.3/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```

Zapisz insert Esc :wq Enter

2. Zainstaluj serwer MariaDB oraz klienta:
```
sudo yum install MariaDB-server MariaDB-client
```

3. Wystartuj MariaDB serwer
```
sudo systemctl start mariadb
```
4. Włącz na stałe serwer MariaDB
```
sudo systemctl enable mariadb
```
5. Sprawdź status usługi MariaDB
```
sudo systemctl status mariadb
```
6. Uruchom obsługę MariaDB poleceniem, ponieważ nie masz nadanego hasła użytkownika root bazy danych.
```
sudo mysql -u root
```
7. Po zalogowaniu się ustal hasło roota do MariaDB poleceniem:
```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
quit;
```
8. Zaloguj się do MariaDB wcześniej ustalonym hasłem:
```
mysql -u root -p
```
9. Sprawdź status MariaDB:
```
status;
quit;
```
10. Wykonaj poniższe polecenie:
```
mysql_secure_installation
```
11. Wyświetli się poniższe okno, w którym na pytanie o zmianę hasła root odpowiadasz literą n, ponieważ ustalone zostało wcześniej. Pozostałe pozycje potwierdzasz drukowaną literą Y.
12. W CentOS firewall jest zainstalowany domyślnie. Sprawdź, czy jest włączony.
```
firewall-cmd --state
```
13. Listowanie portów i serwisów firewall
```
firewall-cmd --list-all
```
14. Dodaj usługę mysql
```
firewall-cmd --permanent --add-service=mysql
firewall-cmd --permanent --add-port=3360/tcp
firewall-cmd --permanent --add-port=3306/tcp
```
15. Zrestartuj firewall
```
firewall-cmd --reload
```
16. Usunięcie usługi lub portu. Nie jest to potrzebne w tym momencie, ale może się przydać kiedyś. Z portu 3360 korzysta MariaDB oraz MySQL. Jeśli chcesz dodać ten port zamiast usługi, co czasem może być rozwiązaniem problemu, to zastąp remove słowem add.
```
firewall-cmd --permanent --remove-port=3360/tcp
firewall-cmd --permanent --remove-service=mysql
```
17. Zaloguj się do MariaDB
```
sudo mysql -u root -p
```
18. Wykonaj poniższe polecenie aby dodać bazę danych o nazwie baza_wp
```
CREATE DATABASE baza_wp;
```
19. Wykonaj polecenie, aby utworzyć użytkownika, który będzie korzystać z tej bazy. Nie powinien być to użytkownik root ze względów bezpieczeństwa.
```
CREATE USER 'wp_uzytk’@’localhost' IDENTIFIED BY 'password';
```
20. Przydziel uprawnienia użytkownikowi wp_uzytk
```
GRANT ALL PRIVILEGES ON baza_wp.* TO 'wp_uzytk’@’localhost';
```

21. Utwórz użytkownika oraz przydziel uprawnienia do zdalnego dostępu do bazy danych baza_wp dla użytkownika wp_uzytk. Adres IP to lokalny adres IP serwera www, na którym znajduje się WordPress. Hasło jest takie samo, jak hasło użytkownika, utworzonego wyżej.
```
CREATE USER 'wp_uzytk'@'150.10.0.10' IDENTIFIED BY 'password';
GRANT SELECT, DELETE, CREATE, INSERT, UPDATE ON baza_wp.* TO `wp_uzytk`@`localhost` IDENTIFIED BY 'password';
GRANT SELECT, DELETE, CREATE, INSERT, UPDATE ON baza_wp.* TO `wp_uzytk`@`150.10.0.10` IDENTIFIED BY 'password';
```
22. Wykonaj poniższe polecenia:
```
FLUSH PRIVILEGES;
exit
```
23. Sprawdź, czy jesteś w stanie zalogować się stworzonym użytkownikiem:
```
mysql -u wp_uzytk -p
status;
exit
```
24. Na serwerze www z Debian wykonaj następujące polecenie:
```
sudo apt update && sudo apt install mariadb-client php-mysql
```
25. Sprawdź, czy możesz się zalogować przy pomocy poniższego polecenia:
```
mysql -u wp_uzytk -h 150.10.0.11 -p
```
użyj hasła: ElF@9GoSvshO0Fn4P&MN
26. Sprawdź status MariaDB
```
status;
```
27. Zamknij połączenie wychodząc z MariaDB.
```
quit;
```
28. Utwórz katalog o nazwie src w katalogu swojej witryny, aby przechowywać nowe kopie plików źródłowych WordPress. W tym przewodniku jako przykład wykorzystano katalog domowy /var/www/html/sysadmin.info.pl/. Przejdź do tego nowego katalogu:
```
sudo mkdir -p /var/www/html/sysadmin.info.pl/src/
cd /var/www/html/sysadmin.info.pl/src/
```
29. Ustaw użytkownika serwera WWW, www-data, jako właściciela katalogu domowego swojej witryny. www-data jest grupą.
```
sudo chown -R www-data:www-data /var/www/html/sysadmin.info.pl/
```
30. Zainstaluj najnowszą wersję WordPress i wypakuj ją:
```
sudo wget http://wordpress.org/latest.tar.gz
sudo -u www-data tar -xvf latest.tar.gz
```
31. Zmień nazwę pliku latest.tar.gz na wordpress, a następnie ustaw datę przechowywania kopii zapasowej oryginalnych plików źródłowych. Będzie to przydatne, jeśli zainstalujesz nowe wersje w przyszłości i będzie potrzeba powrócić do poprzedniej wersji:
```
sudo mv latest.tar.gz wordpress-`date "+%Y-%m-%d"`.tar.gz
```
32. Utwórz katalog public_html, który będzie katalogiem głównym WordPress. Przenieś pliki WordPress do folderu public_html:
```
sudo mkdir -p /var/www/html/sysadmin.info.pl/public_html/
sudo mv wordpress/* ../public_html/
```
33. Nadaj folderowi public_html uprawnienia dla grupy www-data:
```
sudo chown -R www-data:www-data /var/www/html/sysadmin.info.pl/public_html
```
34. Przejdź do katalogu, do którego wyodrębniono WordPress, skopiuj przykładową konfigurację i ustaw ją tak, aby korzystała ze zdalnej bazy danych:
```
cd /var/www/html/sysadmin.info.pl/public_html
sudo cp wp-config-sample.php wp-config.php
```
35. Zmień zmienne logowania tak, aby pasowały do bazy danych i użytkownika. Zastąp 150.10.0.11 prywatnym adresem IP serwera bazy danych. Wyedytuj plik:

```
sudo -e /var/www/html/sysadmin.info.pl/public_html/wp-config.php
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');
/** MySQL database username */
define('DB_USER', 'wp_uzytk');
/** MySQL database password */
define('DB_PASSWORD', 'haslo_użytkownika_bazy_danych');
/** MySQL hostname */
define('DB_HOST', '150.10.0.11');
```

36. Dodaj klucze bezpieczeństwa, aby zabezpieczyć wp-admin.Użyj Generatora kluczy bezpieczeństwa WordPress, aby utworzyć losowe, skomplikowane hashe, których WordPress użyje do zaszyfrowania danych logowania. Skopiuj wynik i zastąp odpowiednią sekcję w pliku wp-config.php:

```
/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {@link
https://api.wordpress.org/secret-key/1.1/salt/ WordPress.org secret-key
service}
 * You can change these at any point in time to invalidate all existing
cookies. This will force all users to have to log in again.
 *
 * @since 2.6.0
 */
define('AUTH_KEY', 'put your unique phrase here');
define('SECURE_AUTH_KEY', 'put your unique phrase here');
define('LOGGED_IN_KEY', 'put your unique phrase here');
define('NONCE_KEY', 'put your unique phrase here');
define('AUTH_SALT', 'put your unique phrase here');
define('SECURE_AUTH_SALT', 'put your unique phrase here');
define('LOGGED_IN_SALT', 'put your unique phrase here');
define('NONCE_SALT', 'put your unique phrase here');
/**#@-*/
```

37. **Zabezpieczenie ruchu do i z bazy danych WordPress za pomocą SSL**
    Domyślnie CentOS ma utworzony katalog z certyfikatami i nie trzeba żadnego katalogu tworzyć. Wejść do niego można:
    ```
    cd /etc/pki/tls/certs/
    ```
38. Wygeneruj klucz urzędu certyfikacji i utwórz certyfikat oraz klucz prywatny. Odpowiadaj na odpowiednie monity. Klucz w tym przykładzie wygasa za 100 lat. Zmień wartość dni 36500 w tym i następnych krokach, aby ustawić certyfikaty do wygaśnięcia w razie potrzeby:
    ```
    sudo openssl genrsa 4096 > ca-key.pem
    sudo openssl req -new -x509 -nodes -days 36500 -key ca-key.pem -out
    cacert.pem
    ```
    Common Name ustaw na MariaDB
39. Utwórz certyfikat serwera i zapisz klucz RSA. Nazwa zwykła (common name) powinna być nazwą FQDN lub adresem IP twojego serwera WWW. W tym przypadku: sysadmin.info.pl
    ```
    sudo openssl req -newkey rsa:4096 -days 36500 -nodes -keyout server-
    key.pem -out server-req.pem
    sudo openssl rsa -in server-key.pem -out server-key.pem
    ```
40. Podpisz certyfikat:
    ```
    sudo openssl x509 -req -in server-req.pem -days 36500 -CA cacert.pem -
    CAkey ca-key.pem -set_serial 01 -out server-cert.pem
    ```
41. Przenieś klucze i certyfikat na stałe miejsce:

```
mv *.* /etc/pki/tls/certs/ && cd /etc/pki/tls/certs/
```

42. Wygeneruj klucz klienta. Odpowiadaj na odpowiednie monity i ustaw wspólną nazwę na FQDN lub adres IP swojego serwera WWW: sysadmin.info.pl
```
sudo openssl req -newkey rsa:2048 -days 36500 -nodes -keyout client-key.pem -out client-req.pem
```
43. Zapisz jako klucz RSA
```
sudo openssl rsa -in client-key.pem -out client-key.pem
```
44. Podpisz certyfikat klienta
```
sudo openssl x509 -req -in client-req.pem -days 36500 -CA cacert.pem -CAkey ca-key.pem -set_serial 01 -out client-cert.pem
```
45. Zweryfikuj certyfikaty
```
openssl verify -CAfile cacert.pem server-cert.pem client-cert.pem
```
46. Powinien przy obu wyświetlić OK.
47. Wyedytuj plik server.cnf na CentOS
```
sudo -e /etc/my.cnf.d/server.cnf
```
48. Dodaj w sekcji [mysqld]:
```
ssl-ca=/etc/pki/tls/certs/cacert.pem
ssl-cert=/etc/pki/tls/certs/server-cert.pem
ssl-key=/etc/pki/tls/private/server-key.pem
ssl-cipher=ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-GCM-SHA384:DHE-RSA-AES128-GCM-SHA256:DHE-DSS-AES128-GCM-SHA256:kEDH+AESGCM:ECDHE-RSA-AES128-SHA256:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA:ECDHE-ECDSA-AES128-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA:ECDHE-ECDSA-AES256-SHA:DHE-RSA-AES128-SHA256:DHE-RSA-AES128-SHA:DHE-DSS-AES128-SHA256:DHE-RSA-AES256-SHA256:DHE-DSS-AES256-SHA:DHE-RSA-AES256-SHA:!aNULL:!eNULL:!EXPORT:!DES:!RC4:!3DES:!MD5:!PSK
```
49. Znajdź zakomentowaną linię #bind-address, usuń # , co spowoduje odkomentowanie i ustaw jak poniżej. Adres IP oczywiście zmień na lokalny adres IP serwera CentOS, na którym jest zainstalowana baza danych MariaDB.
```
bind-address=150.10.0.11
```
Wciśnij ctrl+o, a następnie ctrl+x
50. Przenieś plik server-key do folderu private
```
mv /etc/pki/tls/certs/server-key.pem /etc/pki/tls/private/
```
51. Zaloguj się do MariaDB i wymagaj protokołu SSL dla wszystkich logowań do bazy danych. Zastąp 150.10.0.10 prywatnym adresem IP
```
sudo mysql -u root -p
GRANT ALL PRIVILEGES ON baza_wp.* TO 'wp_uzytk'@'150.10.0.10' REQUIRE SSL;
FLUSH PRIVILEGES;
exit
```
52. Zrestartuj serwer MariaDB:
```
sudo systemctl restart mysql
```
53. Skopiuj certyfikaty i klucz do serwera WWW. Zamień użytkownika deb_usr na użytkownika serwera WWW, a 150.10.0.10 na prywatny adres IP serwera WWW:
```
scp -P 13896 cacert.pem client-cert.pem client-key.pem deb_usr@150.10.0.10:~/certs
```
54. Na serwerze www utwórz katalog i przenieś certyfikaty i klucz do /etc/mysql/ssl:
```
sudo mkdir /etc/mysql/ssl && sudo mv ~/certs/*.* /etc/mysql/ssl
```
55. Jeśli katalog /etc/mysql/ssl już istnieje, to wykonaj samo polecenie po znakach &&.
```
mv /home/deb_usr/certs/*.* /etc/mysql/ssl
```
56. Skonfiguruj klienta MariaDB serwera WWW do korzystania z SSL. Znajdź sekcję [mysql] w pliku 50-mysql-clients.cnf i dodaj lokalizacje dla certyfikatów i klucza:
```
sudo -e /etc/mysql/mariadb.conf.d/50-mysql-clients.cnf
```
57. Wklej poniższą zawartość w sekcji [mysql]
```
ssl-ca=/etc/mysql/ssl/cacert.pem
```

```
ssl-cert=/etc/mysql/ssl/client-cert.pem
ssl-key=/etc/mysql/ssl/client-key.pem
```

58. Zaloguj się z serwera www z Debian do serwera bazy danych MariaDB z CentOS przy pomocy poniższego polecenia:
```
mysql -u wp_uzytk -h 150.10.0.11 -p
```
59. Jeśli się połączy, wyświetlony zostanie wiersz zachęty MariaDB. Wpisz polecenie:
```
status;
exit
```
60. Dodaj dyrektywę przed zdalną bazą danych w wp-config, która zmusza WordPress do używania SSL do połączenia z bazą danych:
```
...
define( 'MYSQL_CLIENT_FLAGS', MYSQLI_CLIENT_SSL );
/** The name of the database for WordPress */
define('DB_NAME', 'wordpress');
/** MySQL database username */
define('DB_USER', 'wp_uzytk');
/** MySQL database password */
define('DB_PASSWORD', 'haslo_użytkownika_bazy_danych');
/** MySQL hostname */
define('DB_HOST', '150.10.0.11');
...
```
61. Uzyskaj dostęp do interfejsu instalacyjnego WordPress poprzez wp-admin. Użyj przeglądarki, aby przejść do sysadmin.info.pl/wp-admin. Jeśli połączenie z bazą danych zakończy się powodzeniem, zobaczysz ekran instalacji:
62. Utworzenie logów dla MariaDB
```
sudo mkdir /var/log/mariadb/
sudo -e /var/log/mariadb/mariadb.log
sudo chown -R mysql:mysql /var/log/mariadb/*
```
63. systemctl status mariadb – pokazuje błąd:
```
[ERROR] Incorrect definition of table mysql.event: expected column 'sql_mode' at position 14
to have type
set('REAL_AS_FLOAT','PIPES_AS_CONCAT','ANSI_QUOTES','IGNORE_SPACE','IGNORE_BAD_TABLE_OPTIONS
','ONLY_FULL_GROUP_BY','NO_UNSIGNED_SUBTRACTION','NO_DIR_IN_CREATE','POSTGRESQL','ORACLE','M
SSQL','DB2','MAXDB','NO_KEY_OPTIONS','NO_TABLE_OPTIONS','NO_FIELD_OPTIONS','MYSQL323','MYSQL
40','ANSI','NO_AUTO_VALUE_ON_ZERO','NO_BACKSLASH_ESCAPES','STRICT_TRANS_TABLES','STRICT_ALL_
TABLES','NO_ZERO_IN_DATE','NO_ZERO_DATE','INVALID_DATES','ERROR_FOR_DIVISION_BY_ZERO','TRADI
TIONAL','NO_AUTO_CREATE_USER','HIGH_NOT_PRECEDENCE','NO_ENGINE_SUBSTITUTION','PAD_CHAR_TO_FU
LL_LENGTH'), found type
set('REAL_AS_FLOAT','PIPES_AS_CONCAT','ANSI_QUOTES','IGNORE_SPACE','NOT_USED','ONLY_FULL_GRO
UP_BY','NO_UNSIGNED_SUBTRACTION','NO_DIR_IN_CREATE','POSTGRESQL','ORACLE','MSSQL','DB2','MAX
DB','NO_KEY_OPTIONS','NO_TABLE_OPTIONS','NO_FIELD_OPTIONS','MYSQL323','MYSQL40','ANSI','NO_A
UTO_VALUE_ON_ZERO','NO_BACKSLASH_ESCAPES','STRICT_TRANS_TABLES','STRICT_A
[ERROR] Event Scheduler: An error occurred when initializing system tables. Disabling the
Event Scheduler.
```
64. Rozwiązanie:
```
sudo mysql_upgrade -u root -p
```
65. Błąd:
```
Could not create the upgrade info file '/var/lib/mysql/mysql_upgrade_info'
in the MariaDB Servers datadir, errno: 13
```
66. Rozwiązanie:
```
sudo -e /var/lib/mysql/mysql_upgrade_info
insert esc :wq!
sudo chown -R mysql:mysql /var/lib/mysql/*
sudo chmod -R 777 /var/lib/mysql/mysql_upgrade_info
sudo mysql_upgrade -u root -p
```
67. po upgrade
```
sudo chmod -R 640 /var/lib/mysql/mysql_upgrade_info
```

Konfiguracja serwera plików Samba w systemie Fedora 29
Instalacja Samby

1. Zainstaluj serwer Samba 4.9.5 w Fedora 29 poleceniem:
   ```
   sudo yum install samba samba-client samba-common
   ```
   Po chwili mamy zainstalowaną usługę w komputerze. Jej systemowa nazwa to smbd i nmbd, przyda nam się to później. Przejdźmy do najważniejszej części, czyli konfiguracji.
2. Do firewall trzeba dodać regułę pozwalającą na działanie Samby
   ```
   $ sudo firewall-cmd --permanent --zone=public --add-service=samba
   $ sudo firewall-cmd –reload
   ```

Konfiguracja Samby

3. Główny plik konfiguracyjny Samby zlokalizowany jest w /etc/samba/smb.conf.
4. Należy wykonać kopię zapasową pliku Samby:
   ```
   $ sudo cp /etc/samba/smb.conf /etc/samba/smb.conf.kopia
   ```
5. Plik jest własnością użytkownika root, z tego względu wszelkie zmiany będą wymagać uprawnień administratora.
6. Konfiguracja Samby polega na konfiguracji poniższych pozycji.
   - `[global]`
     W tej sekcji znajdują się ustawienia globalne, które dotycząc wszystkich użytkowników.
   - `workgroup = WORKGROUP`
     Nazwa grupy pochodzi z systemu Microsoft Windows. Należy ustawić taką nazwę grupy, jaką mają pozostałe komputery w sieci. W przypadku domyślnej instalacji systemu Microsoft Windows w wersji polskiej, jest to: GRUPA_ROBOCZA, w przypadku wersji angielskiej WORKGROUP.
   - `server string = %h server (Samba, Fedora)`
     Pod tą nazwą jest widoczny serwer w lokalnym otoczeniu sieciowym.
   - `security = user`
     Jest to jedno z najważniejszych ustawień, które określa, kto może korzystać z serwera Samba. Gdy zostanie ustawiona opcja user, każda osoba, która będzie chciała pobrać plik z serwera, będzie musiała mieć założone na nim własne konto użytkownika, natomiast w przypadku ustawienia opcji share, nie trzeba tego robić. Należy tę opcję odkomentować (usunąć symbol #).
   - `encrypt passwords = true`
     Powyższy wpis powoduje, że hasło użytkownika jest szyfrowane.
   - `passdb backend = tdbsam`
     Ta opcja określa, w którym miejscu są zapisywane szyfrowane hasła.
   - `map to guest = bad user`
     To ustawienie powoduje ustawienie uprawnień gościa dla wszystkich osób, które nie są zalogowane do serwera Samba jako zarejestrowany użytkownik.
   - Dział Share Definitions odpowiadzialny jest za katalogi, które są udostępniane innym użytkownikom. Można w tym miejscu określić ich uprawnienia, np. atrybut tylko do odczytu.

Security share

7. W przypadku, kiedy jako administrator nie chcemy tworzyć żadnych użytkowników, a jedynie chcemy udostępnić wszystkim innym użytkownikom w sieci swoje pliki, należy w pliku konfiguracyjnym Samby ustawić opcję security na share.

8. Należy dodać lub zmienić te opcje, które zmajdują się w pliku konfiguracyjnym.
```
server string = Samba
workgroup = WORKGROUP
security = share
```
9. W następnej kolejności trzeba dodać dodatkowe opcje na początku konfiguracji w sekcji [global] tak, jak jest to przedstawione poniżej:
```
netbios name = Samba
domain master = yes
local master = yes
browseable = no
```
   - `netbios name` – to ustawienie odpowiedzialne jest za wyświetlanie nazwy serwera Samba w otoczeniu sieciowym
   - `browseable` – ta opcja odpowiedzialna jest za widoczność serwera Samba w otoczeniu sieciowym. Parametr no powoduje, że serwer w ogóle nie jest widoczny w otoczeniu sieciowym, co jest prawidłowe ze względów bezpieczeństwa.
   - `local i domain master` – potrzebne do właściwej komunikacji między serwerem, a klientami

Udostępnianie publiczne w Samba

10. Jeśli istnieje potrzeba, aby wszystkim udostępnić przykładowo folder „Anonymous”, znajdujący się w katalogu /srv/samba, należy wykonać poniższe polecenia:
```
$ sudo mkdir -p /srv/samba/anonymous
$ sudo chmod -R 0775 /srv/samba/anonymous
$ sudo chown -R nobody:nobody /srv/samba/anonymous
```
11. Należy również zmienić kontekst bezpieczeństwa SELinux dla katalogu współdzielonego samby w następujący sposób:
```
$ sudo chcon -t samba_share_t /srv/samba/anonymous
```
12. Następnie trzeba otworzyć plik konfiguracyjny Samby do edycji, w którym można modyfikować/dodawać sekcje poniżej wraz z odpowiednimi dyrektywami.
```
$ sudo -e /etc/samba/smb.conf
```
13. Należy dodać poniższy wpis na samym końcu pliku konfiguracyjnego Samby:
```
[Anonymous]
comment = Anonymous
path = /srv/samba/Anonymous
browseable = yes
writable = no
create mode = 0644
directory mode = 0755
guest ok = yes
```
   - `comment` – nazwa udostępnianego zasobu
   - `path` – ścieżka do niego
   - `browseable` – określa czy zasób ten można przeglądać
   - `writable` – określa to czy można w nim zapisywać i zmieniać pliki
   - `create mode` – określa jakie prawa dostępu mają nowo tworzone w nim pliki
   - `directory mode` – odnosi się do praw tworzonych folderów
   - `guest ok` – ważna opcja, bez niej będzie się nam ciągle pokazywało okienko logowania w Windowsie, kiedy będziemy chcieli otworzyć zasób, a i tak się nie zalogujemy, bo nie stworzyliśmy jeszcze żadnych użytkowników.

14. Teraz sprawdź aktualne ustawienia Samby, wykonując poniższe polecenie.
    ```
    $ sudo testparm
    ```
15. Następnie należy włączyć uruchamianie Samby przy starcie systemu oraz uruchomić poleceniami:
    ```
    $ sudo systemctl enable smb.service
    $ sudo systemctl enable nmb.service
    $ sudo systemctl start smb.service
    $ sudo systemctl start nmb.service
    ```
16. Po chwili serwer Samba powinien być już widoczny w otoczeniu sieciowym pod nazwą Samba. Jeżeli do niego wejdziesz, powinieneś zobaczyć w nim katalog „Anonymous”. W obecnej konfiguracji folder ten jest jednak tylko do odczytu, aby inni mogli tu coś zapisywać należy zrobić kilka zmian. Po pierwsze należy zmienić uprawnienia do katalogu na serwerze na poziomie 777, czyli:
    ```
    $ sudo chmod 777 /srv/samba/Anonymous
    ```
17. Następnie w pliku konfiguracyjnym Samby (przy opcjach dotyczących zasobu [Anonymous]) trzeba zmienić jedną linijkę, mianowicie:
    ```
    writable = yes
    ```
18. Po zrestartowaniu Samby, każdy będzie miał prawa do zapisu, modyfikacji i kasowania w nim plików.
19. Jeżeli administrator planuje udostępnić wszystkim zainstalowane na serwerze drukarki, wystarczy przy udziałach [printers] oraz [print$] zmienić:
    ```
    guest ok = yes
    ```
20. Uwaga: W niektórych dystrybucjach Linux-a należy dodatkowo odblokować porty Samby na firewall-u, jeżeli takowy jest zainstalowany.
    ```
    $ sudo firewall-cmd –zone=public –add-port=135/tcp –permanent
    $ sudo firewall-cmd –zone=public –add-port=137/udp –permanent
    $ sudo firewall-cmd –zone=public –add-port=138/udp –permanent
    $ sudo firewall-cmd –zone=public –add-port=139/tcp –permanent
    ```
21. Jeżeli są jakieś problemy w połączeniu się z Sambą, czasami warto zrestartować połączenie sieciowe klienta.

Security user

22. Teraz zobaczmy jak działa opcja security = user. W tym przypadku do przeglądania jakichkolwiek zasobów, potrzebny jest login i hasło. Zaczynamy od stworzenia nowego użytkownika w systemie Linux, w Fedora można to zrobić poprzez dodanie grupy smbgrp
    ```
    $ sudo groupadd smbgrp
    $ sudo usermod user -aG smbgrp
    ```
23. Dodajemy np. użytkownika user a jego hasło to QbAd3R4htxg&, teraz trzeba tego użytkownika dodać do bazy Samby, wpisujemy w konsoli:
    ```
    $ sudo smbpasswd -a user
    ```
24. Podajemy 2 razy hasło, czyli QbAd3R4htxg&, użytkownik został dodany, teraz idziemy do pliku konfiguracyjnego, zmieniamy oczywiście opcję:
    ```
    security = user
    ```
25. Zrestartuj Sambę poleceniami:
    ```
    $ sudo systemctl restart smb.service
    $ sudo systemctl restart nmb.service
    ```
26. Po zapisaniu konfiguracji i zrestartowaniu Samby, Windows będzie wyświetlał okienko logowania, gdy zechcemy wejść na serwer. Spróbujmy wejść więc teraz do udostępnionego katalogu, wpisujemy login i hasło, i spróbujmy coś w nim zapisać. Jeżeli katalog „Anonymous” ma uprawnienia na poziomie 777 i w pliku konfiguracyjnym mamy opcję writable = yes, to będziemy mogli to uczynić, jest jednak jedno ale. Nowo stworzony plik będzie miał za właściciela user-a, i będzie

tylko do odczytu dla innych, żeby to zmienić, należy zmienić następujące opcje w konfiguracji samby:

```
create mode = 0777
directory mode = 0777
```

Dzięki temu wszystkie tworzone pliki i katalogi będą miały domyślne uprawnienia 777.

27. Dla potrzeb bezpieczeństwa należy stworzyć foldery dla konkretnych grup użytkowników.
28. W tym celu przykładowo należy wykonać poniższe polecenia:

```
$ sudo mkdir -p /srv/samba/secure
$ sudo chmod -R 0770 /srv/samba/secure
$ sudo chown -R root:smbgrp /srv/samba/secure
$ sudo chcon -t samba_share_t /srv/samba/secure
```

Utworzony został katalog secure i nadano grupie smbgrp uprawnienia do niego. Ostatnie polecenie zmienia kontekst bezpieczeństwa SELinux dla katalogu współdzielonego Samby.

29. Należy dokonać edycji pliku konfiguracyjnego Samby poleceniem:

```
$ sudo -e /etc/samba/smb.conf
```

30. Wstawić na samym końcu poniższe parametry

```
[Secure]
   comment = Secure
   path = /srv/samba/secure
   valid users = @smbgrp
   guest ok = no
   writable = yes
   browsable = no
   create mode = 0644
   directory mode = 0755
```

- `valid users` – określa dostęp dla grupy – smbgrp. Tylko użytkownicy tej grupy będą mieć dostęp do tego folderu i prawo zapisu do niego.

31. Ponownie, sprawdź ustawienia konfiguracji samby, wykonując następującą komendę.

$ sudo testparm

32. Uruchom ponownie usługi Samba, aby zastosować zmiany.

```
$ sudo systemctl restart smb.service
$ sudo systemctl restart nmb.service
```

33. Przejdź do maszyny z systemem Windows, otwórz "Sieć" z okna Eksploratora Windows, następnie kliknij na hosta Samba, lub spróbuj uzyskać dostęp do serwera używając jego adresu IP.

\\150.10.0.12

34. Zostaniesz poproszony o podanie nazwy użytkownika i hasła do logowania na serwerze Samba. Po wprowadzeniu danych uwierzytelniających kliknij OK.

Mapowanie dysków w Windows

35. Teraz wystarczy w Windows zmapować jako dysk sieciowy udostępniany przez Sambę zasób.

2.5. Dobre praktyki administracyjne i polityka użytkowania serwera.

Polityka użytkowania serwera

1. **Aktualizacja i łaty bezpieczeństwa**
   Należy pamiętać o instalowaniu najnowszych aktualizacji oraz zadbać o to, aby dla dowolnej używanej wersji oprogramowania zainstalować wszystkie łaty zabezpieczeń.
2. **Ukrywanie wersji oprogramowania i innych wrażliwych informacji**
   - Należy zamaskować wersję oprogramowania, która jest używana w zapytaniach osiągalnych z zewnątrz. Jako przykład posłuży Apache.
   - W zależności od dystrybucji w pliku httpd.cnf lub apache2.conf należy umieścić poniższe dwa wiersze:

   ```
   ServerTokens Prod
   ServerSignature Off
   ```
3. **Powiadomienia o poprawkach dotyczących bezpieczeństwa**
   - Należy być poinformowanym o tym, kiedy oprogramowanie wymaga aktualizacji. Administrator zobowiązany jest do zapisania się na listę mailingową dystrybucji systemu Linux zainstalowanej na serwerze na potrzeby aktualizacji zabezpieczeń.
   - Każde zewnętrzne oprogramowanie dodawane spoza dystrybucji również powinno być wyposażone w pewne mechanizmy informowania o aktualizacjach zabezpieczeń.
   - Warunek konieczny: trzeba dokonać na wszystkich tych listach subskrypcji z adresem e-mail, który zapewnia dostarczenie informacji do wszystkich członków zespołu.
4. **Procedura reagowania na maile z listy dystrybucyjnej**
   - Gdy z jednej z list mailingowych dotyczących zabezpieczeń na wskazany wcześniej adres e-mail, do którego wszyscy uprawnieni członkowie zespołu mają dostęp, dociera informacja dotycząca bezpieczeństwa, administrator odpytuje środowisko, aby sprawdzić, czy oprogramowanie, którego dotyczy alert, zostało zainstalowane na którymkolwiek z komputerów.
   - Jeśli oprogramowanie jest używane i nie zostało zaktualizowane, należy utworzyć zlecenie zawierające informację o luce w zabezpieczeniu, z jawnie wymienioną nową wersją oprogramowania, do której zostanie wykonana aktualizacja.
   - Nadaj zleceniu roboczemu priorytet w zależności od stopnia zagrożenia wymienionego w informacji o luce w zabezpieczeniach oraz podaj opis zagrożeń dla Twojego środowiska.
   - Po zaktualizowaniu oprogramowania zmodyfikuj zlecenie poprzez dołączenie zrzutu ekranu lub zapisu z dziennika zdarzeń, jako dowód na to, że aktualizacja została zainstalowana na serwerze.

Dla procedury zamieszczonej powyżej przyjęto dwa założenia:
1. Istnieje jakiś system zleceń roboczych
2. Istnieje sposób na odpytywanie środowiska, aby przekonać się, czy określony pakiet został zainstalowany, a jeśli tak, to w jakiej wersji.

Gdy nie ma żadnego z tych systemów, zaleca się ich instalację niezależnie od rozmiarów organizacji.

5. **Konta współdzielone**
   Konta współdzielone są złą praktyką, której należy bezwzględnie unikać.
6. **Szyfrowanie dysków**
   - Szyfruj dyski, gdy tylko jest to uzasadnione i wymaga tego polityka firmy, lub też administrujesz systemami, które nie są często restartowane, a zawierają poufne dane. Ewentualnie przenosisz dyski pomiędzy biurami. Szyfrowanie dysków jest nieuzasadnione, gdy zakłóca lub spowalnia cykl pracy. Ostatecznie bezpieczeństwo stanowi równowagę pomiędzy zagrożeniami i użytecznością, a wyznaczenie odpowiedniej granicy nie jest łatwym zadaniem.
   - Należy szyfrować dane „w spoczynku". Nawet jeśli administrator zapomni zniszczyć dane na dysku przed wycofaniem go z użytku, można mieć pewność, że informacje w bazie danych nadal będą bezpieczne.
7. **Zasada najmniejszych przywilejów**
   - Zgodnie z nią każdy powinien dysponować tylko minimalnym poziomem uprawnień - takim, który jest niezbędny do wykonania określonej pracy, i nic ponadto.
   - W przypadku usług i serwerów zasadę najmniejszych przywilejów najbardziej bezpośrednio stosuje się w odniesieniu do reguł zapór firewall. Jeśli usługa nie musi komunikować się z inną usługą, to nie należy jej na to zezwalać. Wynika stąd logiczny wniosek, że administrator powinien utworzyć reguły zapory firewall, które domyślnie blokują cały ruch przychodzący, a zezwalają na ruch tylko tym usługom, które muszą się komunikować, by móc wykonywać swoją pracę. Jeśli usługa nie wymaga do działania uprawnień użytkownika root, to nie powinna działać z tożsamością użytkownika root, ale z tożsamością mniej uprzywilejowanego użytkownika.
8. **Obrona w głąb**
   - Całe otoczenie sieciowe powinno być traktowane jako potencjalnie wrogie środowisko.
   - Każda warstwa sieci powinna ograniczać ruch z innych sieci.
   - Każdy serwer powinien mieć swoje własne, lokalne programowe reguły zapory firewall.
   - Do logowania się do maszyn należy stosować klucze SSH, które powinny być chronione hasłem.
   - Należy stosować mechanizmy TLS w celu zabezpieczenia ruchu sieciowego.
   - Uwierzytelnianie serwera dla klientów należy wykonać przy użyciu certyfikatów.
9. **Dbałość o proste procedury i schematy**
   - Należy zadbać o prosty schemat sieci, który pokazuje dane przepływające prostymi liniami w dół stosu, co pozwala łatwiej go zrozumieć i zabezpieczyć.
   - Należy także zadbać o proste listy kontroli dostępu (ACL).
   - Dzięki prostym środkom zabezpieczeń i prostym systemom możliwa jest analiza wszystkich scenariuszy ataku, wykrycie słabych punktów w projektach i szybsze znalezienie nieprawidłowości lub błędów.
10. **Zabezpieczanie haseł**
    - Należy stosować hasła o minimalnej długości 12 znaków, w tym duże i małe litery, cyfry oraz znaki specjalne.
    - Zabrania się używania tego samego hasła do wszystkich kont**.**

- Należy blokować próby złamania hasła bez względu na metodę. Trzykrotne wpisanie błędnego hasła skutkuje banem. Przykładowe oprogramowanie, które można zastosować: fail2ban.
- Tam, gdzie to możliwe, stosuj dwuskładnikowe uwierzytelnianie.
- Gdy istnieje możliwość, powinno zmienić się domyślne, znane ścieżki logowania do serwisów.

11. Ustawienie niestandardowych portów dla usług

W miarę możliwości należy zadbać o ustawienie niestandardowych portów dla usługi SSH.

12. Kompartmentalizacja

- Należy odizolować każdą usługę na wydzielonym serwerze. Dzięki temu, jeśli nastąpi włamanie do jednej z usług, pozostała część infrastruktury pozostanie bezpieczna.
- Zabrania się przechowywania wszystkich danych w jednej bazie danych.
- Zabrania się przechowywania wszystkich ważnych plików na jednym serwerze.
- Jeśli administrator korzysta z wielu różnych usług, które potrzebują przechowywania danych w bazie danych, każda z nich powinna mieć swoją własną bazę dla swoich danych.
- Należy unikać umieszczania różnych usług sieciowych w tej samej podsieci. Powinny być one odizolowane w miarę możliwości.

13. Zasady kont i grup

- Polityka powinna obejmować utrzymywanie porządku wśród grup i kont użytkowników.
- Rolą administratora jest znajomość grup i ich ról, które pełnią, a także tworzenie dokumentacji w postaci elektronicznej tworzonych kont i grup. Należy pamiętać o usuwaniu nieaktywnych kont oraz grup, jeśli nie są już potrzebne. Przykładem tutaj może być projekt związany z nowym klientem w firmie, czy też przyjęcie nowego pracownika, co wymaga utworzenia grup z poszczególnymi rolami oraz kont użytkowników, którzy będą przypisani do grup.
- W momencie zamknięcia projektu, zmiany ról pracowników lub gdy pracownik opuszcza organizację, administrator powinien usunąć użytkowników z grup, a następnie zadbać o wykonanie kopii potrzebnych plików, które znajdują się w katalogach domowych użytkowników i usunięcie kont z systemu.
- Należy stosować dobre praktyki utrzymywania wszystkich kont w całym ich cyklu życia. Powinno się zdefiniować procedury zarówno podczas przyjmowania nowych pracowników, jak i ich zwalniania. Reguły powinny jasno określać, do jakich kont i grup pracownik może mieć dostęp oraz jak dodawać lub usuwać ich z tych kont oraz grup.
- Zadania dodawania jak i usuwania powinny mieć swoje odzwierciedlenie w zleceniach roboczych, najlepiej w postaci systemu zgłoszeń. Tam, gdzie jest taka potrzeba, konta wymagające dodatkowych uprawnień powinny wymagać zgody od osoby odpowiedzialnej za określoną grupę.

Dobre praktyki administracyjne

1. Należy zadbać o to, aby nie używać zwykłych nazw kont administracyjnych dla administratora superużytkownika GRUB2. Powinno się unikać używania popularnych nazw kont administracyjnych, takich jak root, administrator lub administrator dla konta superużytkownika GRUB2.Zaleca się, aby hasło konta administratora programu ładującego było inne niż główne dane uwierzytelniające.
2. Dobrą praktyką jest, aby nie zabezpieczać dysku twardego hasłem w BIOS, lecz jedynie zmianę ustawień, co sprawi, że BIOS nie będzie pytać o hasło po każdym zaplanowanym restarcie serwera.
3. Dobrą praktyką jest wymaganie szyfrów o średniej sile (ang. *medium*), dzięki czemu można uzyskać przyzwoitą mieszankę zgodności ze zdalnymi serwerami pocztowymi bez dopuszczania niektórych słabszych szyfrów.
4. Należy przenieść wiadomości oznaczone jako spam do specjalnego folderu o nazwie spam lub junk.
5. Jeśli to możliwe adresy IP, nazwy hostów, identyfikatory użytkowników i grup powinny być zarządzane centralnie.
6. Obsługa włamań leży po stronie zespołu specjalistycznego.
7. Kontrola sieciowych systemów plików należy do wyznaczonego w tym celu zespołu.
8. Usuwanie użytkowników z systemu powinno odbyć się najpóźniej do 30 dni po ich odejściu z organizacji. Lista identyfikatorów pracowników powinna być przekazywana co miesiąc do administratora.
9. Administrator zobowiązany jest do obsługi materiału objętego prawami autorskimi.
10. Administrator zobowiązany jest do stworzenia schematu sieci informatycznej organizacji i jej aktualizacji.
11. Należy zadbać o stworzenie procedury kopii zapasowych i ich regularne wykonywanie.
12. Warto przemyśleć wdrożenie SLA (ang. *Service Level Agreements*), które definiuje sposób i czas wykonywania poszczególnych czynności.
13. Należy zadbać o ochronę prywatności i wszelkie jej naruszenia powinny być zgłaszane na odpowiedni adres e-mail.
14. Zagadnienia prawne należą do zespołu prawników, jednak administrator powinien mieć stworzoną procedurę dotycząca postępowania w przypadku instalacji, konfiguracji i utrzymania na serwerze licencjonowanego oprogramowania.
15. Do Administratora należy poinformowanie użytkowników poprzez informacje powitalne w skryptach startowych nowo skonfigurowanego systemu. Jeśli do zdalnego logowania jest wykorzystywany protokół SSH, należy skonfigurować plik sshd_config w taki sposób, aby komunikat był wyświetlany w powitaniu sesji SSH.
16. Należy określić reguły informatyczne w organizacji, które jednoznacznie określają, że użytkownicy nie maja prawa wykorzystywać zasobów organizacji do nielegalnych celów.
17. Administrator powinien podnosić swoją wiedzę w sposób ciągły poprzez szkolenia i programy certyfikacyjne, a także konferencje informatyczne.

3. Podsumowanie

Gdy zrozumie się pewne ogólne zasady bezpieczeństwa, można je zastosować do dowolnego problemu, który ma się do rozwiązania jako administrator serwerów, bez względu na to, czy jest to hartowanie konkretnej usługi, czy też podjęcie decyzji, która dotyczy projektu infrastruktury zupełnie nowej aplikacji. W zależności od konkretnego środowiska i zagrożeń, które w nim występują stosowane przez administratora środki są podejmowane adekwatnie do sytuacji, jednak ogólne zasady obowiązują zawsze. Takie samo podejście zastosowane jest do określonych rodzajów problemów w każdym z pozostałych rozdziałów w tej pracy. W rozdziale pierwszym zastosowałem te zasady przy wybieraniu haseł BIOS, GRUB2 oraz kont użytkowników, w rozdziale drugim podczas zabezpieczania logowania sieciowego i wreszcie w podrozdziale 2.2 i 2.3, podczas konfiguracji kont e-mail w bazie danych oraz kont Samby. W ramach preferowanej strategii dotyczącej haseł opowiedziałem się za prostotą, dążąc do wyboru hasła złożonego z minimum 12 znaków oraz bez wymagań związanych ze złożonością. Zastosowałem ideę obrony w głąb, dodając sól do haseł oraz zabezpieczając przesyłanie haseł, tak, aby uniemożliwić ich złamanie oraz podsłuchanie. Pokazałem też, że niezależnie od usługi działającej na serwerze możliwe jest wykorzystanie kilku podstawowych metod hartowania.

W rozdziale pierwszym skupiłem się na czynnościach utwardzania, które można zastosować do niemalże dowolnego serwera. W szczególności opisałem hartowanie dostępu użytkownika root za pomocą polecenia sudo oraz znaczenie mechanizmów logowania zdarzeń w dziennikach. Ponadto, biorąc pod uwagę, że obecnie prawie na każdym serwerze używa się SSH do zdalnej administracji, omówiłem kilka technik hartowania tej usługi – począwszy od ogólnego hartowania konfiguracji serwera SSH, które polega na wyłączeniu logowania jako root, a skończywszy na użyciu do uwierzytelniania kluczy SSH zamiast hasła. Opisałem też zabezpieczanie sieci i pokazałem, że jest to proces wielopoziomowy. Pierwsza warstwa polega na wykorzystaniu reguł zapory firewall po to, aby zezwolić w sieci tylko na dozwolony ruch i zablokować ruch, który nie jest uprawniony. Następna warstwa jest odpowiedzialna za zabezpieczenie uprawnionego ruchu sieciowego przed przechwyceniem. Do tego celu stosuje się szyfrowanie TLS oraz SSL.

W rozdziale drugim zawarłem informacje o tym jak istotną umiejętnością jest ich wzmacnianie w celu przeciwdziałania atakom. W tym rozdziale omówiłem różne

sposoby wzmacniania serwerów WWW i poczty. Zacząłem od omówienia konfiguracji HTTPS, które pozwala chronić użytkowników odwiedzających witrynę zarówno poprzez szyfrowanie ruchu WWW, jak i zapewnienie sposobu uwierzytelniania serwera. Zaprezentowałem sposób wprowadzania zaawansowanej konfiguracji, która pozwala na zabezpieczenie przed określonymi atakami na połączenia HTTPS, takimi jak ataki degradacji protokołu. Omówiłem także mechanizmy działania zapory WAF, której konfiguracja znajduje się w instrukcji.

W części drugiej podrozdziału 2.1 zwróciłem uwagę na zabezpieczenia dotyczące PHP ze szczególnym naciskiem na bezpieczeństwo i pokazałem w jaki sposób można wykorzystać FastCGI do przetwarzania plików PHP, co jest jedynym bezpiecznym rozwiązaniem. Jeśli jakakolwiek część infrastruktury w firmie przechowuje cenne dane, to jest nią warstwa bazy danych. Nawet te organizacje, które podchodzą do zabezpieczeń nieco luźniej, powinny włożyć wysiłek w zabezpieczenie baz danych i umieścić je głęboko wewnątrz sieci.

W paragrafie 2.1.3 omówiłem szereg kroków, które można podjąć w celu wzmocnienia zabezpieczeń bazy danych – począwszy od podstawowych, polegających na lokalizacji bazy danych w sieci, poprzez sposoby jej bezpiecznego administrowania, a skończywszy na konfigurowaniu uprawnień.

Na koniec w instrukcji wdrażania serwera opisałem, w jaki sposób można wzmocnić dostęp do bazy danych przez sieć, korzystając z zabezpieczeń komunikacji z bazą danych za pomocą szyfrowania TLS.

W podrozdziale 2.2 skupiłem się na hartowaniu serwera poczty. Jak można zaobserwować, większość czynności związanych z utwardzaniem systemu pocztowego skupia się na tym, aby zapobiec rozsyłaniu spamu. W części 1. objaśniłem, jak nie dopuścić do przekształcenia serwera w system open relay. W części 2. omówiłem zagadnienia uwierzytelniania usług poczty oraz zabezpieczania ruchu e-mail za pomocą protokołu TLS. Na koniec w części trzeciej, czwartej, piątej i szóstej tego podrozdziału omówiłem zaawansowane metody ochrony przed spamem i wirusami za pomocą SpamAssassin, ClamAV oraz Fail2ban. Obecnie coraz więcej serwerów dużych dostawców usług pocztowych posiada zaimplementowane wszystkie opisane mechanizmy w celu wzmocnienia swoich serwerów. Administratorzy wiedzą o tym, że inne legalne serwery poczty elektronicznej w internecie wykorzystują te same mechanizmy. Im mniej opisanych protokołów zaimplementujemy, tym większe prawdopodobieństwo, że zdalny serwer uzna naszą wiadomość za spam, nawet, gdy ją

zaakceptuje. Jeśli dla administratora jest to istotna kwestia, aby wiadomość e-mail została dostarczona i nie została oznaczona jako spam, to zdecydowanie powinien on rozważyć zastosowanie wszystkich zaawansowanych czynności z zakresu hartowania.

W podrozdziale 2.3 opisałem krótko zagrożenia związane z protokołem SMB w wersji 1 oraz oprogramowaniem Samba, które pozwala uruchomić serwer plików charakteryzujący się dużą kulturą pracy pod względem stabilności i po odpowiednim zabezpieczeniu oferuje dobry poziom bezpieczeństwa.

Praca ta ma za zadanie pokazać, w jaki sposób można zabezpieczyć systemy przed włamaniami. Jednak niezależnie od tego, jak wielki wysiłek podejmiemy, napastnikowi nadal może się udać przejąć kontrolę nad którymś z systemów. Należy wspomnieć też o tym, że temat bezpieczeństwa serwerów nie został całkowicie wyczerpany, ponieważ chcąc zabezpieczyć się przed skanowaniem moich serwerów oraz zwiększyć ich poziom bezpieczeństwa, zmuszony zostałem do implementacji na serwerze poczty SPF, DKMF oraz DMARC. Dodatkowo zainstalowałem tripwire do sprawdzania integralności systemu plików, snort, który powiadamia mnie o skanowaniu portów serwera oraz logwatch, który wysyła raporty na mój adres e-mail jako administratora serwera poczty. Należy wspomnieć, że jedną z ważniejszych kwestii jest robienie kopii zapasowych danych znajdujących się na serwerach, ponieważ ataki ransomware zdarzają się coraz częściej. Dlatego w zakresie nieoficjalnych obowiązków administratora leży ciągły rozwój wiedzy w zakresie szeroko pojętego bezpieczeństwa.

4. Bibliografia

4.1. Publikacje książkowe:

[1] Matotek D., Turnbull J., Lieverdink P.; Profesjonalne administrowanie systemem; Wyd. II, Gliwice 2018, HELION, s. 454

[2] Nemeth E., Snyder G., Hein T. R., Whaley B., Mackin D., Garnett J., Branca F., Mouat A.; Unix i Linux Przewodnik Administratora Systemów; Wyd. V, Gliwice 2018, HELION, s. 1046

[3] Rankin K.; Hartowanie Linuksa we wrogich środowiskach sieciowych. Ochrona serwera od TLS po Tor; Wyd. Gliwice 2018, HELION

4.2. Strony internetowe:

[i1] Red Hat Training - 3.3. Confined and Unconfined Users, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/selinux_users_and_administrators_guide/sect-security-enhanced_linux-targeted_policy-confined_and_unconfined_users , na dzień 15-03-2019,

[i2] Red Hat Training - Chapter 50. Working With SELinux, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/5/html/deployment_guide/rhlcommon-chapter-0017, na dzień 15-03-2019

[i3] W3Techs - Usage of content management systems, https://w3techs.com/technologies/overview/content_management/all, na dzień 15-03-2019

[i4] obfusc.at/ed - Bayes wiecznie żywy, http://obfusc.at/ed/bayes_filtering_pl.html, na dzień 15-03-2019

[i5] sieve.info - What is Sieve?, http://sieve.info/, na dzień 15-03-2019

[i6] Red Hat Training - 9.15.5. Recommended Partitioning Scheme, https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/installation_guide/s2-diskpartrecommend-x86 , na dzień 15-03-2019