SSH with public keys and TOTP

OpenSSH authentication chains and PAM integration

Two distinct authentication methods

OpenSSH can require more than one successful authentication method. AuthenticationMethods publickey,keyboard-interactive:pam requests a successful public-key signature followed by a PAM-backed keyboard-interactive challenge. This is different from permitting either method. The server must also have PubkeyAuthentication, KbdInteractiveAuthentication and UsePAM enabled appropriately.

Because both a private key and a TOTP app are possession-based, the combination is best described as key-plus-TOTP or two-step authentication rather than assuming it satisfies every formal definition of two-factor authentication.

```
AuthenticationMethods publickey,keyboard-interactive:pam
PubkeyAuthentication yes
KbdInteractiveAuthentication yes
PasswordAuthentication no
UsePAM yes
```

PAM authentication path

On a common Ubuntu setup, /etc/pam.d/sshd includes common-auth, which may require the account password. If the objective is public key plus TOTP, leaving a password authentication module in the SSH auth stack can inadvertently demand a third credential. The exact PAM stack must be examined and designed for the installed distribution.

A typical arrangement replaces the auth include of common-auth with a required pam_google_authenticator.so rule while retaining account, password-management and session lines. Do not indiscriminately delete those other rules: they may enforce account validity and create the expected login session.

```
sudo grep -nE "^(auth|account|session|@include)" /etc/pam.d/sshd
# Typical SSH auth rule:
auth required pam_google_authenticator.so
```

Enrolment and client experience

The account being authenticated must have an enrolled TOTP secret accessible to the PAM module. A user first proves possession of an authorised private key; the server then requests a verification code via keyboard-interactive. Microsoft Authenticator and Google Authenticator both generate standard TOTP values from a compatible enrolled secret.

An SSH private-key passphrase is local to the client and is not an additional server-side authentication method. The client may prompt for the passphrase before the server presents the TOTP challenge.

```
google-authenticator
ssh -vvv admin@server.example.net
```

Configuration precedence and Match blocks

OpenSSH configuration may contain Include files and Match rules that change the effective authentication policy. Use sshd -T with -C connection parameters to verify settings for representative users and source addresses. A successful sshd -t syntax check does not prove that the intended authentication chain applies.

Automated clients cannot normally respond to interactive TOTP prompts. Design a separately restricted automation path, such as a dedicated account and forced command, instead of silently weakening the general interactive policy.

```
sudo sshd -t
sudo sshd -T -C user=admin,host=server.example.net,addr=198.51.100.25 | grep -E "^(authentica
    tionmethods|kbdinteractiveauthentication|passwordauthentication|usepam) "
```

Validation and recovery

Keep an existing SSH session open while changing PAM or sshd configuration. Confirm key authentication works before enabling the second step. Reload sshd only after syntax validation, then test a new connection. Verify that password-only login is refused and that the intended key-plus-code flow succeeds.

If login fails, examine client verbose output, server logs, secret file permissions and time synchronisation. Ensure console or out-of-band access is available for rollback; a broken PAM stack can prevent every remote administrator from logging in.

```
sudo systemctl reload ssh
sudo journalctl -u ssh --since today
timedatectl status
```

Reference scope: commands and defaults vary by distribution and software version. Consult the locally installed manual pages for exact syntax and behaviour.