Bezpieczeństwo komputerowe lista1

Ewa Kasprzak

3 marca 2024

1 Klucze SSH

SSH (Secure Shell) to protokół sieciowy służący do bezpiecznej komunikacji w sieci. Klucze SSH są częścią tego protokołu i służą do uwierzytelniania użytkowników oraz szyfrowania danych.

Klucze SSH składają się z dwóch części:

1. **Klucz prywatny**: Jest to unikalny klucz, który powinien być przechowywany bezpiecznie przez użytkownika. Nie powinien być udostępniany ani przesyłany przez sieć. Klucz prywatny służy do deszyfrowania danych otrzymanych od serwera.
2. **Klucz publiczny**: Jest to klucz, który jest udostępniany serwerowi SSH. Serwer używa klucza publicznego do szyfrowania danych wysyłanych do klienta.

Gdy użytkownik próbuje połączyć się z serwerem SSH, serwer szyfruje losową wiadomość za pomocą klucza publicznego użytkownika. Następnie użytkownik odszyfrowuje wiadomość za pomocą swojego klucza prywatnego i wysyła ją z powrotem do serwera. Jeśli serwer jest w stanie odczytać wiadomość, oznacza to, że użytkownik jest tym, za kogo się podaje, ponieważ tylko on posiada odpowiedni klucz prywatny.

2 Klucze SSH a GitHub

Klucze SSH są często używane do automatycznego logowania się do serwerów, bez potrzeby wpisywania hasła za każdym razem. Są one również używane w systemach kontroli wersji, takich jak Git, do uwierzytelniania użytkowników.

Gdy klucze SSH nie zostały wygenerowane, a chcemy pobrać prywatne repozytorium poleceniem git clone https://gitlab.com/username/reponame, nie trzeba się obawiać o brak bezpieczeństwa przesyłanych danych.

GitHub używa uwierzytelniania podstawowego (Basic Authentication) do uwierzytelniania użytkowników podczas klonowania repozytoriów za pomocą HTTPS. Oznacza to, że musisz podać swoją nazwę użytkownika i hasło (lub token dostępu osobistego, jeśli wolisz) podczas klonowania repozytorium. Te dane uwierzytelniające są również przesyłane w sposób zaszyfrowany.

Dane przesyłane za pomocą protokołu HTTPS nie są przesyłane jako tekst jawny. HTTPS używa protokołu SSL/TLS do szyfrowania danych, co oznacza, że nawet jeśli ktoś byłby w stanie przechwycić te dane, nie byłby w stanie ich odczytać bez klucza deszyfrującego.

Istnieją różne typy kluczy SSH, takie jak RSA, ECDSA, ED25519, warto zaznaczyć jednak, że klucze RSA wygenerowane po 2 listopada 2021 w celu korzystania z nich na platformie GitHub, muszą korzystać z algorytmu podpisu SHA-2.

3 Konfiguracja kluczy SSH na platformie GitHub: Analiza i Wybór

Na platformie GitHub, klucze SSH są wykorzystywane do uwierzytelniania, podpisywania commitów, lub obu tych operacji. Proces generowania nowego klucza jest realizowany za pomocą polecenia:

```
ssh-keygen -t ed25519 -C ‘‘email@example.com’’
```

Polecenie to jest konfigurowane w zależności od wybranego typu klucza. Wygenerowany klucz jest następnie dodawany do ssh-agent za pomocą polecenia:

```
eval ‘‘$(ssh-agent -s)’’
```

oraz

```
ssh-add  /.ssh/id_ed25519,
```

gdzie `id_ed25519` jest nazwą prywatnego klucza. Szczegółowe instrukcje są dostępne na stronie tutaj.

3.1 Wybór klucza ED25519

Wybór klucza ED25519 jest podyktowany dwoma głównymi czynnikami:

1. **Bezpieczeństwo:** ED25519, oparty na kryptografii krzywych eliptycznych, jest uważany za bezpieczniejszy. Jest to spowodowane większą trudnością złamania tej metody w porównaniu do faktoryzacji dużych liczb pierwszych, na której opiera się RSA.
2. **Efektywność:** ED25519 jest bardziej efektywny pod względem rozmiaru klucza, obliczeń i pamięci. Operacje takie jak generowanie i weryfikacja podpisów, czy szyfrowanie i deszyfrowanie wiadomości, są szybsze w przypadku ED25519 niż RSA.

Typ klucza ED25519 jest również sugerowany przez artykuł dostępny na stronie tutaj, który jest częścią oficjalnej dokumentacji platformy GitHub.

3.2 Alternatywa: Klucz RSA

Alternatywą dla klucza ED25519 jest klucz RSA o długości 4096 bitów, klucze RSA o długości 2048 bitów są wspierane jedynie do roku 2030.

4 Uwierzytelnianie dwuskładnikowe(2FA) na platformie GitHub

2FA na GitHubie polega na generowaniu kodu przez aplikację na urządzeniu mobilnym lub wysyłaniu go jako wiadomości SMS. Kod jest generowany za każdym razem, gdy ktoś próbuje zalogować się na swoje konto na GitHub.com.

Opcjonalnie, można dodać do konta klucz dostępu. Klucze dostępu są podobne do kluczy bezpieczeństwa, ale spełniają zarówno wymagania dotyczące hasła, jak i 2FA, dzięki czemu można zalogować się na swoje konto jednym krokiem.

Od marca 2023 do końca 2024 roku, GitHub stopniowo wprowadzał wymaganie od wszystkich użytkowników, którzy dodają kod na GitHub.com, włączenia jednej lub więcej form dwuetapowego uwierzytelniania (2FA).

Dwuetapowe uwierzytelnianie (2FA) pomaga zabezpieczyć konto przed różnymi wektorami ataku, takimi jak:

1. **Ataki typu phishing**: 2FA może zapobiec atakom phishingowym, ponieważ nawet jeśli atakujący zdobędzie hasło, nadal będzie potrzebował drugiego czynnika, aby uzyskać dostęp do konta.
2. **Ataki na podstawie słabych haseł**: Jeśli hasło jest słabe, 2FA dodaje dodatkową warstwę ochrony, wymagając drugiego czynnika uwierzytelniania.
3. **Ataki typu brute force**: Ataki te polegają na wielokrotnym próbowaniu różnych kombinacji haseł, aby uzyskać dostęp do konta. Z 2FA, nawet jeśli atakujący odgadnie hasło, nadal będzie potrzebował drugiego czynnika uwierzytelniania.