Bezpieczeństwo Lista 1

Bartosz Michalak

10 March 2024

1 Zadanie 1

1. SSH key information:

   The **SSH key pair** is used to **authenticate the identity of a user or process** that wants to access a remote system using the SSH protocol. The **public key** is used by **both the user and the remote server** to **encrypt messages**. On the remote server side, it is saved in a public key file. On the user's side, it is stored in SSH key management software or in a file on their computer. The **private key** remains **only on the system being used** to access the remote server and is used to **decrypt** messages.

   When a user or process requests a connection to the remote server using the SSH client, a **challenge-response** sequence is initiated to complete **authentication**. The **SSH server** recognizes that a connection is being requested and **sends an encrypted challenge** request using the shared public key information. The **SSH client** then **decrypts the challenge message** and responds back to the server. The **user** or process **must respond correctly** to the challenge to be granted access. This **challenge-response sequence happens automatically** between the SSH client and server without any manual action by the user.

2. Connecting SSH keys to account, what of it?

   Adding an SSH key to your GitHub account allows you to **securely authenticate with GitHub without having to enter your username and password** every time you interact with GitHub over SSH.

3. What happens without SSH, how data is transferred (for eg. via git clone command) is it encrypted?

It isn't encrypted. It is sent as a plain text (**risky**).

4. What types of SSH keys are supported?

   GitHub supports SSH keys in the following formats:

   (a) **RSA**: RSA (Rivest-Shamir-Adleman) is one of the most commonly used asymmetric cryptographic algorithms. It generates keys in pairs: public and private keys. GitHub supports RSA keys for SSH authentication.

   (b) **DSA**: DSA (Digital Signature Algorithm) is another asymmetric cryptographic algorithm used for digital signatures. While less commonly used than RSA, GitHub also supports DSA keys for SSH authentication.

   (c) **ECDSA**: ECDSA (Elliptic Curve Digital Signature Algorithm) is an asymmetric cryptographic algorithm based on elliptic curve cryptography. It provides similar security levels to RSA but with shorter key lengths. GitHub supports ECDSA keys for SSH authentication.

   (d) **Ed25519**: Ed25519 is a modern public-key signature system based on the elliptic curve known as Curve25519. It provides high security with relatively short key lengths and is becoming increasingly popular. **GitHub highly recommends using Ed25519** keys for SSH authentication due to their **security benefits**.

5. What type of key and what length did you choose, why?

   I chose **Ed25519** which is **256 bits long**, as it provides the best security and I have no "Legacy reasons" to use (for eg.) RSA. It is also **the most secure ssh key type available on Github**.

6. What types of 2FA are supported by GitHub?

   GitHub supports several methods for Two-Factor Authentication (2FA), allowing users to choose the option that best suits their preferences and security needs. These methods include:

   (a) **Authentication Apps**: You can use an authentication app such as **Google Authenticator**(this is the one I use), Authy, or Microsoft Authenticator to generate temporary codes for 2FA. Once configured,

these apps provide you with a time-based one-time password (TOTP) that you use alongside your password when logging in.

(b) **SMS Text Messages**: GitHub can send you a temporary code via SMS to your registered mobile phone number. You enter this code along with your password when logging in.

(c) **Universal 2nd Factor (U2F) Security Keys**: GitHub also supports U2F security keys such as YubiKey or Google Titan Security Key. These physical devices plug into your computer's USB port or use NFC (Near Field Communication) and provide an additional layer of security by requiring physical presence to authenticate.

(d) **Recovery Codes**: GitHub provides recovery codes that you can download and use as backup codes in case you lose access to your primary 2FA method. Each recovery code can be used only once and should be stored securely.

7. What kinds of attack vectors 2FA eliminates?

(a) **Password Guessing**: Even if an attacker manages to obtain **or guess your password** through methods like **brute-force** attacks or **dictionary attacks**, they would still need the second factor (e.g., temporary code from an authentication app or SMS) to successfully authenticate. 2FA mitigates the risk posed by weak or compromised passwords.

(b) **Phishing**: Traditional phishing attacks rely on tricking users into entering their login credentials into fake login pages controlled by attackers. With 2FA enabled, even if a user falls victim to phishing and provides their password, the attacker would still need the second factor to gain access to the account. This significantly reduces the effectiveness of phishing attacks.

(c) **Credential Stuffing**: In credential stuffing attacks, attackers **use lists of compromised usernames and passwords obtained from data breaches** to attempt unauthorized access to various accounts. With 2FA enabled, even if attackers have valid credentials, they would still need the second factor to successfully authenticate, making credential stuffing attacks much less effective.

(d) M**an-in-the-Middle (MITM) Attacks**: In MITM attacks, an attacker **intercepts communication between a user and a server** to eavesdrop on or manipulate the data exchanged. 2FA mitigates this risk by requiring a second factor that is typically time-sensitive and cannot be intercepted or reused by the attacker.

(e) **Keylogging**: Keyloggers are malicious software or hardware devices that **record keystrokes entered by users, including passwords**.

While keyloggers can capture passwords, they cannot capture the second factor provided by 2FA methods like authentication apps or U2F security keys, thus protecting the account from unauthorized access.