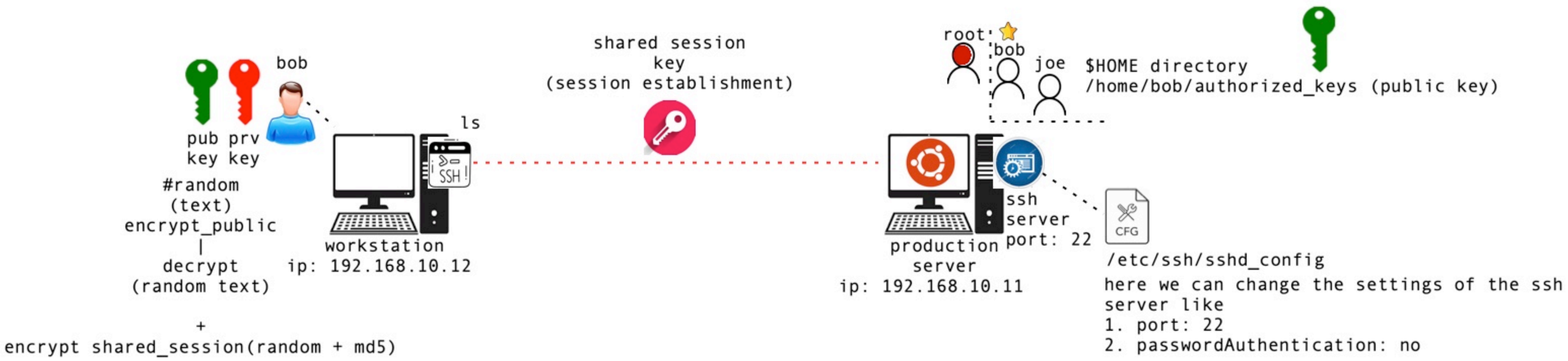
asymmetric key-based authentication



#random
(text)
encrypt_public
|
decrypt
(random text)

+
encrypt shared_session(random + md5)

here we can change the settings of the ssh server like
1. port: 22
2. passwordAuthentication: no

#1 Initial Key [setup] (authentication) @Administrator
1. The server administrator generates an public & private key, and shares the public/private key with the user for whom he wanted to grant the access to the remote server machine.

2. Along with that this public key will be registered on the remote server machine for one of the linux user as authorized_keys, stating who ever comes with this public key is authorized access the computer onbehalf of this user.
For eg.. the administrator can register the public key for bob user on the remote machine authorizing the users to connect to the machine as bob user using that public key

#2 Session establishment
The client upon sending the request to the server asking for communication, the session establishment process will begin, here both the parties will compute an shared session key. It will be used as symmetric key for encrypting/decrypting data that is being exchanged by both the parties

#3 Authentication
1. The SSH server challenges the client asking him to authenticate in gaining the access to the remote computer
2. The client will send the username and the public key of the user onbehalf of whom he wanted to access the remote server
bob + public_key

3. server upon receiving the username & public_key, it verifies whether this public key is associated with the bob as authorized key or not
if this key is not authorized for bob user, immediately it denies the access

if the key is associated with bob as authorized_key, then server wants to verify whether the client has private key or not (for authentication)

4. For this the server will generate a random text, encrypts with bob's public_key and sends to the client challenging him to decrypt and send back if he has corresponding private key

5. The client upon receiving the encrypted random text, will decrypt it using private key. Then attaches an md5 hash to it, encrypts it with shared session key and send to the server back

6. The server will takes the random text, decrypts using shared session key, and verifies the random text the client sent is matching with original text or not.
If it matches, it proves that the client holds private key and can be considered as authenticated and hance grants the access allowing him to perform operation onbehalf of the bob user.

How to revoke the access?
remove the public key from authorized_keys of the bob user on the server computer.