LABORATÓRIO 08 – Certificados Digitais e ICP-BRASIL

Este laboratório tem por objetivo exercitar o uso das facilidades do componente de segurança para autenticação com Certificados Digitais e o tratamento das Extensões da ICP-BRASIL.
É necessário o conhecimento/conclusão do Laboratório do módulo 6.

Objetivos:

- Configuração do Servidor JBoss e Navegador Internet.
- Alteração do *security-provider* e uso das extensões da ICP-BRASIL
- Login/Autenticação com Certificado Digital.

Exercício 8.1 – Configuração do JBOSS e Navegador Internet.

1. **Arquivos necessários:**
   - http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/localhost_truststore.jks?view=log
   - http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/keystore_localhost.jks?view=log
   - http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/AC_DEMOISELLE_TESTES.cer?view=log
   - http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/teste_pj.p12?view=log
   - TODOS OS CERTIFICADOS DISPONIBILIZADOS SÃO APENAS PARA TESTES, SEM QUALQUER VALIDADE OU USO.

2. **Configuração inicial do JBoss:**
   - Edite o arquivo *server.xml* localizado em <jboss_home>/server/default/deploy/jboss-web.deployer/ e inclua a tag abaixo:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
               address="${jboss.bind.address}"
                maxThreads="150" scheme="https" secure="true"
                clientAuth="true" sslProtocol="TLS"
               keystoreFile="${jboss.server.home.dir}/conf/keystore_localhost.jks"
                keystorePass="teste1"
               truststoreFile="${jboss.server.home.dir}/conf/localhost_truststore.jks"
               truststorePass="teste1"
                />
<?xml version="1.0" encoding="UTF-8"?>
```

3. Configuração do Navegador Internet (Firefox):

- Primeiramente abra o navegador Internet, nosso exemplo o Firefox: , acione o menu: Editar → Preferencias..
  - Selecione o Ícone Avançado e aba Criptografia conforme a figura abaixo*:*

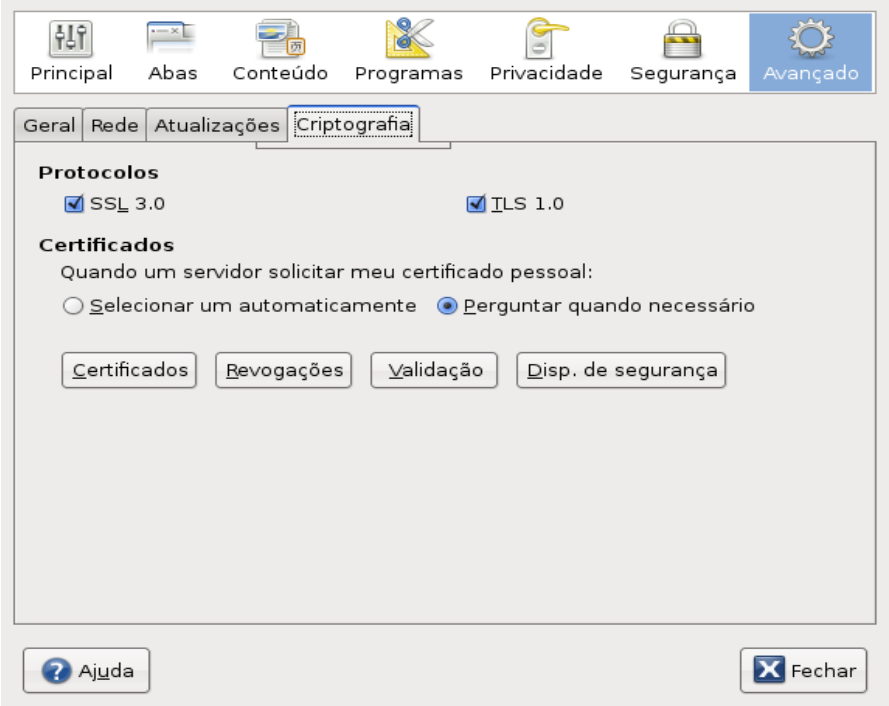

Figura 1. Configurando Certificados

  - Em seguida clique no Botão “Certificados” e selecione a Aba **Autoridades**.

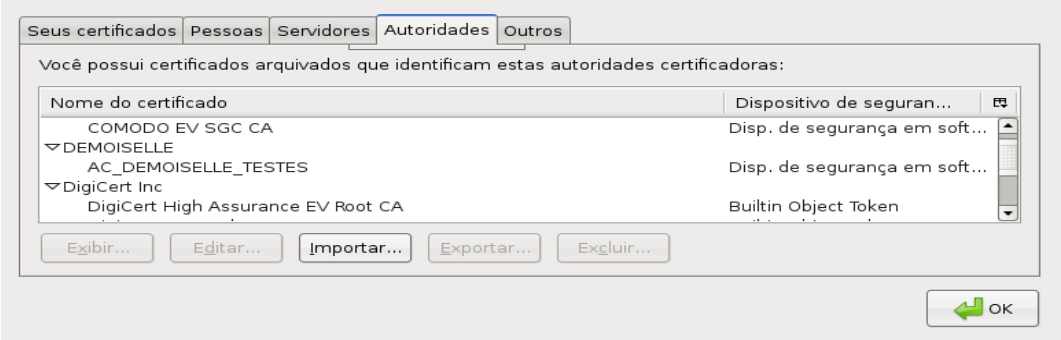

Figura 2. Importando AC.

Clique no botão importar e selecione o arquivo previamente localizado:

(http://demoiselle-comp.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/AC_DEMOISELLE_TESTES.cer?view=log).

Na tela seguinte, marque todas as opções:

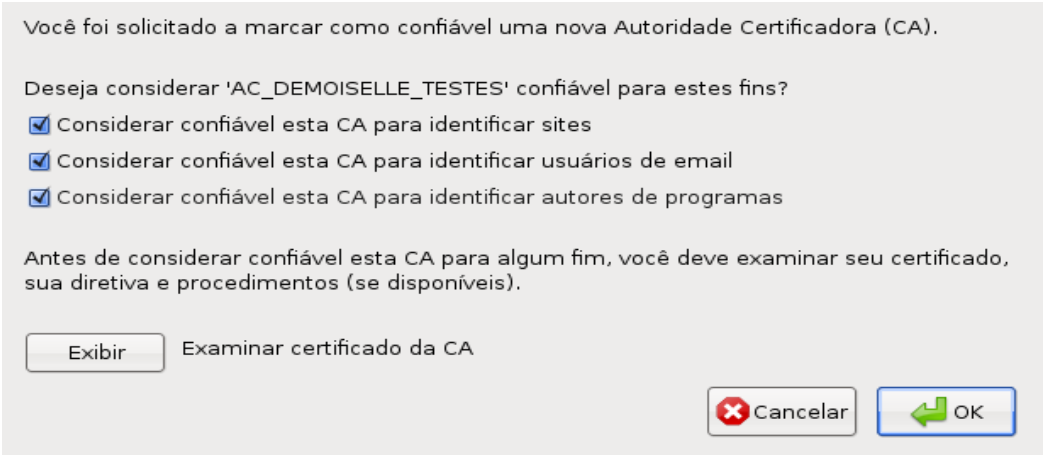

Figura 3. Confiando no Certificado.

Reinicie (abrir e fechar) o Navegador.

Refaça os passos do item para chegar novamente à condição da Figura 2. Mas desta vez selecione a aba **Seus Certificados**. Clique no Botão importar, e selecione o arquivo previamente localizado: (http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/teste_pj.p12?view=log)

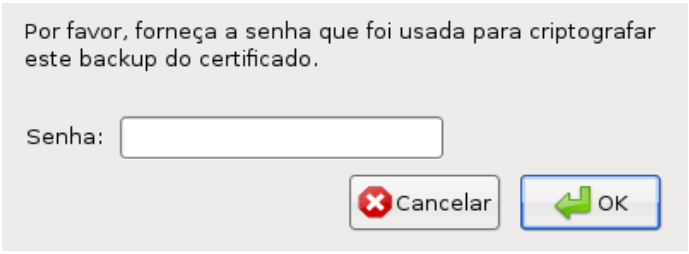

Figura 4. Senha do Certificado.

Quando solicitado informe a senha : “teste1” para importar o certificado.
Os resultados serão os seguintes:

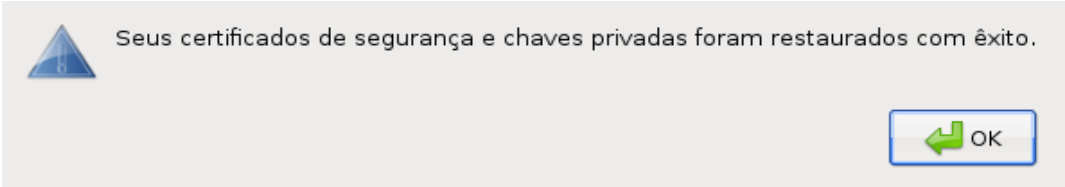

Figura 5. Certificado Aceito.

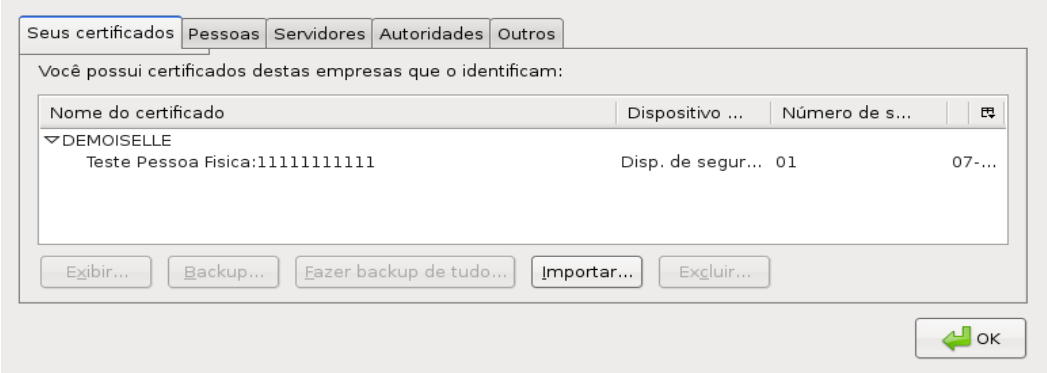

Figura 6. Certificado Importado Corretamente.

4. Testando a Configuração:

Inicie (start) o servidor de aplicações JBoss.
Em seguida, utilizando o Navegador configura, acesse o endereço:
https://localhost:8443/
Será apresentada a tela abaixo:

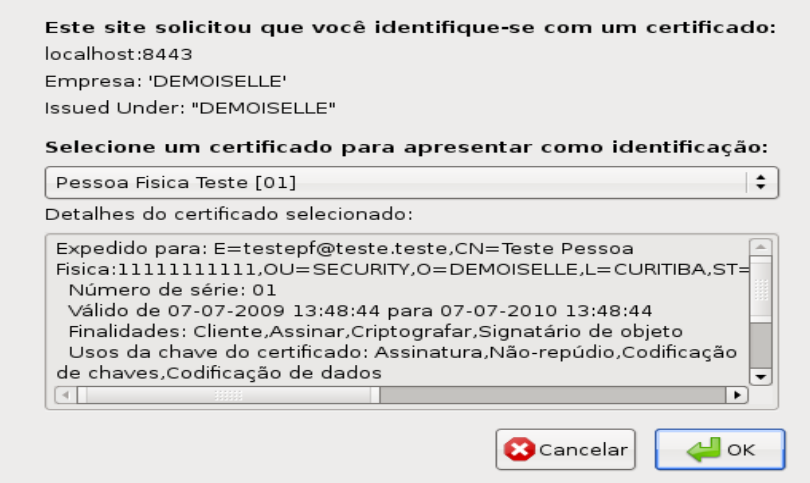

Figura 7. Pedido de identificação.

Clique no botão OK, e em seguida será apresentada a tela inicial do JBoss, a única diferença entre o processo normal (default) é que a conexão estará certificada (cadeado no rodapé da página).
Se não ocorrer nenhum erro, a configuração está correta.

Exercício 8.2 – Alteração do *Security-Provider* e uso das extensões da ICP-BRASIL.

Tendo sido executado com êxito o laboratório do Módulo 6, dos Tutoriais do DEMOISELLE, volte novamente a este projeto.

- No arquivo POM.XML, inclua as seguintes dependências:

```
<dependency>
        <groupId>bouncycastle</groupId>
        <artifactId>bcprov-jdk15</artifactId>
        <version>140</version>
        <scope>provided</scope>
</dependency>
```

```
<dependency>
	<groupId>junit</groupId>
	<artifactId>junit</artifactId>
	<version>4.5</version>
	<scope>test</scope>
</dependency>
<dependency>
	<groupId>log4j</groupId>
	<artifactId>log4j</artifactId>
	<version>1.2.14</version>
	<scope>provided</scope>
</dependency>
```

- Confira a versão do componente de segurança:

```
<dependency>
	<groupId>br.gov.component.demoiselle</groupId>
	<artifactId>demoiselle-security</artifactId>
	<version>1.0.5</version>
	<scope>provided</scope>
</dependency>
```

- Altere a classe `public class EscolaPrincipal implements Principal,` incluindo o atributo

```
private boolean usuarioCertificado=false;
```

e os seguinte métodos:

```
public EscolaPrincipal(String name, String cpf, String nomeCompleto, boolean usuarioCertificado) {
	super();
	this.name = name;
	this.cpf = cpf;
	this.nomeCompleto = nomeCompleto;
	this.usuarioCertificado = usuarioCertificado;
}

public boolean isUsuarioCertificado() {
	return this.usuarioCertificado;
}
public void setUsuarioCertificado(boolean usuarioCertificado) {
	this.usuarioCertificado = usuarioCertificado;
}
```

- Crie uma nova classe que irá representar os certificados no padrão ICP-BRASIL:

```
package br.gov.serpro.bcmdweb.security.provider.security.auth.provider;

import java.util.Date;
import java.util.List;
import br.gov.component.demoiselle.security.certificate.extension.DefaultExtension;
import br.gov.component.demoiselle.security.certificate.extension.DefaultExtensionType;
import br.gov.component.demoiselle.security.certificate.extension.ICPBrasilExtension;
import br.gov.component.demoiselle.security.certificate.extension.ICPBrasilExtensionType;

public class Certificado {

	@ICPBrasilExtension(type=ICPBrasilExtensionType.CPF)
	private String cpf;
	@ICPBrasilExtension(type=ICPBrasilExtensionType.NOME)
	private String nome;
	@ICPBrasilExtension(type=ICPBrasilExtensionType.CEI_PESSOA_FISICA)
	private String ceiPessoaFisica;
	@ICPBrasilExtension(type=ICPBrasilExtensionType.NIS)
	private String pisPasep;
	@ICPBrasilExtension(type=ICPBrasilExtensionType.EMAIL)
	private String email;
	@ICPBrasilExtension(type=ICPBrasilExtensionType.DATA_NASCIMENTO)
	private String dataNascimento;
	@ICPBrasilExtension(type=ICPBrasilExtensionType.NUMERO_IDENTIDADE)
```

```
private String numeroIdentidade;
@ICPBrasilExtension(type=ICPBrasilExtensionType.ORGAO_EXPEDIDOR_IDENTIDADE)
private String orgaoExpedidorIdentidade;
@ICPBrasilExtension(type=ICPBrasilExtensionType.UF_ORGAO_EXPEDIDOR_IDENTIDADE)
private String UfExpedidorIdentidade;
@ICPBrasilExtension(type=ICPBrasilExtensionType.NUMERO_TITULO_ELEITOR)
private String numeroTituloEleitor;
@ICPBrasilExtension(type=ICPBrasilExtensionType.ZONA_TITULO_ELEITOR)
private String zonaTituloEleitor;
@ICPBrasilExtension(type=ICPBrasilExtensionType.SECAO_TITULO_ELEITOR)
private String secaoTituloEleitor;
@ICPBrasilExtension(type=ICPBrasilExtensionType.MUNICIPIO_TITULO_ELEITOR)
private String municipioTituloEleitor;
@ICPBrasilExtension(type=ICPBrasilExtensionType.UF_TITULO_ELEITOR)
private String ufTituloEleitor;
@ICPBrasilExtension(type=ICPBrasilExtensionType.CNPJ)
private String cnpj;
@ICPBrasilExtension(type=ICPBrasilExtensionType.CEI_PESSOA_JURIDICA)
private String ceiPessoaJuridica;
@ICPBrasilExtension(type=ICPBrasilExtensionType.NOME_RESPONSAVEL_PESSOA_JURIDICA)
private String nomeResponsavelPessoaJuridica;
@ICPBrasilExtension(type=ICPBrasilExtensionType.NOME_EMPRESARIAL)
private String nomeEmpresarial;
@ICPBrasilExtension(type=ICPBrasilExtensionType.NIVEL_CERTIFICADO)
private String nivelCertificado;
@ICPBrasilExtension(type=ICPBrasilExtensionType.TIPO_CERTIFICADO)
private String tipoCertificado;
@DefaultExtension(type=DefaultExtensionType.CRL_URL)
private List<String> crlURL;
@DefaultExtension(type=DefaultExtensionType.AFTER_DATE)
private Date afterDate;
@DefaultExtension(type=DefaultExtensionType.BEFORE_DATE)
private Date beforeDate;
@DefaultExtension(type=DefaultExtensionType.CERTIFICATION_AUTHORITY)
private Boolean certificationAuthority;
@DefaultExtension(type=DefaultExtensionType.SERIAL_NUMBER)
private String serialNumber;
@DefaultExtension(type=DefaultExtensionType.ISSUER_DN)
private String issuerDN;
@DefaultExtension(type=DefaultExtensionType.SUBJECT_DN)
private String subjectDN;
@DefaultExtension(type=DefaultExtensionType.KEY_USAGE)
private String keyUsage;
@DefaultExtension(type=DefaultExtensionType.PATH_LENGTH)
private int pathLength;
@DefaultExtension(type=DefaultExtensionType.AUTHORITY_KEY_IDENTIFIER)
private String authorityKeyIdentifier;
@DefaultExtension(type=DefaultExtensionType.SUBJECT_KEY_IDENTIFIER)
private String subjectKeyIdentifier;


public List<String> getCrlURL() {
        return crlURL;
}
public String getCpf() {
        return cpf;
}
public String getNome() {
        return nome;
}
public String getCeiPessoaFisica() {
        return ceiPessoaFisica;
}
public String getPisPasep() {
        return pisPasep;
}
public String getEmail() {
        return email;
}
public String getDataNascimento() {
```

```
		return dataNascimento;
	}
	public String getNumeroIdentidade() {
		return numeroIdentidade;
	}
	public String getOrgaoExpedidorIdentidade() {
		return orgaoExpedidorIdentidade;
	}
	public String getUfExpedidorIdentidade() {
		return UfExpedidorIdentidade;
	}
	public String getNumeroTituloEleitor() {
		return numeroTituloEleitor;
	}
	public String getZonaTituloEleitor() {
		return zonaTituloEleitor;
	}
	public String getSecaoTituloEleitor() {
		return secaoTituloEleitor;
	}
	public String getMunicipioTituloEleitor() {
		return municipioTituloEleitor;
	}
	public String getUfTituloEleitor() {
		return ufTituloEleitor;
	}
	public String getCnpj() {
		return cnpj;
	}
	public String getCeiPessoaJuridica() {
		return ceiPessoaJuridica;
	}
	public String getNomeResponsavelPessoaJuridica() {
		return nomeResponsavelPessoaJuridica;
	}
	public String getNomeEmpresarial() {
		return nomeEmpresarial;
	}
	public String getNivelCertificado() {
		return nivelCertificado;
	}
	public String getTipoCertificado() {
		return tipoCertificado;
	}
	public Date getAfterDate() {
		return afterDate;
	}
	public Date getBeforeDate() {
		return beforeDate;
	}
	public Boolean getCertificationAuthority() {
		return certificationAuthority;
	}
	public String getSerialNumber() {
		return serialNumber;
	}
	public String getIssuerDN() {
		return issuerDN;
	}
	public String getSubjectDN() {
		return subjectDN;
	}
	public String getKeyUsage() {
		return keyUsage;
	}
	public int getPathLength() {
		return pathLength;
	}
	public String getAuthorityKeyIdentifier() {
		return authorityKeyIdentifier;
```

```
	}
	public String getSubjectKeyIdentifier() {
		return subjectKeyIdentifier;
	}
}
```

- Altere a classe EscolaAutenticacao,

  Incluindo:

```
import java.security.cert.X509Certificate;
import org.apache.log4j.Logger;
import br.gov.component.demoiselle.security.certificate.CertificateManager;
```

  E o Atributo:

  private static Logger *log* = Logger.*getLogger*(EscolaAutenticacao.**class**.getName());

  E a implementação do método:

```
	public Principal authenticate(X509Certificate x509) {

		try {

			// Para este teste, não habilitaremos os validadores

			CertificateManager cm = new CertificateManager(x509, false);
			Certificado cert = cm.load(Certificado.class);

			if (cert.getTipoCertificado() =="PF"){
				log.info("Certificado de Pessoa Fisica!");
				return new EscolaPrincipal("PF", cert.getCpf(),cert.getNome(),true);
			}
			if (cert.getTipoCertificado() =="PJ"){
				log.info("Certificado de Pessoa Juridica!");
				return new EscolaPrincipal("PJ", cert.getCnpj(),cert.getNome(),true);
			}
		}catch (Exception e) {
			e.printStackTrace();
			return null;
		}
		log.info("Outro tipo de certificado não aceito!");
		return null;
	}
```

```
	public Principal authenticate(String user, String password) {
		if (user.equals("aluno") && password.equals("aluno")) {
			return new EscolaPrincipal("aluno", "001", "Aluno da Escola", false);
		}
		if (user.equals("professor") && password.equals("professor")) {
			return new EscolaPrincipal("professor", "002",
					"Professor da Escola", false);
		}
		if (user.equals("admin") && password.equals("admin")) {
			return new EscolaPrincipal("admin", "003",
					"Administrador da Escola", false);
		}
		return null;
	}
```

- Altere a classe EscolaAutorizacao:

```
package br.gov.demoiselle.escola.security.provider.security.auth.provider;

import java.security.Principal;
```

```
import java.util.ArrayList;
import java.util.Collection;
import java.util.Properties;

import br.gov.component.demoiselle.security.auth.Role;
import br.gov.component.demoiselle.security.auth.provider.IAuthorizationProvider;

public class EscolaAutorizacao implements IAuthorizationProvider{

	public Collection<Role> authorize(Principal callerPrincipal) {
		EscolaPrincipal principal = (EscolaPrincipal) callerPrincipal;
		Collection<Role> papeis = new ArrayList<Role>();

		if (principal.getName().equals("aluno")){
			papeis.add(new Role("role_aluno"));

		}else if (principal.getName().equals("professor")){
			papeis.add(new Role("role_professor"));

		}else if (principal.getName().equals("admin")){
			papeis.add(new Role("role_administrador"));
		}
		  else if(principal.isUsuarioCertificado()){
			//com certificado vai ser Administrador, neste exemplo
			papeis.add(new Role("role_administrador"));
		}

		}
		return papeis;
	}

	public void initialize(Properties properties) {

	}
}
```

- Testes
  - Baixe o arquivo http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/teste_pf.pem?view=log e coloque na pasta: /src/test/resources/.
  - Crie um caso de testes do JUnit para a classe EscolaAutenticacao (EscolaAutenticacaoTest), no pacote de Testes (/src/test/java). Conforme o exemplo abaixo

```
package br.gov.demoiselle.escola.security.provider.security.auth.provider;
import static org.junit.Assert.*;
import java.security.Principal;
import java.security.cert.CertificateException;
import java.security.cert.CertificateFactory;
import java.security.cert.X509Certificate;
import org.junit.Before;
import org.junit.Test;

public class EscolaAutenticacaoTest {

	private X509Certificate x509;
	@Before
	public void setUp() throws Exception {

	System.getProperties().put("demoiselle.component.security.certificate.crl.path",
"/tmp");
	System.getProperties().put("demoiselle.component.security.certificate.crl.index",
".index");

	}
	@Test
```

```
		public void testAuthenticateX509Certificate() {
			try {
				x509 = (X509Certificate)
CertificateFactory.getInstance("X.509").generateCertificate(Thread.currentThread().getContextClassLo
ader().getResourceAsStream("teste_pf.pem"));
			} catch (CertificateException e) {
				e.printStackTrace();
				assertFalse("Erro ao carregar o arquivo do certificado",false );
			}
			EscolaAutenticacao autenticacao = new EscolaAutenticacao();
			Principal resultado = autenticacao.authenticate(x509);
			assertTrue(resultado.getName().length() > 0);
		}

	}
```

- Rode o teste e verificará se a implementação está correta.

- Gere (conforme mostrado no módulo 6) novamente o projeto escola-security-provider utilizando o comando package do maven: Menu Eclipse → Run → Run As → Maven Package

- O pacote será gerado na pasta: /escola-security-provider/target/ com o nome: escola-security-provider-0.0.1-SNAPSHOT.jar

- **Configuração JAAS (Jboss)**

  - Verifique se os arquivos pedidos no módulo 06 (demoiselle-security-1.0.5.jar e demoiselle-security-catalina-1.0.5.jar), estejam no diretório <jboss_home>/server/default/lib do servidor.

  - Copie o jar escola-security-provider-0.0.1-SNAPSHOT.jar que foi gerado para a pasta {jboss_home}/server/default/lib. (se o arquivo já existir, sobrescreva)

  - Altere o arquivo $jboss_home/server/default/conf/login-config.xml, incluindo um novo login-module, e altere a flag para ***sufficient*** no anterior, conforme abaixo:

```
<application-policy name = "escola">
  <authentication>
   <login-module
        code="br.gov.component.demoiselle.security.auth.provider.ProviderLoginModule"
        flag="sufficient">

        <module-option name="credential-type">user_password</module-option>
        <module-option name="authentication-provider-class">
                br.gov.demoiselle.escola.security.provider.security.auth.provider.EscolaAutenticacao
        </module-option>
        <module-option name="authorization-provider-class">
                br.gov.demoiselle.escola.security.provider.security.auth.provider.EscolaAutorizacao
        </module-option>
    </login-module>

    <login-module
        code="br.gov.component.demoiselle.security.auth.provider.ProviderLoginModule"
        flag="sufficient">

        <module-option name="credential-type">certificate</module-option>
        <module-option name="authentication-provider-class">
                br.gov.demoiselle.escola.security.provider.security.auth.provider.EscolaAutenticacao
        </module-option>
        <module-option name="authorization-provider-class">
                br.gov.demoiselle.escola.security.provider.security.auth.provider.EscolaAutorizacao
        </module-option>
        <module-option name="certificate-callback-adapter-class">
                br.gov.component.demoiselle.security.auth.adapter.JBossCertificateCallbackAdapter
```

```
        </module-option>
    </login-module>

  </authentication>
</application-policy>
```

- Copie o jar bcprov-jdk15-140.jar (http://mirrors.ibiblio.org/pub/mirrors/maven2/bouncycastle/bcprov-jdk15/140/bcprov-jdk15-140.jar) para a pasta {jboss_home}/server/default/lib. Está disponível também no site oficial do projeto:http://www.bouncycastle.org/latest_releases.html
  Você pode utilizar versões mais recentes, recomendamos apenas que seja para JDK 1.5.

Com estas alterações, o *provider* de Segurança estará pronta paro aceitar tanto a autenticação por chave e senha como com o uso do certificado digital que foi incluído no navegador.

Exercício 8.3 – Login/Autenticação com Certificado Digital.

Com as alterações anteriores, até o momento, apenas o *provider* está preparado para aceitar os certificados digitais, a aplicação Escola estará funcionando da mesma forma (se reiniciar a aplicação verificará que não há nenhuma diferença aparente).

Caso queira testar o uso do certificado, bastaria informar o protoloco de segurança e a porta corretamente: *https://localhost:8443/escola/*

Também existem algumas formas para habilitar o uso da certificação digital na aplicação e até impedir que o acesso seja feito de outra forma, mas como nosso objetivo é apenas testar as funcionalidades, não faremos grandes mudanças no projeto e faremos apenas uma chamada ao contexto seguro.

Vamos fazer uma pequena alteração no projeto, para melhorar nossa usabilidade.

Para isto abra novamente o projeto Escola e faça a seguinte alteração no arquivo login.jsp, localizado em src/main/webapp/public/pages/security/login.jsp:

```
  ... inicio do código
 <td colspan="2" align="right">
          <input type="submit" value="login" tabindex="3" />
 </td>
</tr>

<tr>
    <td><a href="https://localhost:8443/escola/">Link para Certificado Digital</a>
    </td>
</tr>
```

Publique novamente a aplicação e reinicie o servidor JBoss.
Abra a aplicação e verá que a página de login estará da seguinte forma:

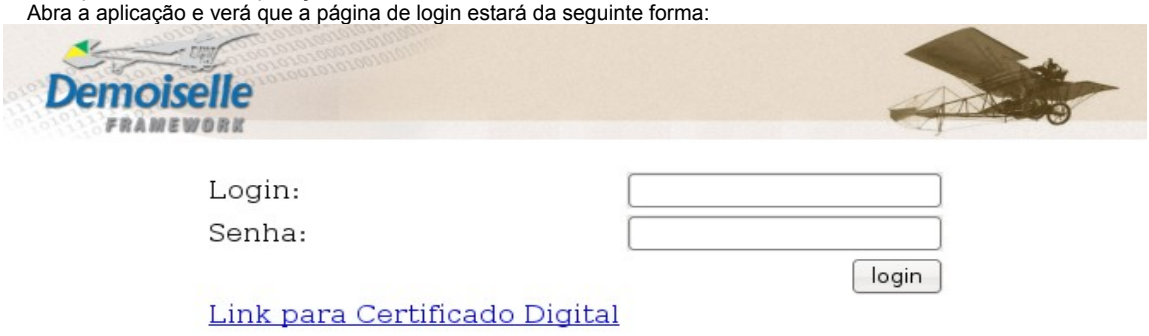

Figura 8 – Nova tela de Login

Ao clicar no “link para Certificado Digital”, o navegador irá pedir o certificado.
Caso possua um certificado ICP-BRASIL válido, é possível utilizá-lo neste exemplo, pois o truststore

do JBoss que disponibilizamos, já contém a cadeia ICP-BRASIL.

Pelo Eclipse você poderá acompanhar no Console as logs geradas pelo provider, ex:

```
- 14:17:02,587 INFO  [EscolaAutenticacao] Certificado de Pessoa Fisica!
```

- Sugestões de exercício:

  No endereço: http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/ existem alguns outros tipos de certificados de testes (Pessoa Jurídica, Equipamento, Níveis A3)

Alterando a classe EscolaAutenticacao do *provider* de segurança, é possível fazer outras verificações, como por exemplo:

Bloquear o acesso de certificados de níveis menos seguros (A1):

```
if (cert.getNivelCertificado() == "A1"){
        log.info("Não permitido Certificado A1!");
        return null;
}
```

Bloquear o uso de certificados de equipamento:

```
if (cert.getTipoCertificado() =="EA"){
  log.info("Não é permitido para Equipamento!");
  return null;
}
```

Habilite os validadores:

```
CertificateManager cm = new CertificateManager(x509, true);
```

Utilize o arquivo: http://demoiselle.svn.sourceforge.net/viewvc/demoiselle/component/trunk/demoiselle-security/src/test/resources/b9e29b971afedc2108ec7f90d7a0e0d2 para validar a CRL (incluindo o arquivo no diretório definido em: demoiselle.component.security.certificate.crl.path)