



**INSTITUTO FEDERAL**

Norte de Minas Gerais

Campus Januária

# Admin. Serviços de Redes

## - *Acesso Remoto* -



**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Acesso Remoto





**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Telnet

# TELNET



# Telnet

- **Telnet** é um dos protocolos padrões da Internet para acesso remoto a hosts (p.ex. servidores).
- Acesso remoto permite que um usuário efetue comandos e altere configurações em *hosts* distantes, através da **visualização do terminal remoto em sua própria estação**.
- Entretanto, o **TELNET não utiliza criptografia** na comunicação entre a máquina local e remota, o que pode causar um grave problema de segurança.



# Telnet

- Por padrão, o serviço Telnet baseia-se em conexões TCP através da Porta 23... Ou outra porta definida em:

```
# /etc/services
```

- Devido às suas limitações de segurança, **é usado somente em casos muito específicos.**







# Telnet

## ■ Instalação

```
# apt-get install telnetd
```

## ■ Configuração

```
# /etc/inetd.conf
```

## ■ Ativação do Servidor

```
# service openbsd-inetd start
```



# Segurança de Ambiente

## ***ATENÇÃO***

***Por questões de segurança  
NUNCA  
faça um acesso remoto  
diretamente para o usuário root.***



# Gestão de Usuários

- Crie um novo usuário no Server:

```
# adduser nome_usuario
```

- Conceder permissões de root ao usuário (se necessário)

```
# usermod -aG sudo nome_usuario
```

- Trocar para novo usuário (*Switch User*)

```
# su nome_usuario
```

- Trocar para usuário root

```
# su -
```





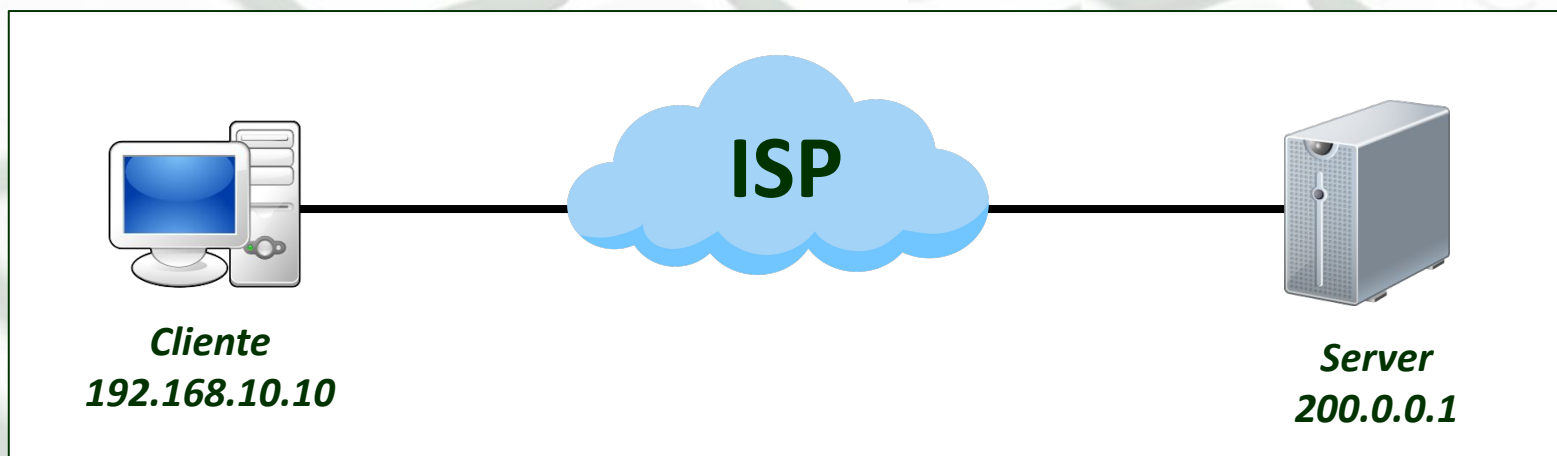
# Laboratório 09-1

- Crie um novo usuário no Server:

```
# adduser nome_usuario
```

- A partir da VM Cliente, acesse o Server remotamente.

```
# telnet 200.0.0.1
```

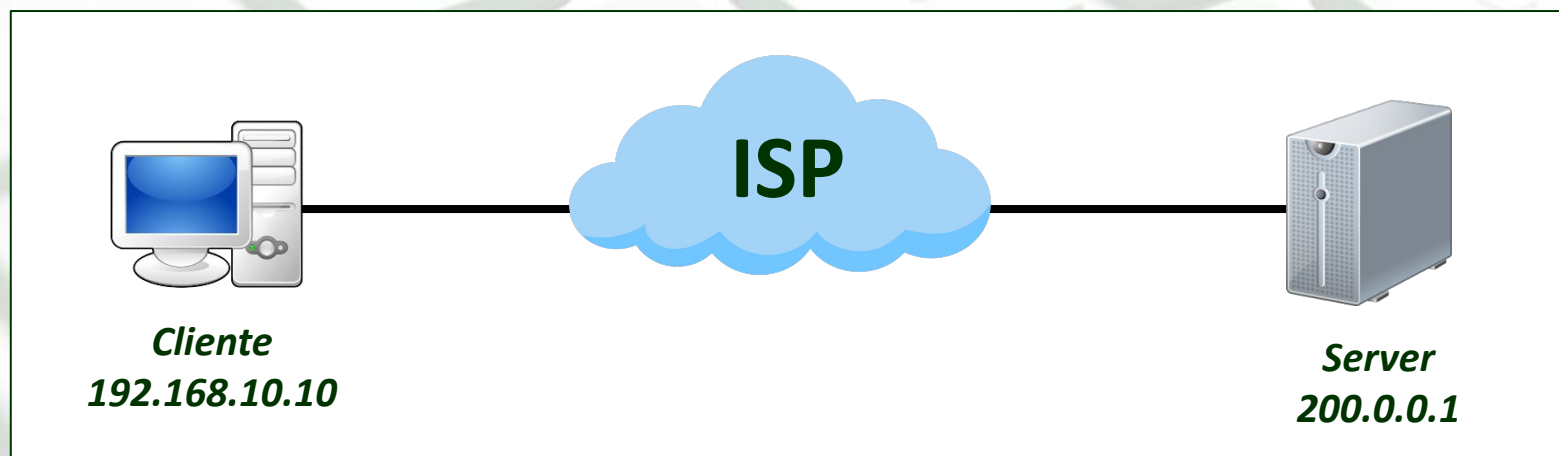




# Laboratório 09-1

- No ISP, utilize um *Sniffer* + Analisador de Pacotes para inspecionar como as credenciais de autenticação são transmitidas entre o Cliente e o Server.

```
# tcpdump -w escutaSSH.pcap
```

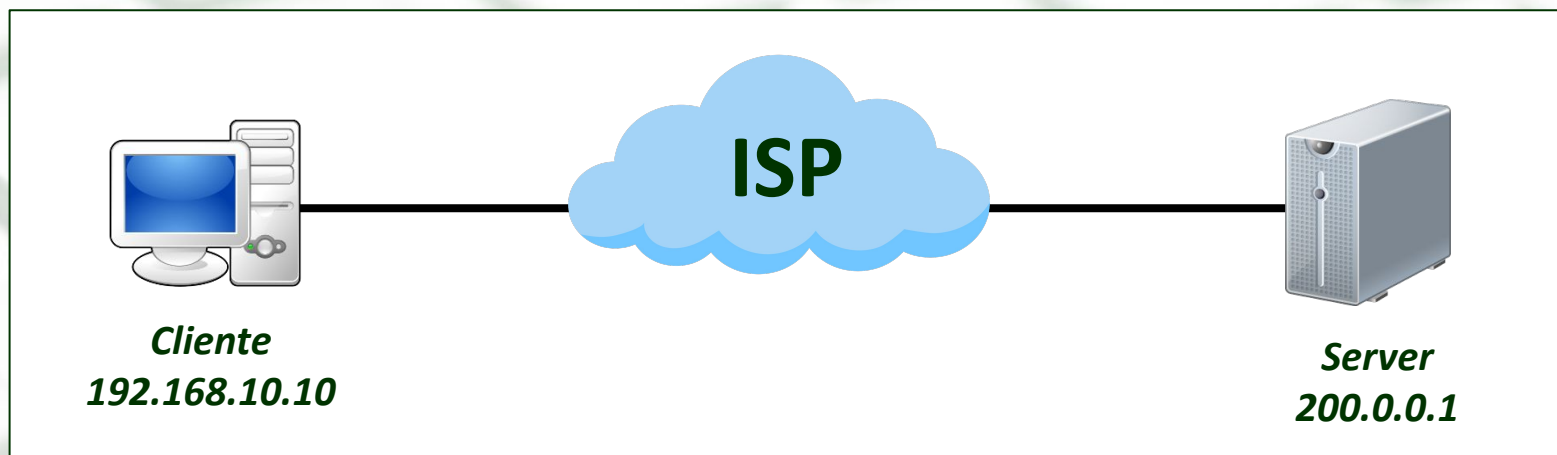




# Laboratório 09-1

- Façamos outro teste...
  - Inicie o servidor HTTP (Apache) do server...

```
# /etc/init.d/apache2 start
```



- Inicie uma conexão TELNET para a porta 80, e verifique...

```
# telnet 200.0.0.1 80  
> GET / HTTP/1.1
```



# Laboratório 09-1

- Outro teste... No terminal da sua máquina real...

```
$> telnet ipinfo.io 80
Trying 34.117.59.81...
Connected to ipinfo.io.
Escape character is '^]'.
GET / HTTP/1.1
Host: ipinfo.io
```

**INSTITUTO FEDERAL**

Norte de Minas Gerais

Campus Januária

## *Resultado da requisição a uma API Rest*

**HTTP/1.1 200 OK**

access-control-allow-origin: \*

x-frame-options: SAMEORIGIN

x-xss-protection: 1; mode=block

x-content-type-options: nosniff

referrer-policy: strict-origin-when-cross-origin

**content-type: application/json; charset=utf-8**

Content-Length: 357

date: Fri, 18 Apr 2025 21:43:19 GMT

vary: accept-encoding

via: 1.1 google

strict-transport-security: max-age=2592000; includeSubDomains

```
{
  "ip": "138.122.149.46",
  "hostname": "138-122-149-46.clientes.nettellinternet.com.br",
  "city": "Brasília de Minas",
  "region": "Minas Gerais",
  "country": "BR",
  "loc": "-16.2064,-44.4333",
  "org": "AS264337 NET COM INFORMATICA LTDA - ME",
  "postal": "39330-000",
  "timezone": "America/Sao_Paulo",
  "readme": "https://ipinfo.io/missingauth"
}
```





# Serviços sem Criptografia

Por razões óbvias, **protocolos inseguros** como o **Telnet** que oferece serviço de acesso remoto, e outros serviços, como **HTTP** (serviço WEB), **FTP** (transferência de arquivos), e **DNS** (resolução de nomes), estão caindo em desuso e sendo substituídos por versões correspondentes, que **adotam algum sistema de criptografia moderna** (**SSH, HTTPS, SCP/SFTP e DNS-Sec, respectivamente**), garantindo segurança ao tráfego gerado por essas aplicações.





**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia



Alice

Olá Bob!



Bob



**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia



Alice

Olá Bob!

**Algoritmo de Criptografia:** Cada letra da mensagem deve avançar N posições à frente...



Bob



**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia



Alice

Olá Bob!

**Algoritmo de Criptografia:** Cada letra da mensagem deve avançar N posições à frente...

**Chave da Criptografia:**  $N = 3$



Bob



**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia



Alice

Olá Bob!

**Algoritmo de Criptografia:** Cada letra da mensagem deve avançar N posições à frente...

**Chave da Criptografia:**  $N = 3$

Rod Ere!



Bob



**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia



Alice

Olá Bob!

**Algoritmo de Criptografia:** Cada letra da mensagem deve avançar N posições à frente...

**Chave da Criptografia:**  $N = 3$

Rod Ere!



Bob

O que Bob precisa saber para conseguir ler a mensagem de Alice?





**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia



Alice

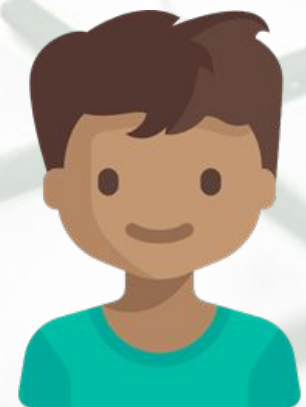
Olá Bob!

**Algoritmo de Criptografia:** Cada letra da mensagem deve avançar N posições à frente...

**Chave da Criptografia:**  $N = 3$

Como Alice informa a chave para Bob SEM que Darth também a veja?

Rod Ere!



Bob





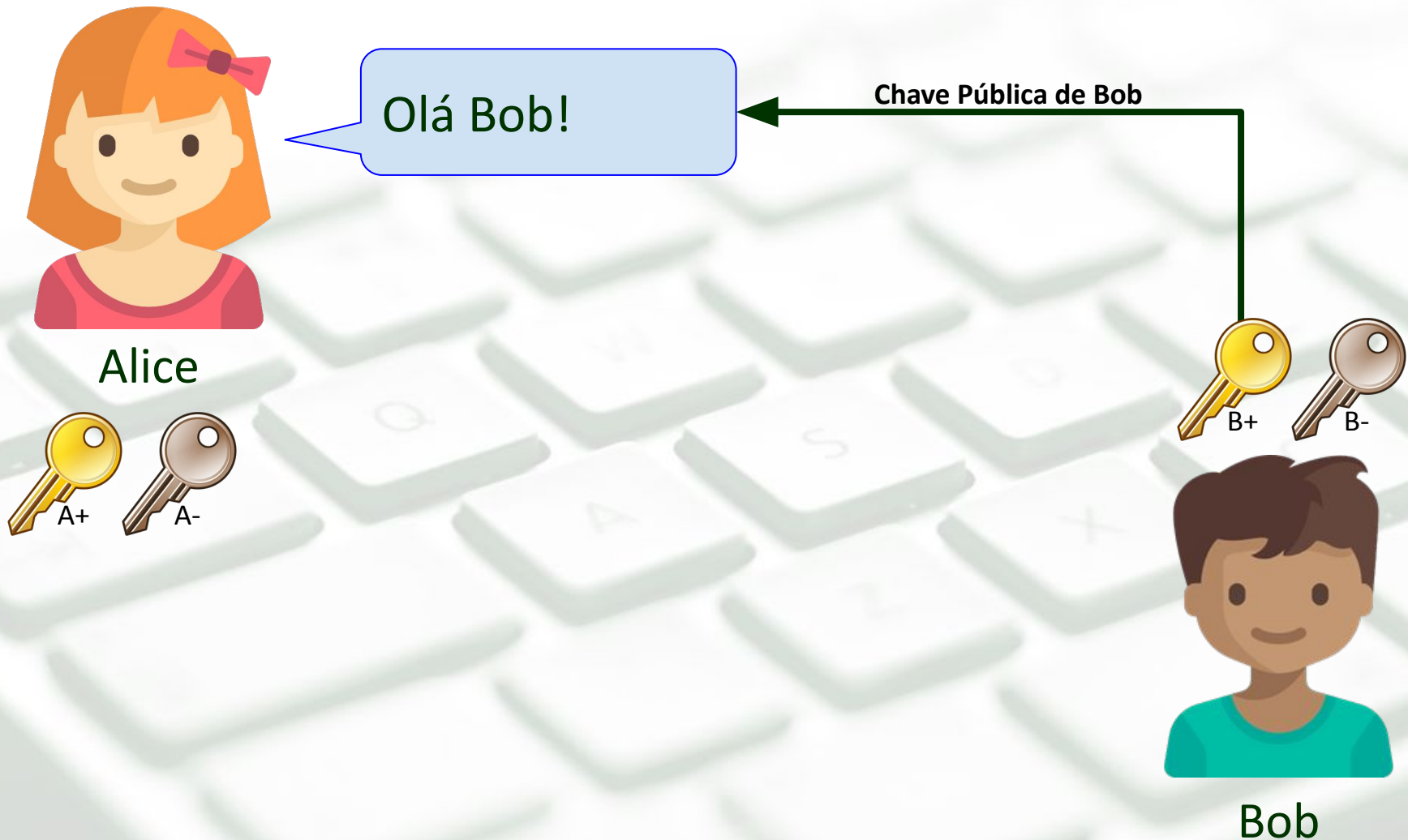
# Fundamentos de Criptografia

- Existem dois modelos básicos de criptografia...
- **Criptografia Simétrica**
  - Como mostrado no exemplo anterior...
  - A chave usada para criptografar deve ser a mesma para descriptografar (*simetria*).
- **Criptografia Assimétrica**
  - Arquitetura de Chaves Públicas (e Privadas)
  - Cada ente possui um par de chaves inter-relacionadas matematicamente.



**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

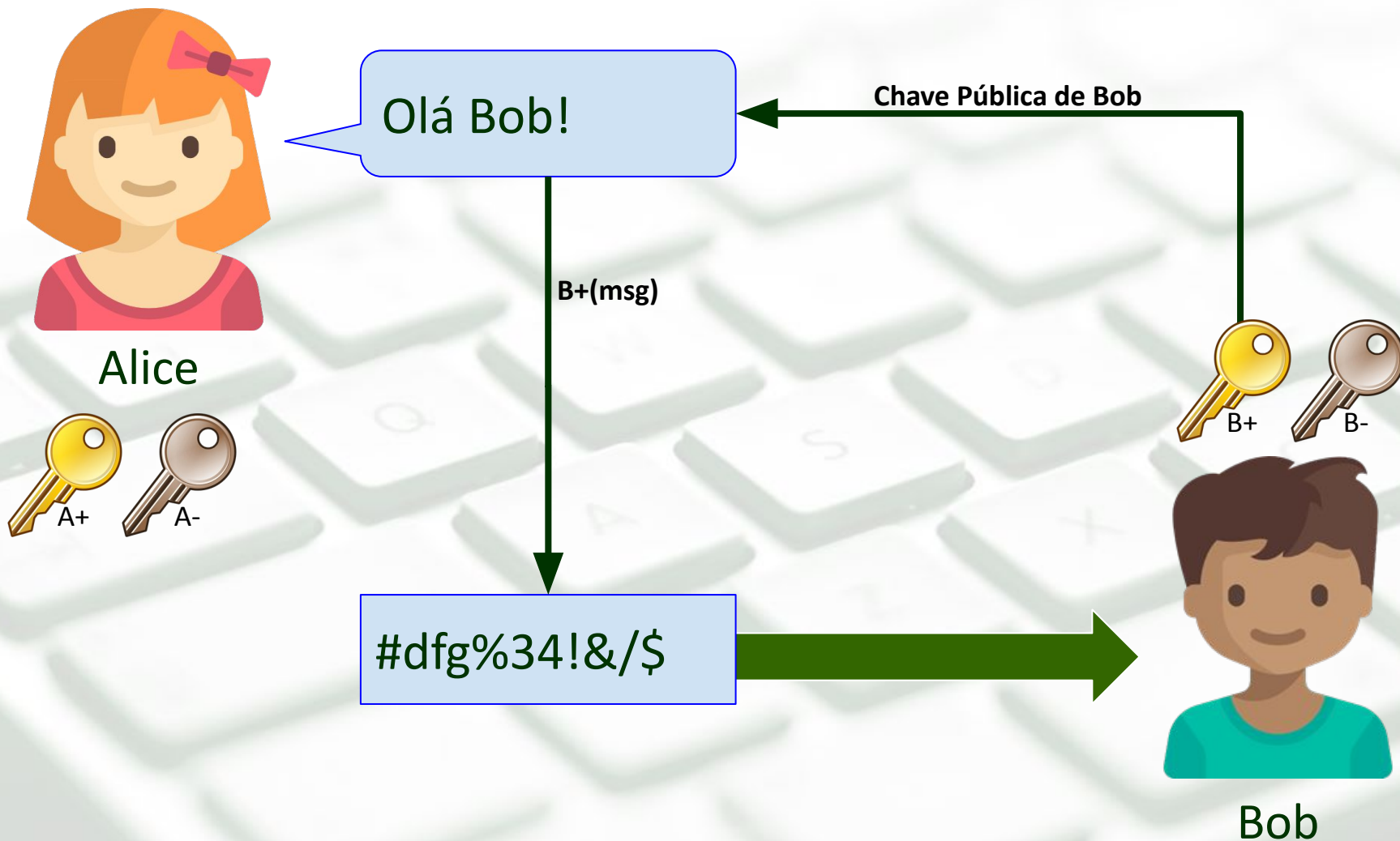
# Fundamentos de Criptografia





**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

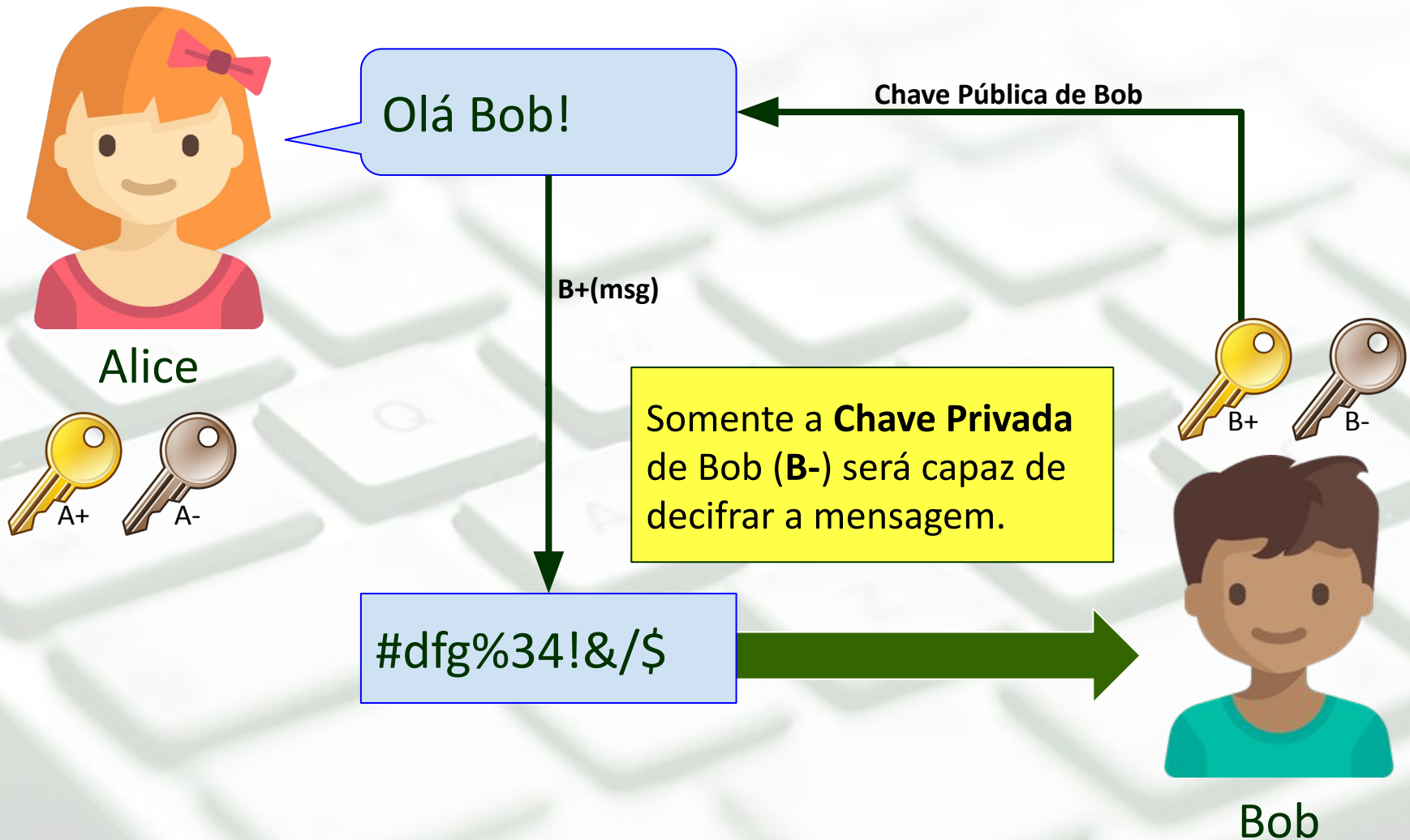
# Fundamentos de Criptografia





INSTITUTO FEDERAL  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia

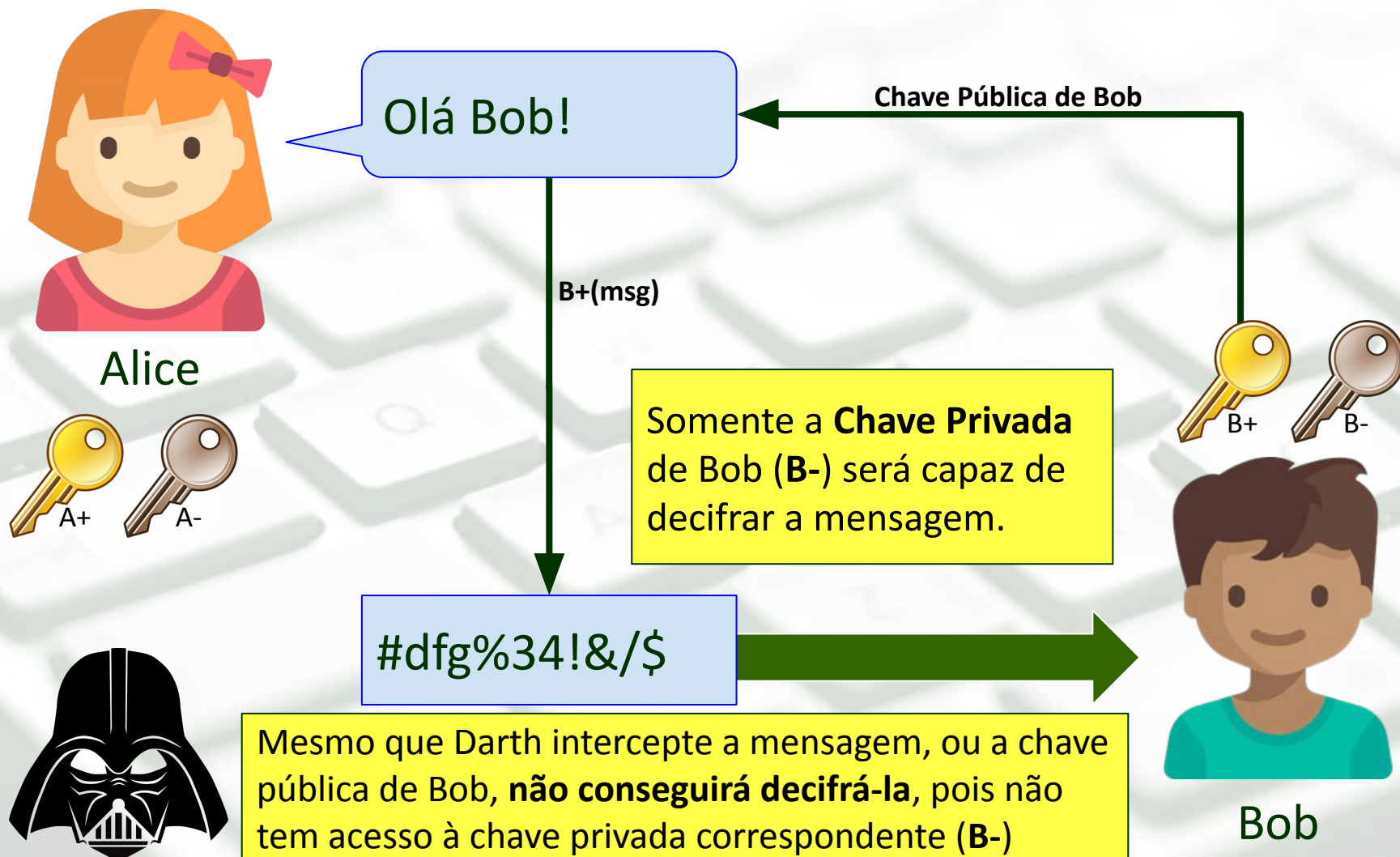






INSTITUTO FEDERAL  
Norte de Minas Gerais  
Campus Januária

# Fundamentos de Criptografia





**INSTITUTO FEDERAL**  
Norte de Minas Gerais  
Campus Januária

# *Secure Shell*

# SSH





# SSH

- **SSH (Secure SHell)** também é um **protocolo padrão** da arquitetura TCP/IP para acesso remoto a *hosts*.
- Ao contrário do Telnet, o SSH implementa **comunicação criptografada** entre o cliente e o servidor remoto.
- A autenticação é baseada em **Criptografia Assimétrica: Algoritmo RSA** (*Rivest, Shamir e Adleman*).

**Maior Segurança**



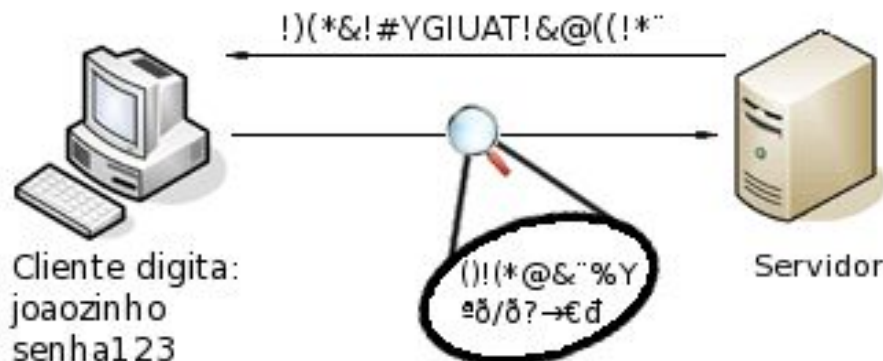


# Telnet vs. SSH

## Sessão de login sem criptografia como no telnet



## Sessão de login criptografada como no SSH





# Autenticação SSH

```
ls /etc/ssh
```

*Chave Pública C+*  
*Chave Privada C-*



## Cliente SSH

## Server SSH

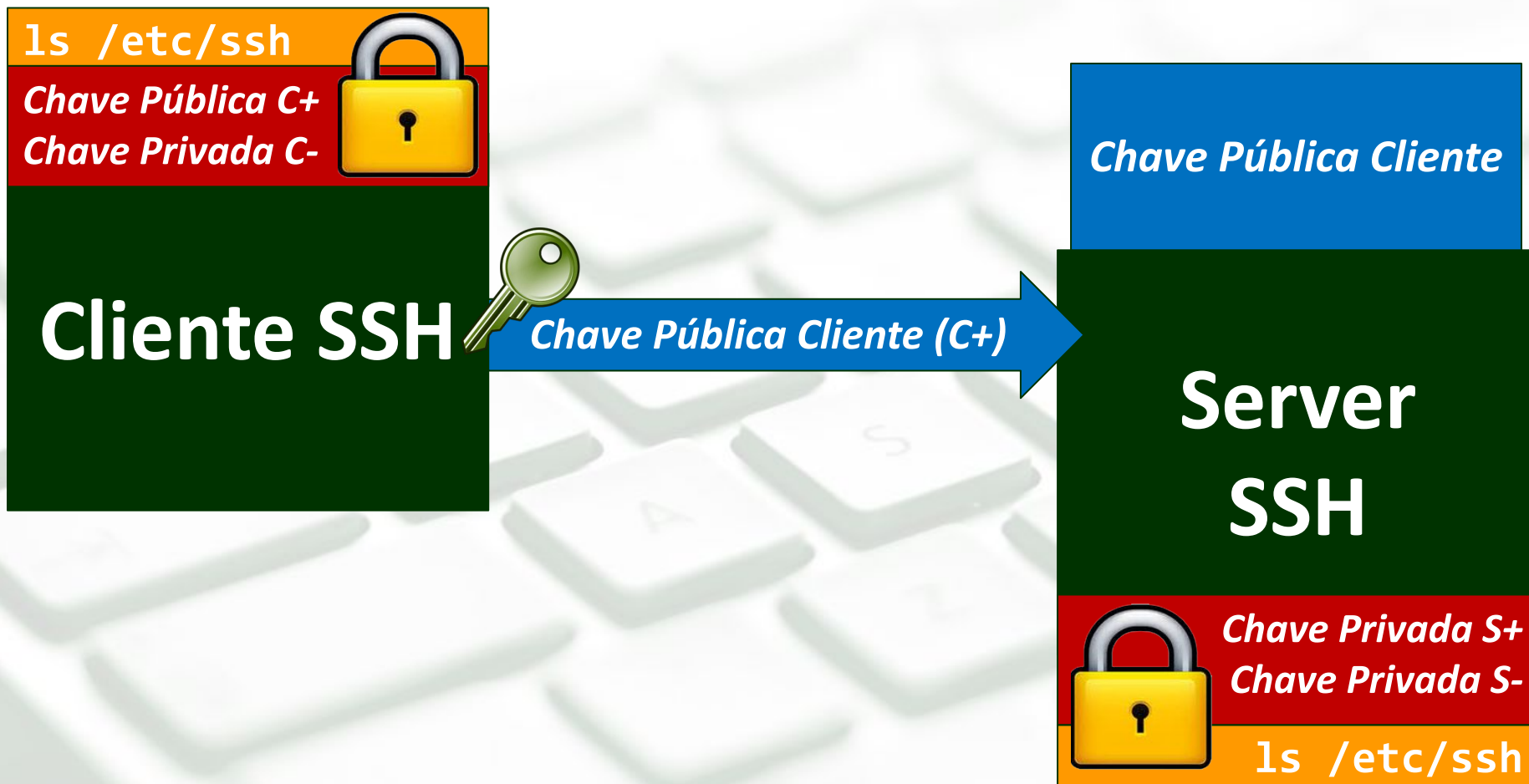


*Chave Privada S+*  
*Chave Privada S-*

```
ls /etc/ssh
```



# Autenticação SSH





# Autenticação SSH

```
ls /etc/ssh
```

*Chave Pública C+*  
*Chave Privada C-*



## Cliente SSH

*Chave Pública Cliente (C+)*

*Chave Pública Server (S+)*

*Chave Pública Server*



**Repositório de Chaves**  
~/ .ssh/known\_hosts

*Chave Pública Cliente*

## Server SSH

*Chave Privada S+*  
*Chave Privada S-*



```
ls /etc/ssh
```





# Autenticação SSH

`ls /etc/ssh`

*Chave Pública C+*  
*Chave Privada C-*



**Cliente SSH**

*Chave Pública Cliente (C+)*

*Chave Pública Server (S+)*

*Chave Pública Server*



**Repositório de Chaves**  
`~/.ssh/known_hosts`



**S+(Desafio)**

*Chave Pública Cliente*

**Server  
SSH**

*Chave Privada S+*  
*Chave Privada S-*



`ls /etc/ssh`





# Autenticação SSH

`ls /etc/ssh`

*Chave Pública C+*  
*Chave Privada C-*



**Cliente SSH**

*Chave Pública Cliente (C+)*

*Chave Pública Server*



**Repositório de Chaves**  
`~/.ssh/known_hosts`



**C+(Desafio)**

*Chave Pública Cliente*



*Chave Pública Server (S+)*

**Server SSH**

*Chave Privada S+*  
*Chave Privada S-*



`ls /etc/ssh`



**S+(Desafio)**



# Autenticação SSH

`ls /etc/ssh`

*Chave Pública C+*  
*Chave Privada C-*



**Cliente SSH**

*Chave Pública Server*



**Repositório de Chaves**  
`~/.ssh/known_hosts`



C+(Desafio)



*Chave Pública Cliente*

**Sessão SSH**  
**Chave Simétrica**

**Server SSH**

*Chave Privada S+*  
*Chave Privada S-*



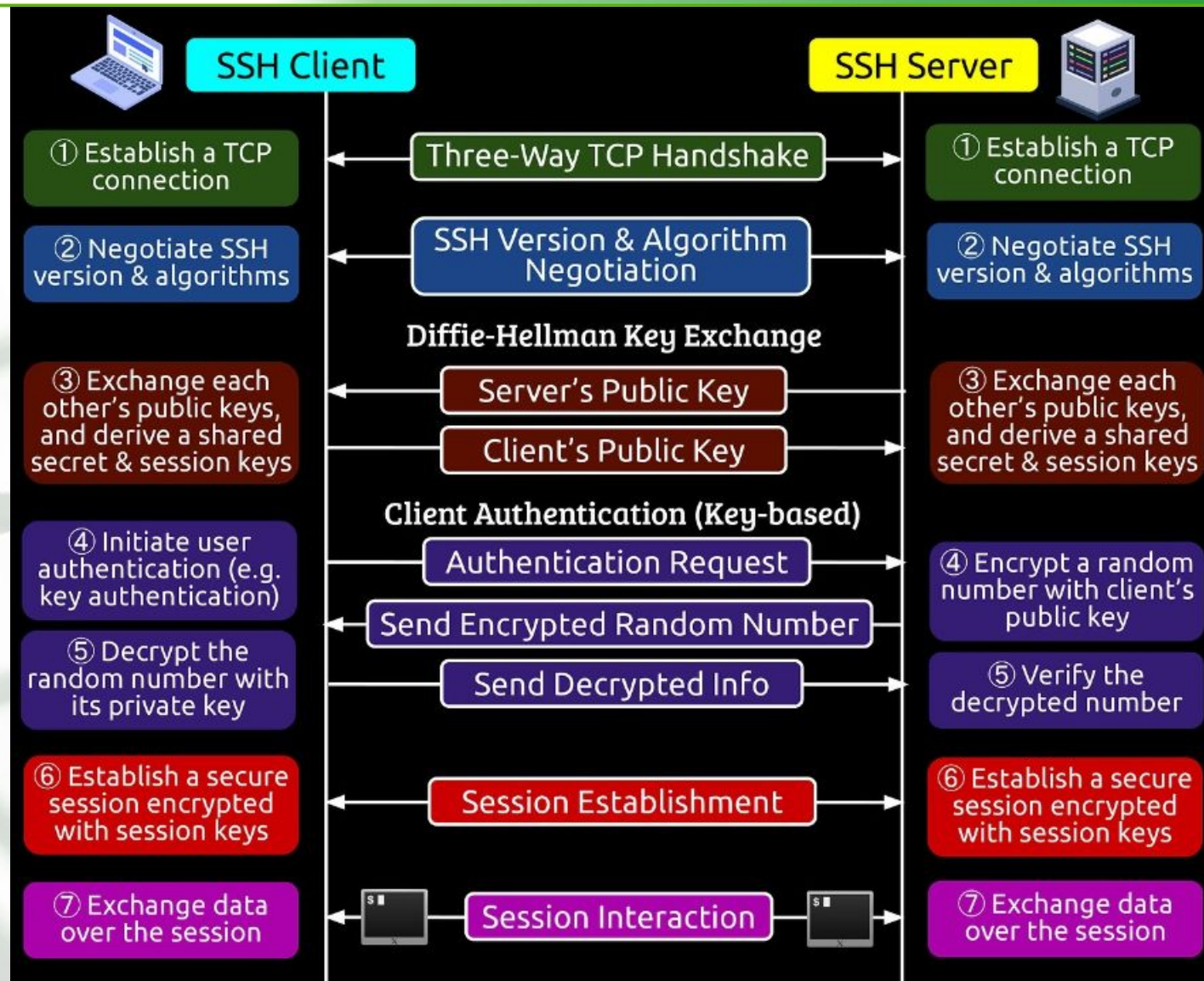
`ls /etc/ssh`



S+(Desafio)



# Autenticação SSH





# SSH

## ■ Instalação:

```
# apt-get install openssh-server  
# apt-get install openssh-client
```

## ■ Configuração:

```
# /etc/ssh/sshd_config (server)  
# /etc/ssh/ssh_config (client)
```

## ■ Ativação

```
# /etc/init.d/ssh start
```





## Laboratório 09-2

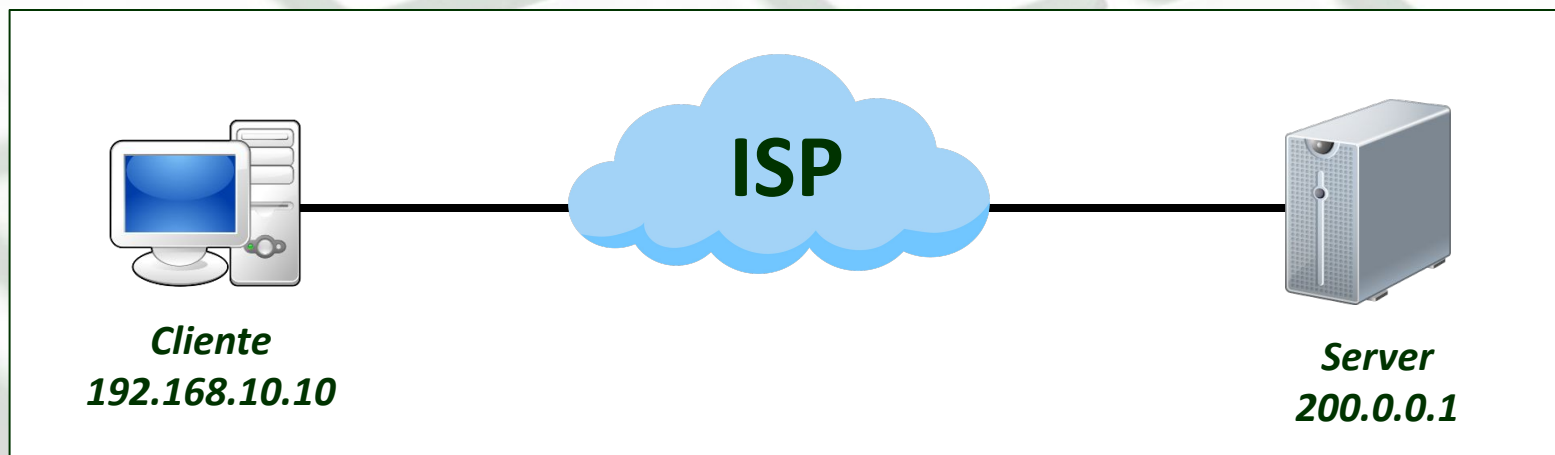
- Crie um usuário no Server.

```
# adduser nome_usuario
```

- Acesse remotamente o Server.

```
# ssh nome_usuario@200.0.0.1
```

```
# su -
```







# Chaves de Autenticação

- Abra o arquivo abaixo no **Cliente** e veja a identificação da chave pública do **Server**:

```
# nano ~/.ssh/known_hosts
```

- Exclua uma chave pública do repositório do Cliente:

```
# ssh-keygen -R 200.0.0.1
```

- As chaves dos **hosts** estão localizadas em:

```
# /etc/ssh/ssh_host_rsa_key ←  
# /etc/ssh/ssh_host_rsa_key.pub
```

***Sempre mantenha as chaves privadas bem protegidas!***



# Autenticação por Chaves

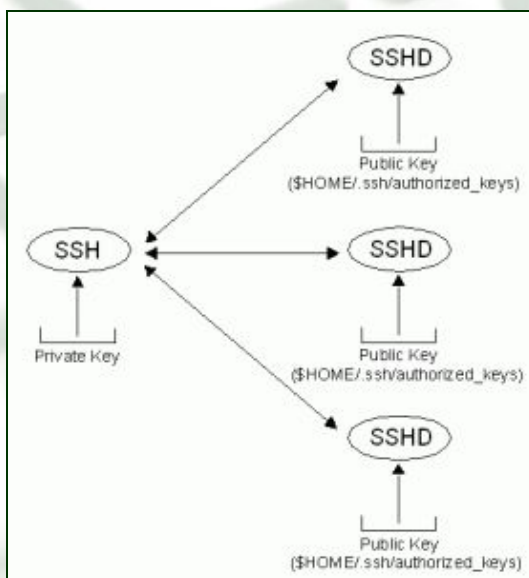
- **Autenticação por Chaves** ou **Autenticação de Duas Vias** é um método ainda mais seguro para fazer a autenticação entre duas máquinas remotas.
- Nesse método, a autenticação é feita através de **chaves assimétricas geradas pelo usuário** - ao invés de usar a sua própria senha de acesso.
  - *Evita roubo de senhas por “olhudos” de plantão e Ataques de brute-force.*
- A **chave pública** gerada pelo usuário deve ser instalada no servidor, e a **chave privada** (armazenada localmente) pode ser (ou não) protegida por uma ***passphrase***.



# Autenticação por Chaves

- Porque a autenticação por chaves é mais segura?

***Para que um invasor consiga ter acesso indevido a um servidor é necessário que ele roube a chave privada do usuário, e ainda conheça a passphrase que a decodifica.***





# Chaves de Autenticação

- Para gerar um par de chaves utilize o comando:

```
# ssh-keygen
```

- As chaves serão salvas no diretório “home” do usuário:

```
# ~/.ssh/id_rsa
```

```
# ~/.ssh/id_rsa.pub
```

- Instale a chave pública no servidor:

```
# ssh-copy-id -i ~/.ssh/id_rsa.pub login@server
```



# Chaves de Autenticação

- Recomenda-se (por simplificação) que o nome do usuário no servidor remoto seja o mesmo nome de usuário do cliente.
- As chaves públicas autorizadas a acessar uma determinada conta de usuário no servidor, são instaladas no arquivo:

```
# ~/.ssh/authorized_keys
```

*\* Também é possível instalar (copiar/colar) a chave pública do cliente diretamente no arquivo `authorized_keys`, ao invés de usar o comando `ssh-copy-id`.*





# Autenticação via Chaves

- Se a chave de usuário for criada com o nome padrão (`id_rsa` e `id_rsa.pub`), basta acessar normalmente...

```
# ssh admin@200.0.0.1
```

- Se a chave de usuário tiver um nome customizado, é necessário informar este parâmetro no acesso...

```
# ssh admin@200.0.0.1 -i ChavePrivada
```



## Laboratório 09-3

- Crie um par de chaves de autenticação SSH para acessar o usuário “Admin” no “Server”.
- O acesso remoto deve ser realizado sem a necessidade de senhas (**autenticação deve ser por chaves assimétricas**).





## Laboratório 09-4

### ■ Vamos criar nossa **primeira Instância na Cloud AWS.**

- Crie uma nova instância Debian
- Baixe a chave privada para acesso remoto.
- **VOCÊ NÃO PODE PERDER ESTE ARQUIVO.**
- Faça o primeiro acesso remoto ao usuário “admin” do IP público da Instância (utilize a chave privada baixada).
- Após isso crie e instale (manualmente) uma chave de usuário (**id\_rsa.pub**) para facilitar o acesso...
- Acesse apenas com `# ssh admin@ip_instancia`





# Laboratório 09-4

- Inspecionar os últimos acessos remotos de um servidor...

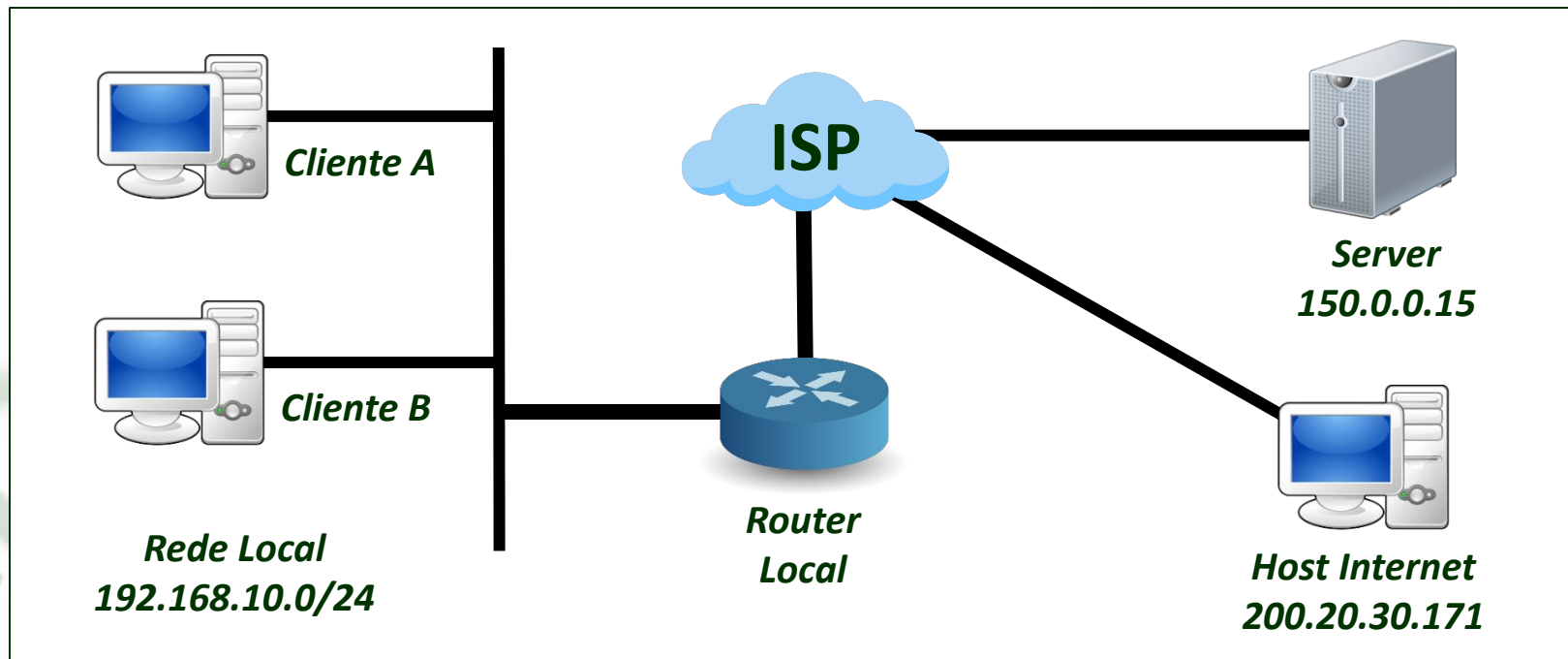
```
# sudo journalctl -u ssh | grep Accepted
```

```
admin@ip-172-31-25-137: ~  
Arquivo  Editar  Ver  Pesquisar  Terminal  Ajuda  
admin@ip-172-31-25-137:~$ sudo journalctl -u ssh | grep Accepted  
Jul 07 19:37:37 ip-172-31-25-137 sshd[530]: Accepted publickey for admin from 200.131.5.201 port 56463 s  
sh2: RSA SHA256:iFDp7tc0/hd5WvB86Qx3VfrRFQBeAZthGfFW7Y0fi+0  
Jul 07 19:41:29 ip-172-31-25-137 sshd[1513]: Accepted publickey for admin from 200.131.5.201 port 31471  
ssh2: RSA SHA256:iFDp7tc0/hd5WvB86Qx3VfrRFQBeAZthGfFW7Y0fi+0  
Jul 10 17:02:32 ip-172-31-25-137 sshd[536]: Accepted publickey for admin from 138.122.149.32 port 16961  
ssh2: RSA SHA256:iFDp7tc0/hd5WvB86Qx3VfrRFQBeAZthGfFW7Y0fi+0  
Jul 10 17:10:41 ip-172-31-25-137 sshd[569]: Accepted publickey for admin from 138.122.149.32 port 16811  
ssh2: RSA SHA256:iFDp7tc0/hd5WvB86Qx3VfrRFQBeAZthGfFW7Y0fi+0  
Jul 10 17:14:33 ip-172-31-25-137 sshd[661]: Accepted publickey for admin from 138.122.149.32 port 16441  
ssh2: RSA SHA256:aYzTMA0mwJGa0TMMmzzXI67T+PA+lfTejPBemqrH5B0  
Jul 10 17:14:57 ip-172-31-25-137 sshd[677]: Accepted publickey for admin from 138.122.149.32 port 17070  
ssh2: RSA SHA256:aYzTMA0mwJGa0TMMmzzXI67T+PA+lfTejPBemqrH5B0  
admin@ip-172-31-25-137:~$
```





# Laboratório 09-5



- A partir do “Cliente A”, use o SSH (com autenticação via chaves) para subir a porta 80 em modo escuta no Server (use o netcat).
- A partir de um “Host qualquer na Internet”, use o SSH (login+senha) para “invadir” o “Router Local” e criar um sniffer (captura de tráfego) para gerar e salvar em arquivo todos os pacotes que trafegam ali.
- A partir do “Cliente B”, conecte-se a porta 80 do server, enviando e recebendo informações.
- Através do Wireshark, inspecione o arquivo gerado pelo *sniffer* do Router Local.





# Boas Práticas SSH

```
# nano /etc/ssh/sshd_config
```

```
# Alterar a porta padrão (22) do serviço
```

```
Port 1025
```

```
# Endereço de escuta da conexão
```

```
ListenAddress 192.168.10.1
```

```
# Desabilite login do usuário root (apenas por chaves já é padrão!)
```

```
PermitRootLogin prohibit-password || no
```

```
# Desabilite login de usuários por senha (força que a autenticação aconteça apenas por chaves)
```

```
PasswordAuthentication no
```

```
# Nº Tentativas de conexão sem sucesso (5), % de recusa de chamadas após as 5 iniciais, Nº Máximo Total até bloqueio total.
```

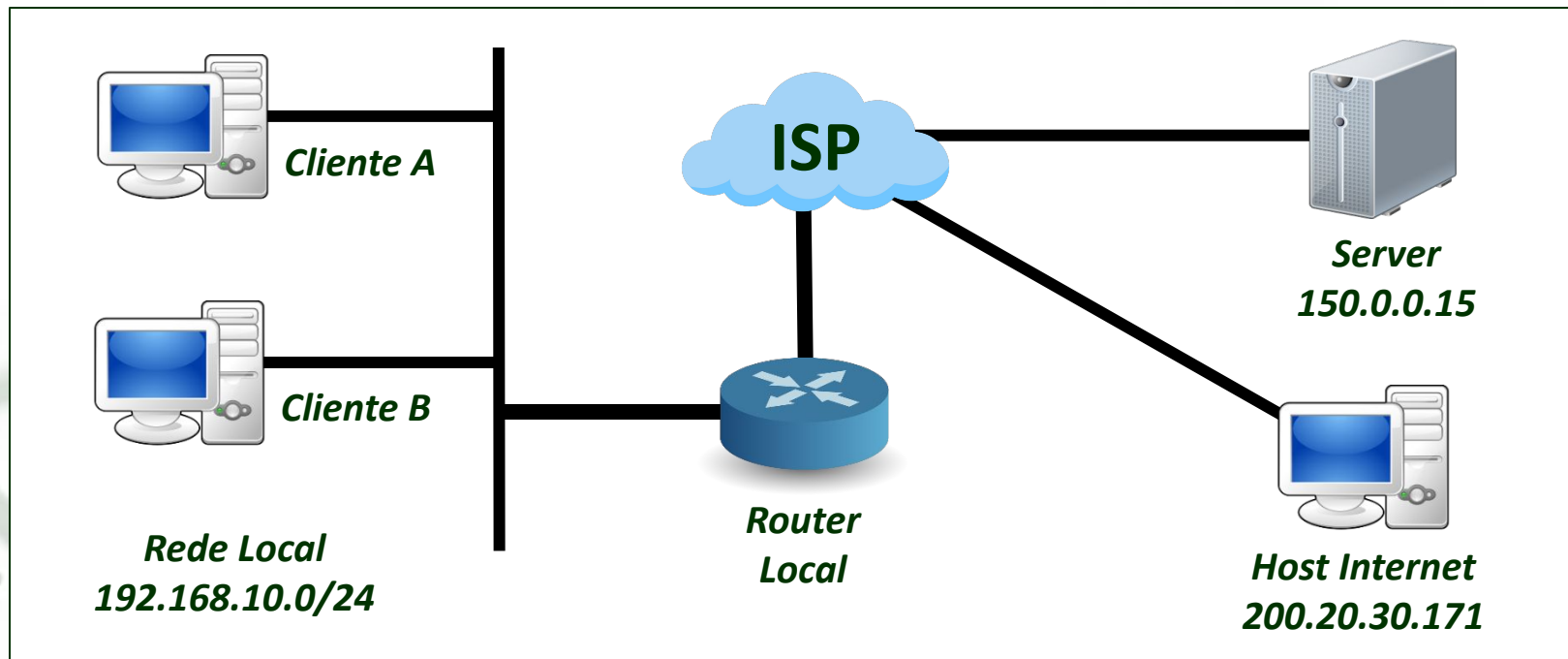
```
MaxStartups 5:70:8
```

```
# Permite apenas conexões para os usuários explicitamente indicados.
```

```
AllowUsers adminIfnmg
```



# Laboratório 09-6



- Aprimore a segurança do Lab09-5:
  - Conexões SSH para o “Router Local” devem ser aceitas APENAS provenientes da Rede Local, e específicas para um usuário bem definido (p.ex.:adminIfnmg)
  - Conexões SSH para o “Server” devem ser feitas EXCLUSIVAMENTE via chaves assimétricas, porta padrão deve ser a 5001, e sendo 5 tentativas no máximo.



# Token Authentication

- Podemos implementar uma outra camada de segurança para autenticação de usuários através de **Tokens mutáveis**.
- Essa técnica é conhecida como MFA (Multi-Factor Authentication) ou 2FA.
- Combinações MFA:
  - Senha + Token
  - Chave + Token





## Laboratório 09-7

- Aproveite a instância criada no Lab. 08-6 e incremente a sua segurança configurando a autenticação 2FA.
- Acesse-a e instale a ferramenta de token:  
***Google-Authenticator***

```
# apt update  
# apt install libpam-google-authenticator -y
```

- Altere os arquivos a seguir...



# Laboratório 09-7

```
# nano /etc/pam.d/sshd
```

```
# ATENÇÃO! Se for usar Chave+Token, comente a linha  
abaixo, se for usar Senha+Token, deixe como está...
```

```
@include common-auth
```

```
# Token Authentication via Google PAM
```

```
auth required pam_google_authenticator.so
```





## Laboratório 09-7

```
# nano /etc/ssh/sshd_config
```

```
# Habilita a autenticação via token no SSH
```

```
ChallengeResponseAuthentication yes
```

```
ou...
```

```
KbdInteractiveAuthentication yes
```

```
UsePAM yes
```

```
# SE o modelo for CHAVE+TOKEN, adicionar essa  
instrução no final do arquivo...
```

```
AuthenticationMethods publickey,keyboard-interactive
```



## Laboratório 09-7

- Acesse a conta do usuário que fará a autenticação por token, e configure o APP externo...

```
# su admin
```

```
# Instale um APP para visualizar os tokens gerados, p.ex.: Google Authenticator
```

```
# Através deste APP, escaneie o QR code gerado pelo comando abaixo...
```

```
# google-authenticator
```

```
# Recomendado (Y)es para todas as opções seguintes...
```



# Laboratório 09-7

FA, e

adriano@adriano-notebook: ~/Dropbox/aws

Arquivo Editar Ver Pesquisar Terminal Ajuda

```
adriano@adriano-notebook:~/Dropbox/aws$ ssh -i chaveAWS.pem admin@18.234.125.54  
Verification code: 
```

#

#

g

#

p

#

#

seguintes...

Google Authenticator

Pesquisar...

Google: elisa.g.beckett@gmail.com

361 976

Google: hikingfan@gmail.com

152 781

Google: surfingfan@gmail.com

324 833



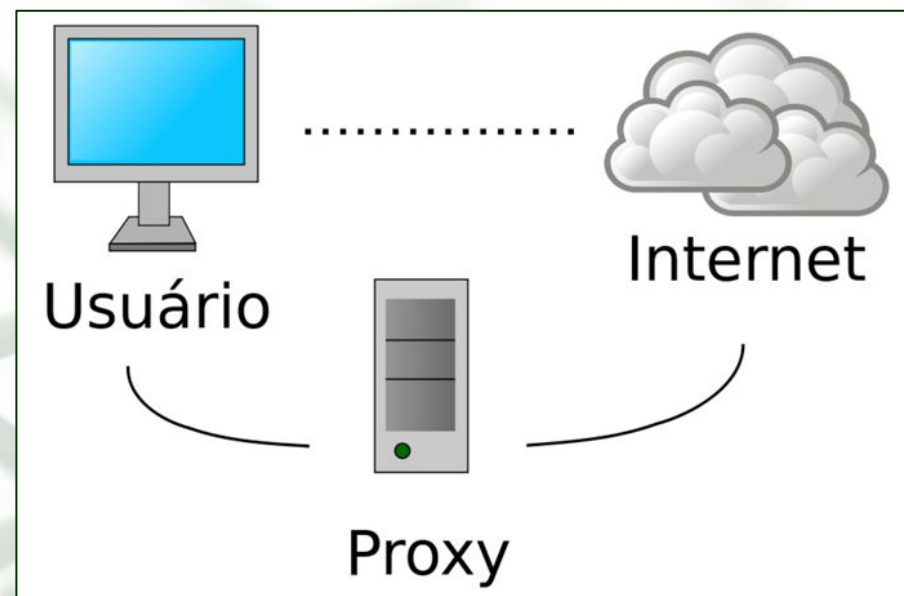
# Seminário Individual

## PROXYs

*O que são e para que servem?*

*Proxy Socks vs. Proxy HTTP*

*Proxy vs. VPNs*



**LINK para Vídeo Introdotório**



# Referências

- **Guia Foca GNU/Linux.**  
Disponível em <http://www.guiafoca.org/>
- MORIMOTO, Carlos E; **Servidores Linux – Guia Prático.**
- [Set Up SSH Two-Factor Authentication \(2FA\) on Debian 11 Server.](#)