SEG12 - Semana 1 - Sessão 10

Francisco Marcelo, Marcelo Karam e Felipe Scarel

06-08-2018

Servidor Web

As atividades desta sessão serão realizadas na máquina virtual *Server_Linux*, com pequenas exceções apontadas pelo enunciado dos exercícios.

O objetivo de um servidor web é, em essência, servir conteúdo para a *world wide web*. Esse objetivo é atingido servindo requisições enviadas ao servidor através do protocolo HTTP, bem como protocolos relacionados. Nesta sessão iremos instalar e configurar o servidor web Apache, um dos mais populares servidores HTTP *open source* do mundo.

1) Instalação do servidor web Apache

Instale o servidor web Apache (pacote `apache2`). Teste o funcionamento da instalação acessando a página web a partir de qualquer navegador (seja na máquina física, *Client_Linux* ou *Win7-padrao*).

1. Instale o servidor web Apache:

```
# apt-get install apache2
```

2. Vamos testar o funcionamento acessando o IP do servidor *Server_Linux* através de um navegador instalado na máquina *Win7-padrao*:

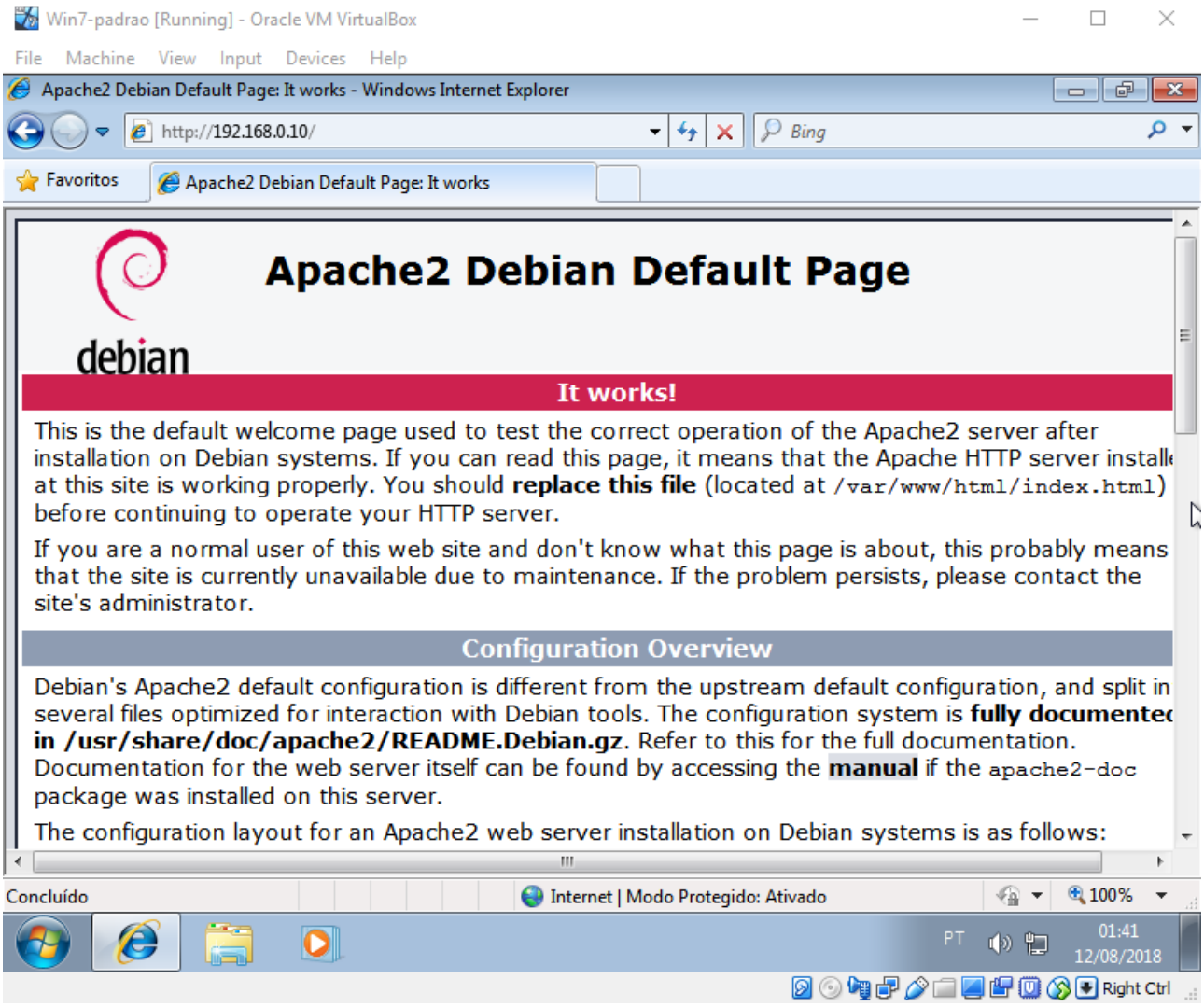

Figura 6: Apache instalado com sucesso

2) Configuração de *virtualhosts*

Virtualhosts, ou servidores virtuais, podem ser utilizados nos seguintes casos comuns:

- Hospedar múltiplos *sites* diferentes em um mesmo endereço IP;
- Hospedar múltiplos *sites*, cada um com seu IP específico.

Destes, o primeiro cenário é o mais usual, e o que será abordado nesta atividade.

No servidor web Apache instalado em nosso servidor Debian, os arquivos de configuração de todos os *sites* devem ser colocados no pasta `/etc/apache2/sites-available`. Esses *sites* podem estar ativos ou inativos:

- Para ativar um *site*, basta criar um *link* simbólico do arquivo original para a pasta `/etc/apache2/sites-enabled` e recarregar o servidor Apache. Esse *link* pode ser criado manualmente, ou através do comando `a2ensite` ("*Apache 2 enable site*").
- Para desabilitar um *site*, toma-se o caminho oposto: apague o *link* simbólico da pasta `/etc/apache2/sites-enabled`, ou use o comando `a2dissite` ("*Apache 2 disable site*").

Relembrando a sessão 7 — DNS e NFS, criamos duas entradas `CNAME` apontando para a máquina *Server_Linux*, quais sejam:

```
# cat /etc/bind/db.empresa.com.br | grep 'CNAME *servidor'
www       IN    CNAME         servidor
meusite   IN    CNAME         servidor
```

1. Crie dois *virtualhosts* na máquina *Server_Linux*, um respondendo requisições enviadas para `www.empresa.com.br` e outro para `meusite.empresa.com.br`.
2. Crie pastas específicas para cada *virtualhost* dentro do diretório `/var/www`.
3. Crie arquivos `index.html` na raiz dessas pastas que identifiquem cada um dos *virtualhosts*.
4. Acesse os nomes de domínio a partir de um navegador (seja na máquina física, *Client_Linux* ou *Win7-padrao*) e verifique que suas configurações surtiram efeito.

Siga os passos abaixo:

1. Crie o arquivo de *virtualhost* `/etc/apache2/sites-available/www.conf` para o domínio `www.empresa.com.br`, como se segue:

   ```
   <VirtualHost *:80>
     ServerAdmin webmaster@empresa.com.br
     ServerName www.empresa.com.br
     DocumentRoot /var/www/www

     ErrorLog ${APACHE_LOG_DIR}/www-error.log
     CustomLog ${APACHE_LOG_DIR}/www-access.log combined
   </VirtualHost>
   ```

2. Faça o mesmo para o domínio `meusite.empresa.com.br`, editando o arquivo `/etc/apache2/sites-available/meusite.conf`:

```
<VirtualHost *:80>
  ServerAdmin webmaster@empresa.com.br
  ServerName meusite.empresa.com.br
  DocumentRoot /var/www/meusite

  ErrorLog ${APACHE_LOG_DIR}/meusite-error.log
  CustomLog ${APACHE_LOG_DIR}/meusite-access.log combined
</VirtualHost>
```

3. Crie a pasta e arquivo `index.html` para o *virtualhost* `www.empresa.com.br`:

```
# mkdir /var/www/www

# cat /var/www/www/index.html
<html>
  <head>
    <title>Test</title>
</head>
  <body>
    Welcome to www.empresa.com.br
  </body>
</html>
```

4. Faça o mesmo para o domínio `meusite.empresa.com.br`:

```
# mkdir /var/www/meusite

# cat /var/www/meusite/index.html
<html>
  <head>
    <title>Test</title>
</head>
  <body>
    Welcome to meusite.empresa.com.br
  </body>
</html>
```

5. Habilite ambos os *virtualhosts* e recarregue a configuração do Apache:

```
# a2ensite www
Enabling site www.
To activate the new configuration, you need to run:
  service apache2 reload

# a2ensite meusite.conf
Enabling site meusite.
To activate the new configuration, you need to run:
  service apache2 reload

# systemctl reload apache2
```

6. Vamos testar o funcionamento do *virtualhost* `www.empresa.com.br` através de um navegador instalado na máquina *Win7-padrao*:

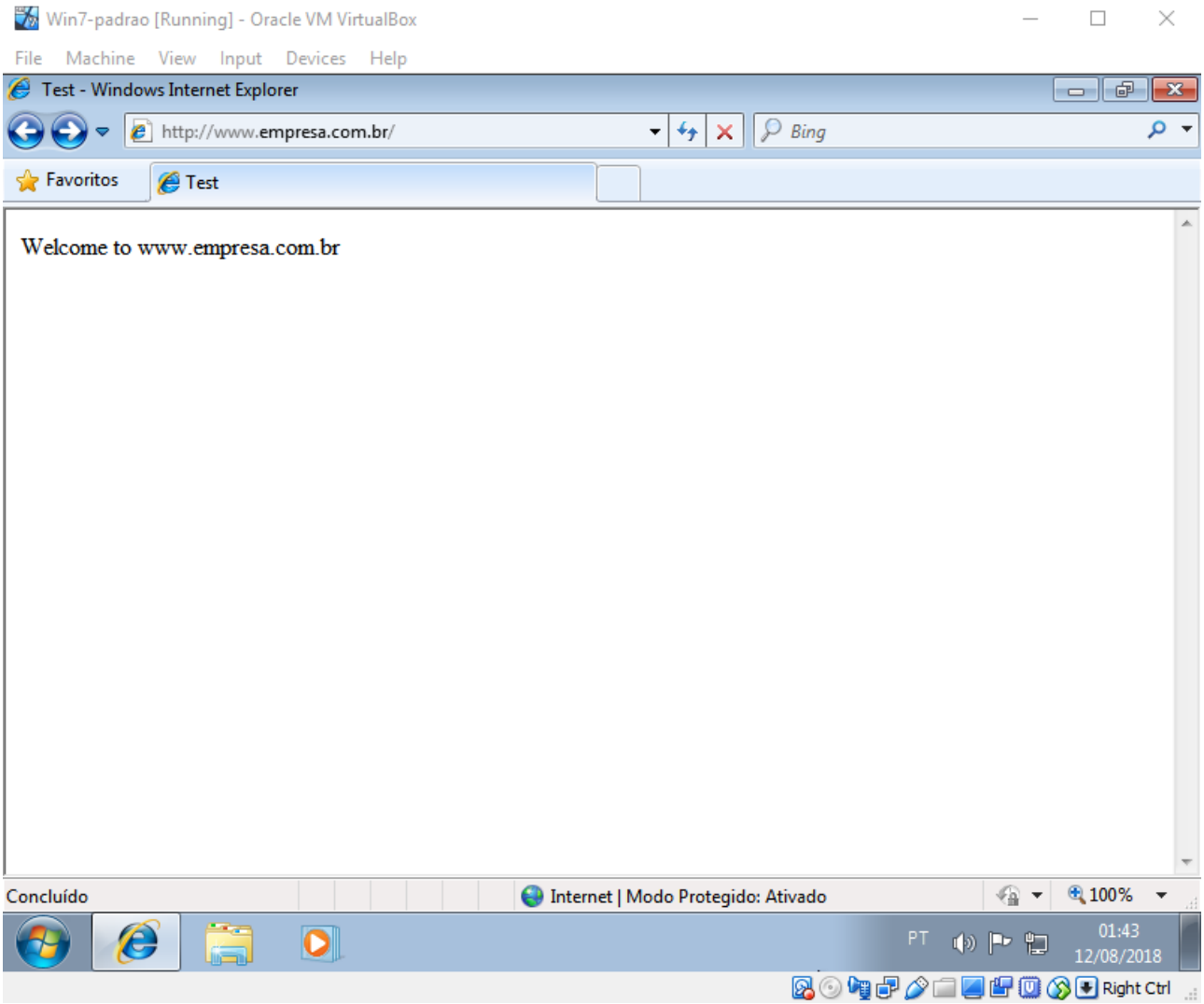

Figura 7: Virtualhost www.empresa.com.br acessível

7. E, novamente, repetiremos o teste para o *virtualhost* `meusite.empresa.com.br`:

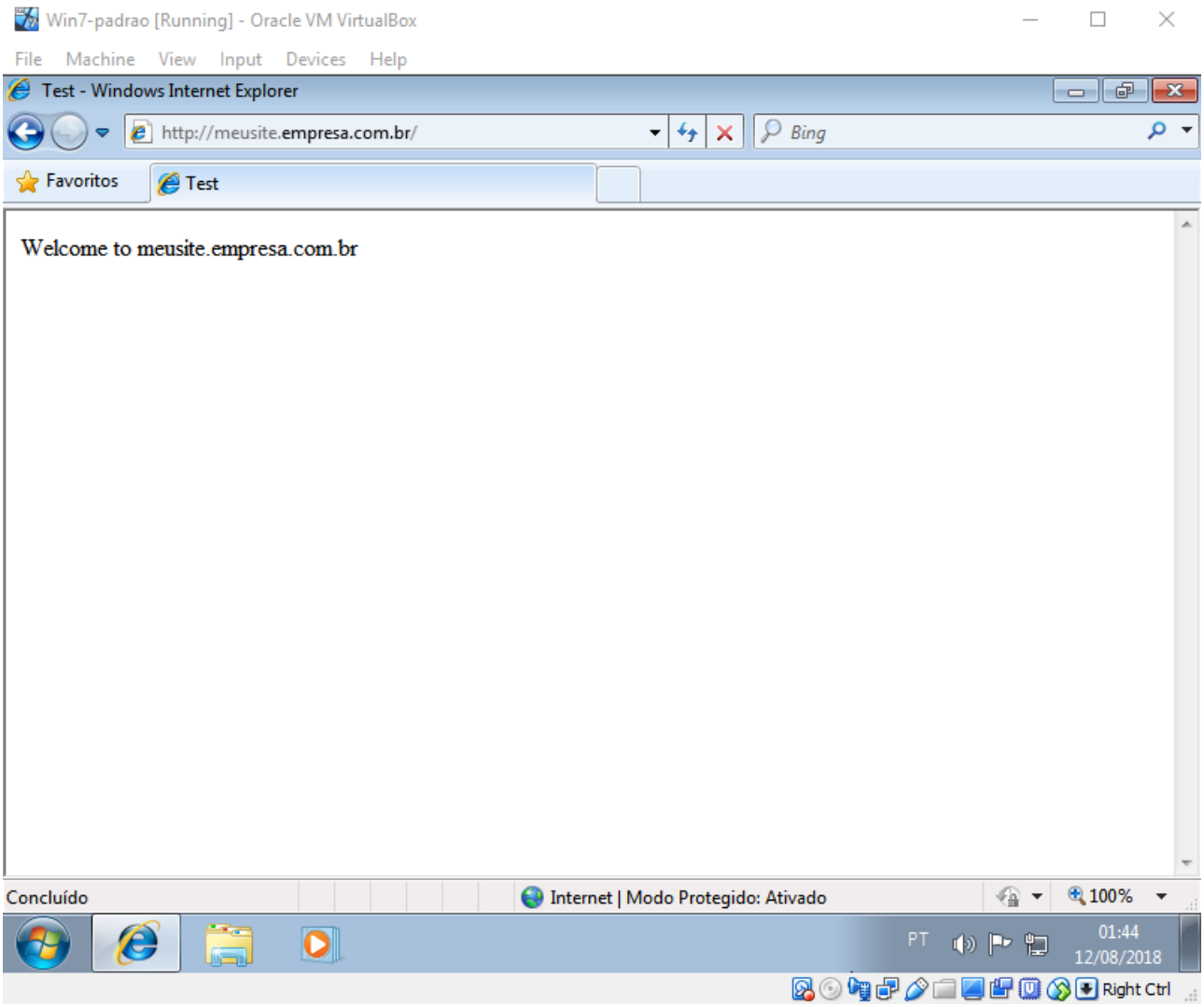

Figura 8: Virtualhost meusite.empresa.com.br acessível

3) Configuração de criptografia SSL

O protocolo HTTP não possui nenhum recurso de criptografia e, por consequência, todo o tráfego de rede gerado entre cliente e servidor poderia ser visualizado por um atacante. Para aumentar a segurança de aplicações web, é interessante habilitar o suporte a conexões cifradas através do *Secure Sockets Layer* (SSL).

1. Habilite o módulo SSL do Apache através do comando `a2enmod` ("*Apache 2 enable module*").
2. Crie um certificado auto-assinado RSA de 4096 bits para o *virtualhost* `meusite.empresa.com.br`, com validade de um ano. Armazene a chave pública na pasta `/etc/ssl/certs`, e a chave privada em `/etc/ssl/private`. Tenha atenção às permissões de arquivo e usuário/grupo dono.
3. Configure o *virtualhost* `meusite.empresa.com.br` para utilizar o protocolo HTTPS em qualquer conexão. Redirecione qualquer conexão sem criptografia direcionada à porta 80/HTTP para a porta 443/HTTPS.
4. Acesse o domínio `meusite.empresa.com.br` a partir de um navegador (seja na máquina física, *Client_Linux* ou *Win7-padrao*) e verifique que suas configurações surtiram efeito.

Siga os passos abaixo:

1. Habilite o módulo SSL no Apache:

```
# a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create
self-signed certificates.
To activate the new configuration, you need to run:
  service apache2 restart
```

2. Gere o certificado auto-assinado usando o comando `openssl`. Para gerar um par de chaves com os parâmetros solicitados, basta usar as opções `-days 365` e `-newkey rsa:4096`. Observe, ainda, que a permissão da chave privada gerada pelo comando é muito leniente — utilize o comando `chmod 600` para corrigir isso.

```
# openssl req -x509 -nodes -days 365 -newkey rsa:4096 -keyout
/etc/ssl/private/meusite.key -out /etc/ssl/certs/meusite.crt
Generating a 4096 bit RSA private key
.....................................................++
.....................................................................
.....................++
writing new private key to '/etc/ssl/private/meusite.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:BR
State or Province Name (full name) [Some-State]:DF
Locality Name (eg, city) []:Brasilia
Organization Name (eg, company) [Internet Widgits Pty Ltd]:RNP
Organizational Unit Name (eg, section) []:ESR
Common Name (e.g. server FQDN or YOUR name) []:meusite.empresa.com.br
Email Address []:webmaster@empresa.com.br

# chmod 600 /etc/ssl/private/meusite.key

# ls -ld /etc/ssl/private/meusite.key
-rw------- 1 root root 3272 Ago 12 02:09 /etc/ssl/private/meusite.key
```

3. Edite o arquivo `/etc/apache2/sites-available/meusite.conf`, habilitando o redirecionamento de requisições da porta 80/HTTP para a porta 443/HTTPS, ativando a *engine* SSL e informando o caminho para as chaves pública e privada do *virtualhost*:

```
<VirtualHost *:80>
  ServerName meusite.empresa.com.br
  Redirect permanent / https://meusite.empresa.com.br/
</VirtualHost>

<VirtualHost *:443>
  ServerAdmin webmaster@empresa.com.br
  ServerName meusite.empresa.com.br
  DocumentRoot /var/www/meusite

  SSLEngine On
  SSLCertificateFile    /etc/ssl/certs/meusite.crt
  SSLCertificateKeyFile /etc/ssl/private/meusite.key

  ErrorLog ${APACHE_LOG_DIR}/meusite-error.log
  CustomLog ${APACHE_LOG_DIR}/meusite-access.log combined
</VirtualHost>
```

4. Reinicie o Apache para que ele ative o módulo SSL e releia o arquivo de configuração do *virtualhost*:

```
# systemctl restart apache2
```

5. Agora, basta testar. Acessamos a URL http://meusite.empresa.com.br e, de fato, o redirecionamento para HTTPS funcionou. Somos apresentados a uma tela de certificado inválido:

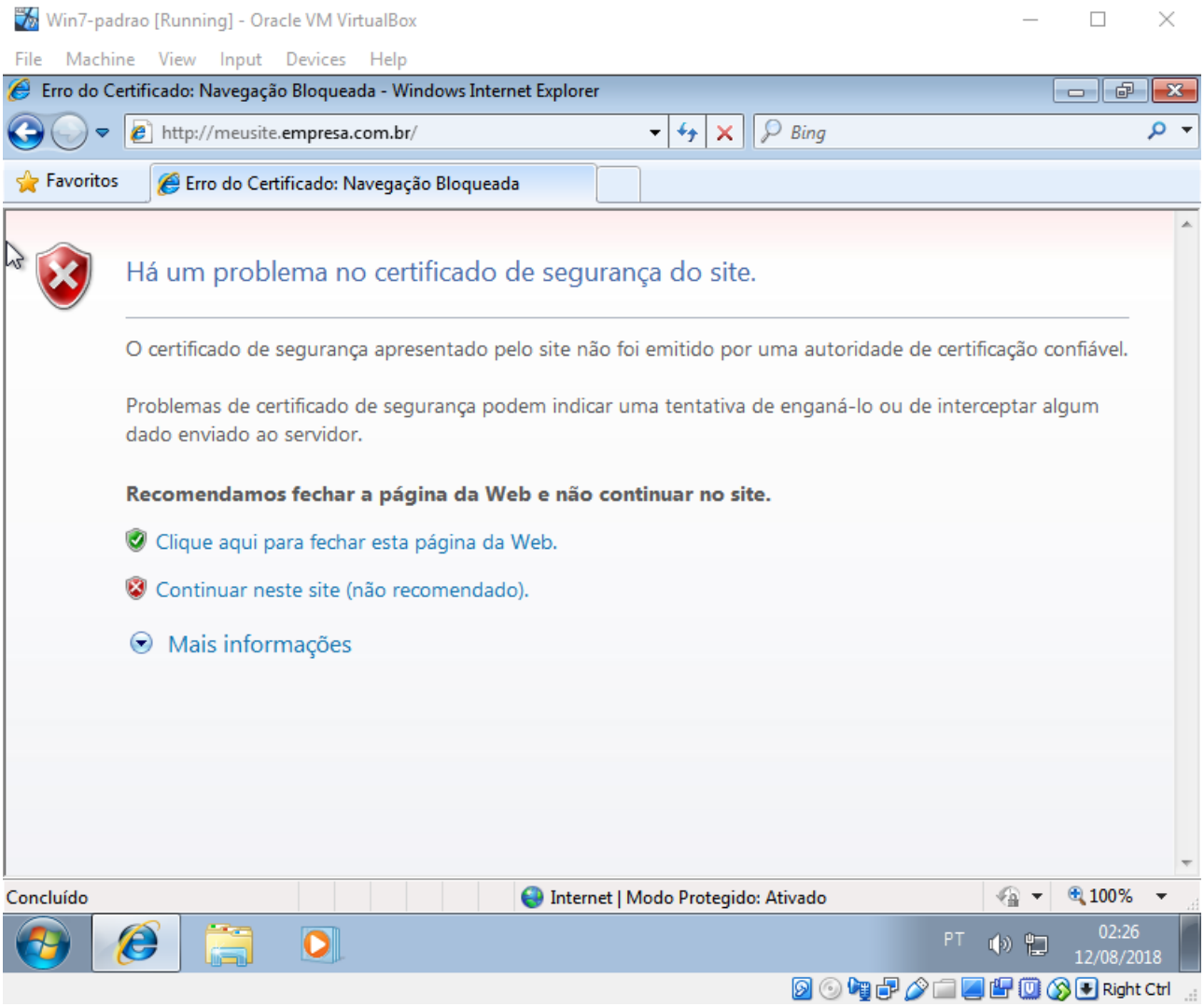

Figura 9: Alerta de certificado inválido

6. Esse erro é esperado, já que o certificado SSL que estamos utilizando é auto-assinado, e não pode ser verificado pelas autoridades certificadoras raiz instaladas no navegador. Após clicar em "Continuar neste *site*", conseguimos acessar a página objetivada:

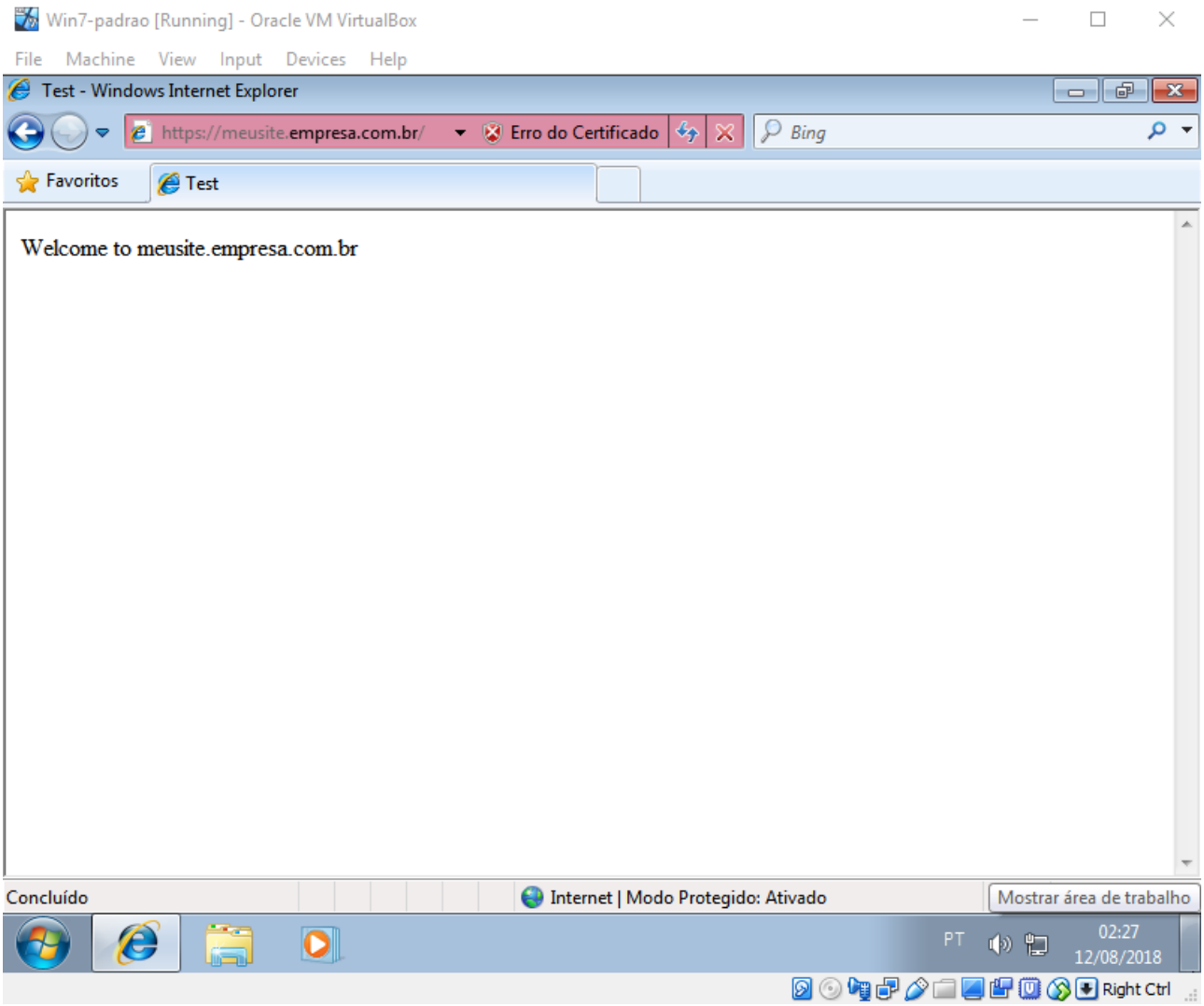

Figura 10: Acesso via HTTPS a meusite.empresa.com.br

4) Autenticação e acesso a conteúdo restrito usando LDAP

Autenticação de usuários, especialmente em áreas sensíveis de um *site*, é integral à configuração de segurança de servidores web. Em particular, estamos interessados em habilitar autenticação para uma área restrita do *virtualhost* `meusite.empresa.com.br`.

1. Habilite o módulo de autenticação LDAP do Apache, `authnz_ldap`, através do comando `a2enmod`.
2. Crie uma pasta `/restrito` dentro da raiz do *virtualhost*. Dentro dessa pasta, crie um arquivo `index.html` que possa ser usado para testar a configuração.
3. Configure o *virtualhost* para requerer autenticação quando um usuário tentar acessar a URL `meusite.empresa.com.br/restrito`. Exija que o cliente forneça uma combinação de usuário/senha válida e existente na base LDAP local.
4. Acesse a URL `meusite.empresa.com.br/restrito` a partir de um navegador (seja na máquina física, *Client_Linux* ou *Win7-padrao*) e verifique que suas configurações surtiram efeito.

Siga os passos abaixo:

1. Habilite o módulo de autenticação LDAP no Apache:

```
# a2enmod authnz_ldap
Considering dependency ldap for authnz_ldap:
Enabling module ldap.
Enabling module authnz_ldap.
To activate the new configuration, you need to run:
  service apache2 restart
```

2. Crie o diretório `/var/www/meusite/restrito`, e dentro dele edite um arquivo `index.html` que indique com clareza que foi possível obter acesso à área restrita:

```
# mkdir /var/www/meusite/restrito

# cat /var/www/meusite/restrito/index.html
<html>
  <head>
    <title>Test auth</title>
</head>
  <body>
    Restricted area for meusite.empresa.com.br
  </body>
</html>
```

3. Edite o arquivo `/etc/apache2/sites-available/meusite.conf`, habilitando autenticação via LDAP caso o cliente solicite acesso ao diretório `/var/www/meusite/restrito`. Tenha especial atenção ao configurar o filtro da URL LDAP:

```
<VirtualHost *:80>
  ServerName meusite.empresa.com.br
  Redirect permanent / https://meusite.empresa.com.br/
</VirtualHost>

<VirtualHost *:443>
  ServerAdmin webmaster@empresa.com.br
  ServerName meusite.empresa.com.br
  DocumentRoot /var/www/meusite

  SSLEngine On
  SSLCertificateFile    /etc/ssl/certs/meusite.crt
  SSLCertificateKeyFile /etc/ssl/private/meusite.key

  <Directory /var/www/meusite/restrito>
    AuthType basic
    AuthBasicProvider ldap
    AuthName "meusite LDAP login"
    AuthLDAPURL ldap://127.0.0.1/ou=People,dc=empresa,dc=com,dc=br?uid?sub?
(objectClass=posixAccount)
    AuthLDAPBindDN cn=admin,dc=empresa,dc=com,dc=br
    AuthLDAPBindPassword rnpesr
    Require valid-user
  </Directory>

  ErrorLog ${APACHE_LOG_DIR}/meusite-error.log
  CustomLog ${APACHE_LOG_DIR}/meusite-access.log combined
</VirtualHost>
```

4. Reinicie o Apache para que ele ative o módulo de autenticação LDAP e releia o arquivo de configuração do *virtualhost*:

```
# systemctl restart apache2
```

5. Agora, basta testar. Acessamos a URL https://meusite.empresa.com.br/restrito e imediatamente fomos apresentados a uma tela de autenticação solicitando usuário e senha:

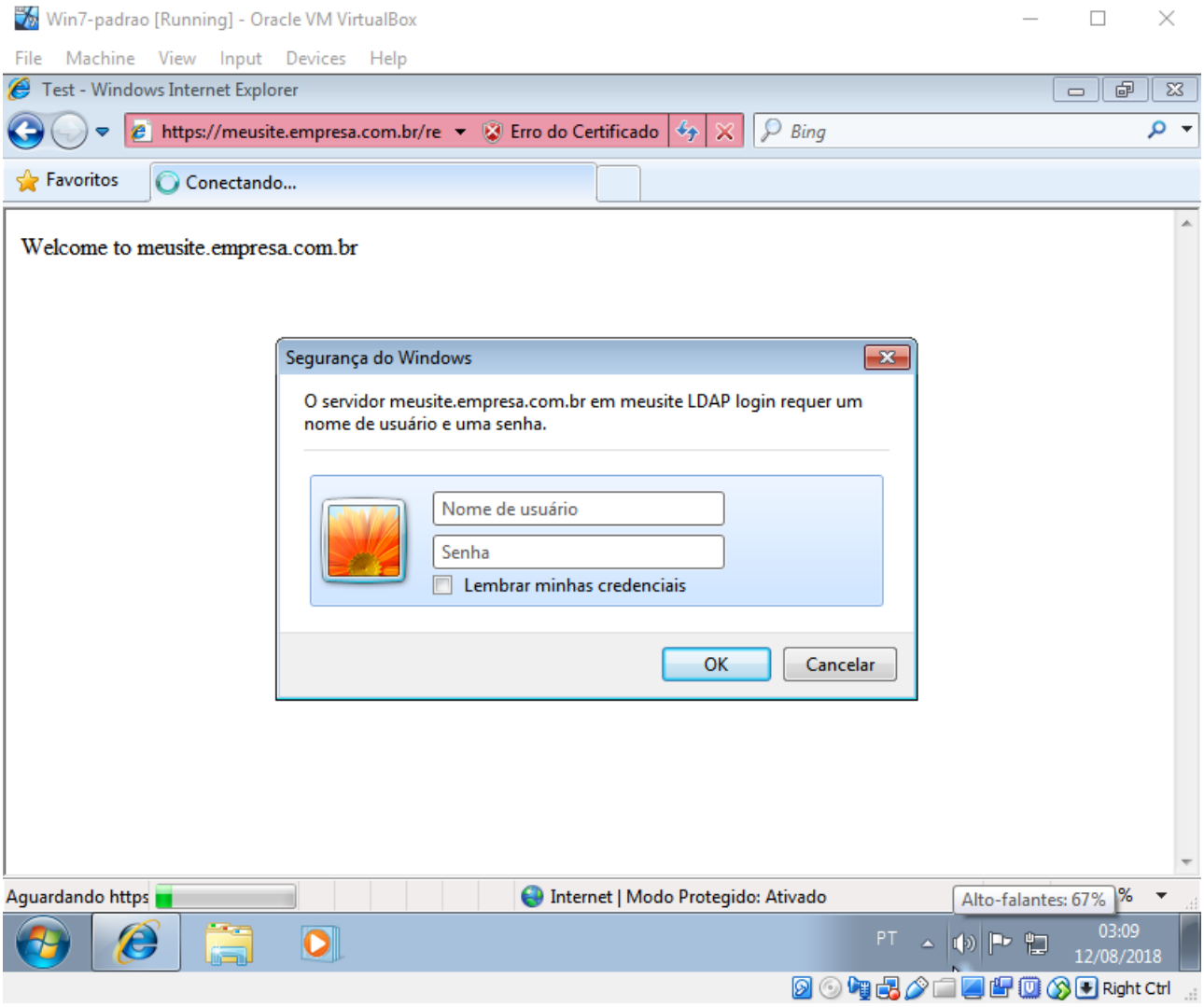

Figura 11: Autenticação LDAP no Apache

6. Ao informar uma combinação válida (por exemplo, usuário `aluno` e senha `rnpesr`), o Apache autoriza o acesso à área restrita do *virtualhost*:

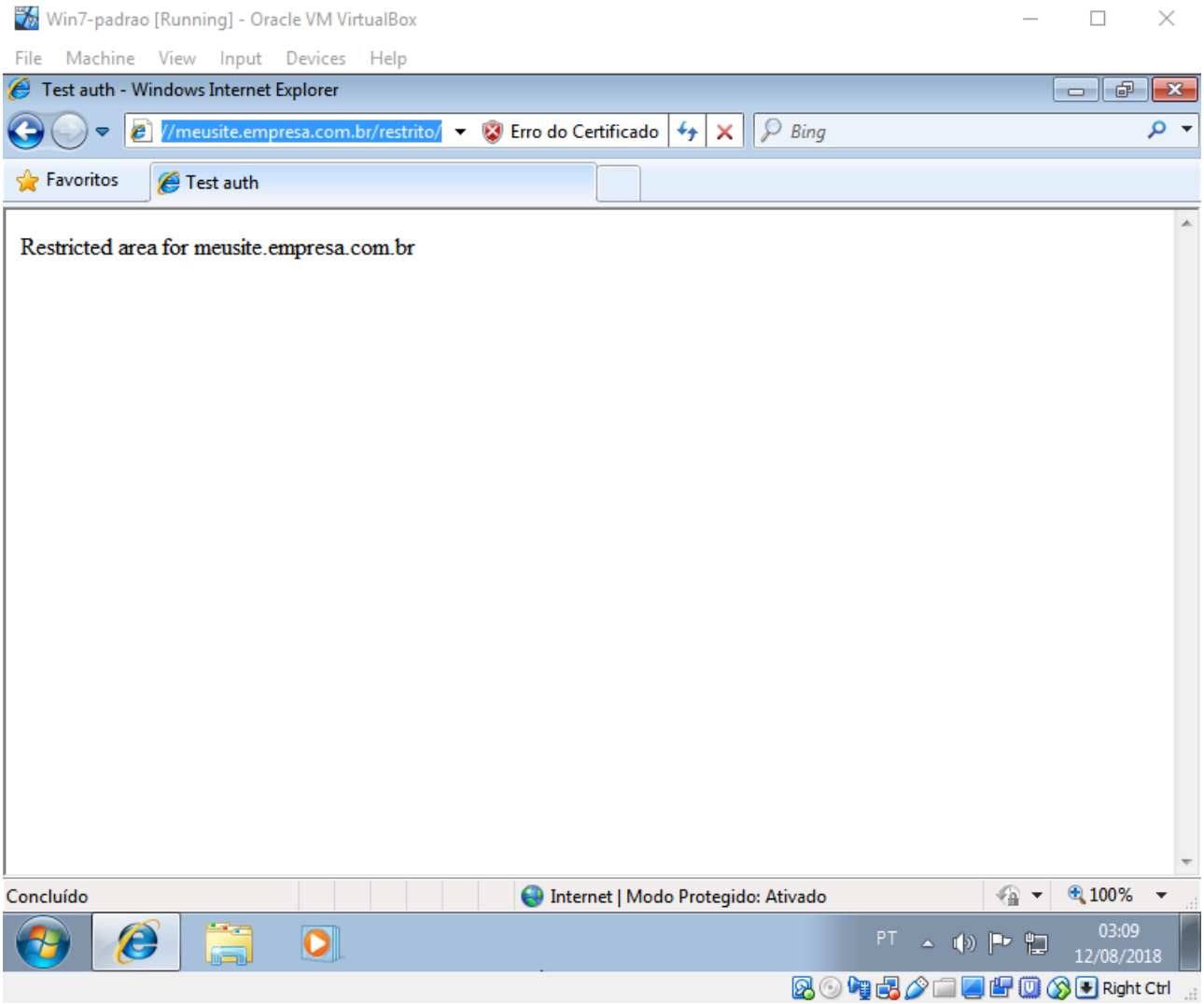

Figura 12: Usuário autenticado no LDAP/Apache com sucesso

7. Refazendo o acesso, mas desta vez informando uma combinação de usuário/senha inexistente, o servidor web informa que não estamos autorizados a acessar a área restrita:

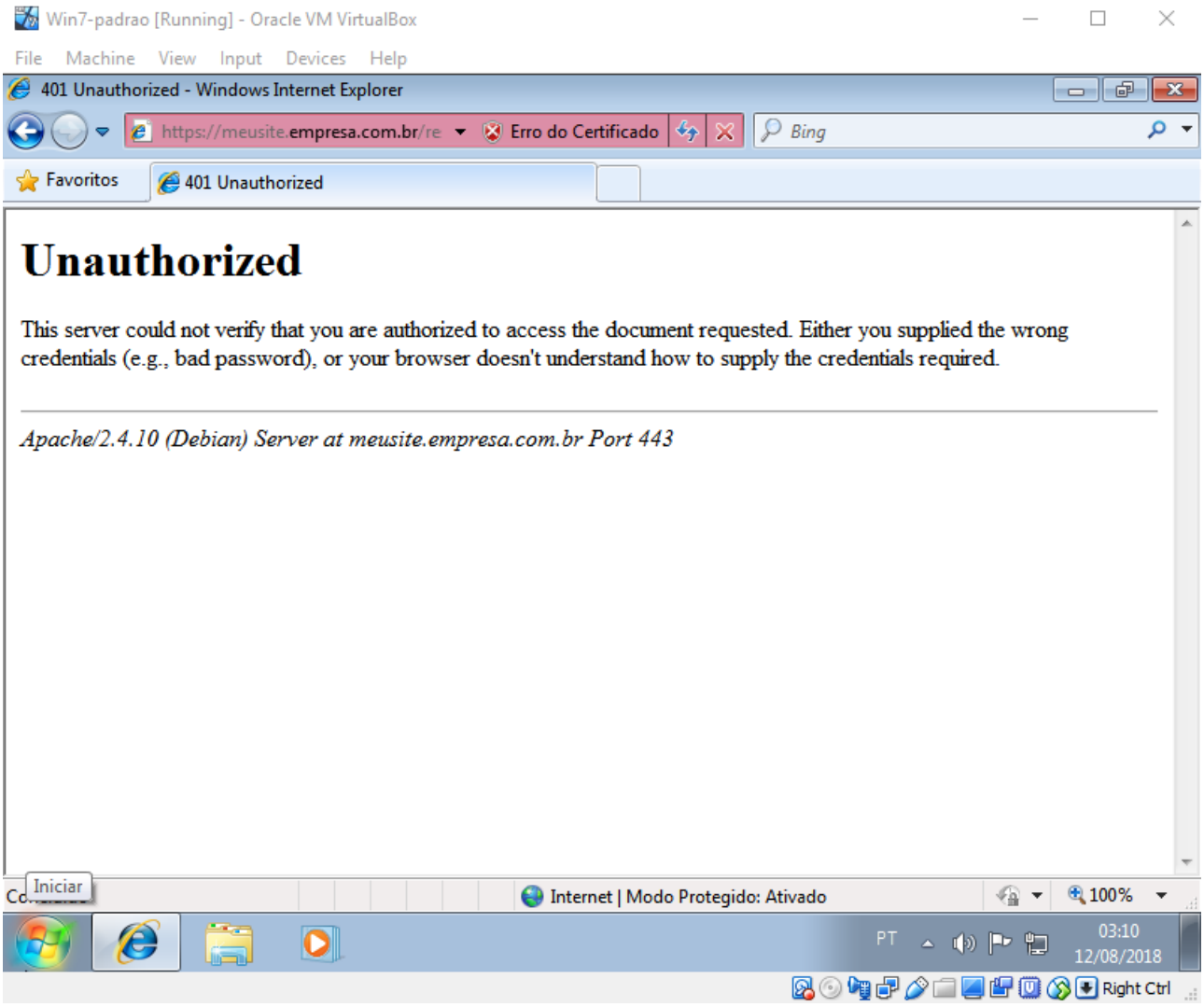

Figura 13: Usuário não autorizado pelo LDAP/Apache

5) Habilitando páginas pessoais de usuários

O módulo `userdir` do Apache permite a um usuário publicar seu próprio *site*, localizado dentro da sua pasta pessoal. Ele procura uma pasta com nome `public_html` dentro do diretório *home* do usuário e, caso existente, serve o conteúdo dessa pasta via HTTP.

1. Habilite o módulo páginas pessoais do Apache, `userdir`, através do comando `a2enmod`.
2. Crie a pasta `public_html` dentro do diretório *home* do usuário `aluno` e insira dentro dela um arquivo `index.html` que permita testar a configuração.
3. Configure o sistema para que todos os usuários criados futuramente já tenham a pasta `public_html` criada automaticamente eu seus diretórios *home*.
4. Teste o acesso à página pessoal do usuário `aluno` a partir de um navegador (seja na máquina física, *Client_Linux* ou *Win7-padrao*), verificando que suas configurações surtiram efeito.

Siga os passos abaixo:

1. Habilite o módulo de publicação de páginas pessoais no Apache:

```
# a2enmod userdir
Enabling module userdir.
To activate the new configuration, you need to run:
  service apache2 restart
```

2. Crie a pasta `/home/aluno/public_html`, e crie nela um arquivo `index.html` com conteúdo sugestivo:

```
$ mkdir ~/public_html

$ cat public_html/index.html
<html>
  <head>
    <title>Test userdir</title>
</head>
  <body>
    Welcome to the ALUNO homepage
  </body>
</html>
```

3. Para que usuários criados no futuro possuam a pasta `public_html` criada automaticamente eu seus diretórios *home*, basta criar uma pasta de mesmo nome no diretório `/etc/skel`:

```
# mkdir /etc/skel/public_html
```

4. Reinicie o Apache para que ele ative o módulo de publicação de páginas pessoais:

```
# systemctl restart apache2
```

5. Agora, basta testar. Acessamos a URL http://192.168.0.10/~aluno e logo podemos ver a página pessoal do usuário `aluno`, como esperado:

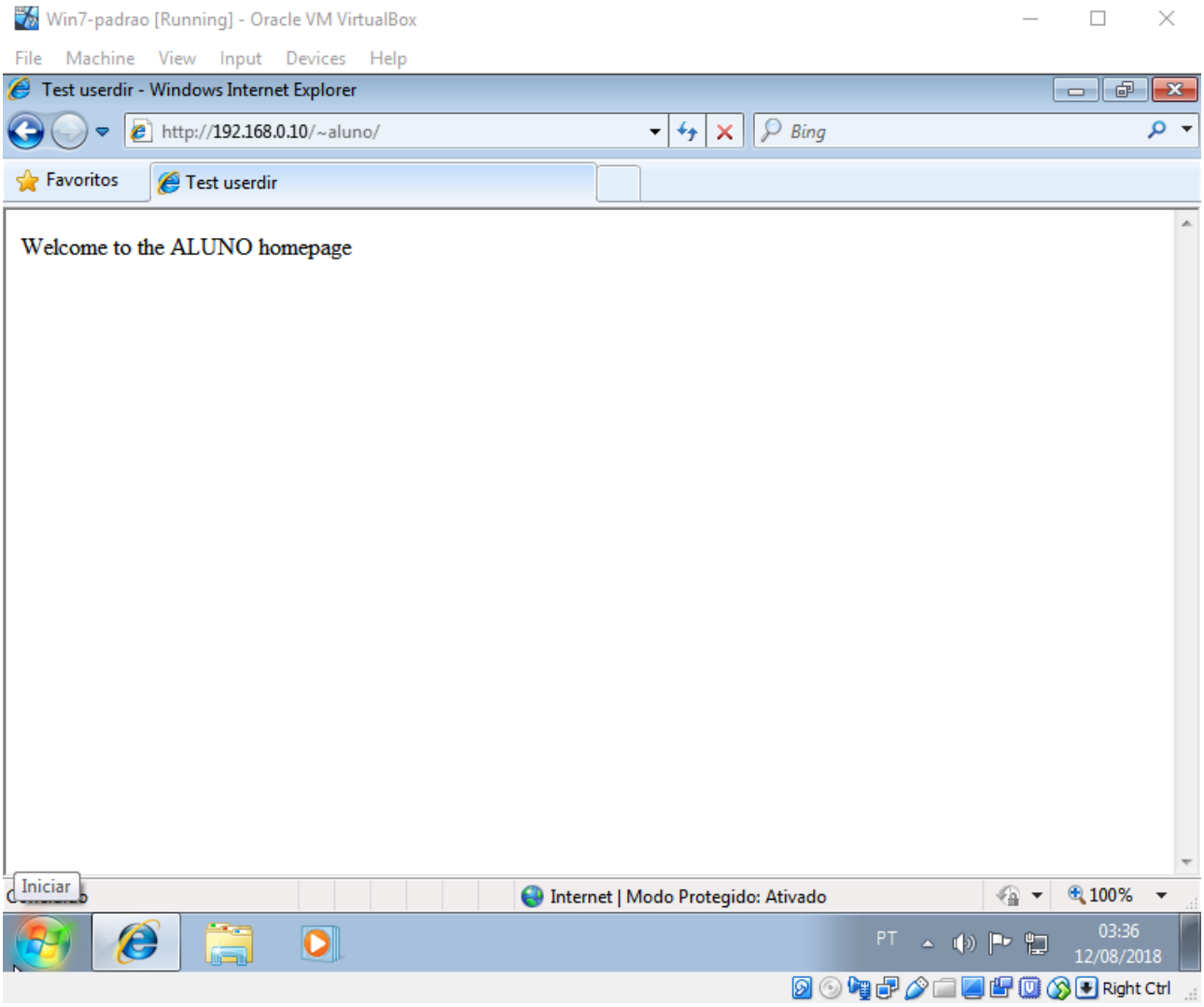

Figura 14: Acesso à página pessoal do usuário aluno