Servizi di aggiornamento delle chiavi pubbliche relative agli strumenti di revisione

Autore: C.S.R.P.A.D.

Descrizione: Manuale web-service

Documento: 2010-02-25_MU-001_IT

Rev 001 del 25/02/2010

Rev 002 del 30/06/2010

Rev 003 del 18/06/2015

Sommario

Scopo e campo di applicazione

Il Portale del C.S.R.P.A.D. fornisce il servizio per lo scaricamento delle chiavi pubbliche relative agli strumenti utilizzati nel processo di revisione dei veicoli. Il servizio è accessibile esclusivamente a tutti i costruttori che si sono precedentemente registrati nel portale attraverso l'apposita procedura.

Descrizione del servizio

Il servizio è accessibile tramite Web Services (in formato SOAP).

Esso consente di reperire:

- l'elenco di tutte le chiavi valide (relative a tutti gli strumenti)
- l'elenco di tutte le chiavi valide (relative a tutti gli strumenti) con data di ultima modifica compresa in un intervallo di tempo
- l'elenco di tutte le chiavi valide (relative a tutti gli strumenti) con data di ultima modifica precedente ad una data
- l'elenco delle chiavi valide relative ad una omologazione
- l'elenco delle chiavi valide relative ad una omologazione con data di ultima modifica compresa in un intervallo di tempo
- l'elenco delle chiavi valide relative ad una omologazione con data di ultima modifica precedente ad una data
- l'elenco delle chiavi valide relative ad un determinato tipo di crittografia
- l'elenco delle chiavi valide relative ad un determinato tipo di crittografia con data di ultima modifica compresa in un intervallo di tempo
- l'elenco delle chiavi valide relative ad un determinato tipo di crittografia con data di ultima modifica precedente ad una data
- l'elenco di tutte le chiavi (comprese anche le chiavi non attive o scadute)
- l'elenco di tutte le chiavi (comprese anche le chiavi non attive o scadute) con data di ultima modifica compresa in un intervallo di tempo

- l'elenco di tutte le chiavi (comprese anche le chiavi non attive o scadute) con data di ultima modifica precedente ad una data
- l'elenco delle chiavi non attive
- l'elenco delle chiavi non attive relative ad un determinato tipo di crittografia
- una singola chiave

Il servizio fornisce anche funzioni aggiuntive per lo scaricamento parziale delle chiavi, per consentire ad esempio di impostarne la paginazione.

Tipo di crittografia delle chiavi

Ogni omologazione può contenere chiavi con crittografia di tipo RSA 1024 oppure RC4.

Per una stessa omologazione possono essere presenti entrambi i formati, oppure solamente uno di essi.

Scadenza delle chiavi

Quando un costruttore inserisce una nuova chiave per una omologazione, la precedente chiave inserita per la stessa omologazione, con lo stesso tipo di crittografia, inizia un periodo di validità che ha la durata di trenta giorni.

Per questo motivo quando si richiedono le chiavi valide per uno specifico strumento, queste possono essere più di una. Infatti il servizio fornisce l'ultima chiave inserita dal costruttore, oltre a tutte le chiavi non ancora scadute. Sono escluse le chiavi con omologazione non attiva.

Sono disponibili altri metodi per ottenere lo storico delle chiavi relative ad una specifica attrezzatura, anche nel caso in cui la relativa omologazione sia stata disattivata. Questi metodi potranno essere utilizzati, al fine di verificare la validità di una prova effettuata in passato.

Descrizione delle chiavi

I dati relativi ad una singola chiave sono incapsulati nell'oggetto Key che viene restituito singolarmente o contenuto in una lista (array) restituita dai metodi preposti allo scaricamento delle chiavi stesse.

L'oggetto Key espone le seguenti proprietà:

1. *key*: la chiave (in formato testo)
2. *code*: codice (in formato numerico a 5 cifre)
3. *encryptionType*: tipo di crittografia
4. *linkType*: tipo di collegamento
5. *approvalNumber*: numero di omologazione dello strumento
6. *creationDate*: data di creazione
7. *expirationDate*: data di scadenza
8. *inhibitionDate*: data della disattivazione
9. *updateDate*: data dell'ultima modifica

Dettaglio dei campi

Testo della chiave (key)

Contiene la chiave.

Il formato dipende dal tipo di crittografia:

- le chiavi RSA 1024 sono in formato xml
- le chiavi RC4 sono in formato testo esadecimale di 8 caratteri.

Esempio di xml della chiave RSA 1024:

```
<RSAKeyValue>
    <Modulus>
```

yPIZrZHeMPqVoRnclEwSUdRFkO2k4eEvjBfi6y9aVVpfzgJ34fRL6txizafmrKu8L
4rAGklNX95ueMTGmgR84uo1F4sWHtgL4SIb2EE/SJWKMQeSKcEbHfCu+a17Xma40e
Td23Ld/LoV5uhs5FSRlVBDlld1APZllcDdAeUKu/k=
</Modulus>

<Exponent>AQAB</Exponent>

</RSAKeyValue>

Codice (code)
Individua una stringa numerica di cinque cifre che rappresenta il progressivo associato dal sistema alla chiave.

Tipo di crittografia (encryptionType)
Individua un testo che identifica il tipo di crittografia utilizzata per la chiave. I valori possibili sono RSA oppure RC4.

Tipo di collegamento(linkType)
Individua un testo che identifica il tipo di collegamento a cui è associata la chiave. I valori possibili sono DIR, RETE, RSSE, RSCE.

Numero di omologazione dello strumento (approvalNumber)
Individua un testo (massimo 50 caratteri) che identifica l'omologazione a cui sono associate le chiavi.

Data di creazione (creationDate)
Individua la data di creazione della chiave.

Data di scadenza (expirationDate)
Individua la data di scadenza della chiave. Questa coincide con la data di inserimento (posticipata di 30 giorni) di una nuova chiave relativa allo stesso tipo di crittografia.

Data di disattivazione (inhibitionDate)
Individua l'eventuale data nella quale è stata disattivata l'omologazione. Una chiave disattivata è da considerarsi a tutti gli effetti come non più valida a partire dalla data di disattivazione.

Data dell'ultima modifica (updateDate)
Individua la data dell'ultima modifica dello stato della chiave. Ad esempio una chiave cambia di stato quando per la relativa omologazione viene associata una nuova chiave, o anche qualora la chiave stessa venga disattivata.

Specifiche del servizio di autenticazione

Note introduttive

Per poter accedere ai servizi per lo scaricamento delle chiavi pubbliche è necessario prima autenticarsi nel sistema. Allo scopo occorre invocare gli appositi metodi del servizio di autenticazione fornendo lo username e la password relativi al proprio utente costruttore.

Accesso al servizio

Il servizio di autenticazione è pubblicato all'indirizzo:

https://www.csrpad.it/csrpad-csrpad/AuthenticatorService?wsdl

Metodi implementati

boolean loginPlain(String username, String password)

Permette di creare una sessione di lavoro autenticata, tramite username e password. Restituisce *true* in caso di successo (autenticazione avvenuta).

public String login(String username, String password)

Permette di creare una sessione di lavoro autenticata, tramite username e password. Attiva in più la gestione della conversazione (descritta in seguito), restituendone il *conversationId* in caso di successo (autenticazione avvenuta), altrimenti una stringa vuota.

boolean logout()

Permette di chiudere una sessione di lavoro autenticata, terminando l'eventuale conversazione aperta. Restituisce un boolean di avvenuta chiusura.

Entrambi i metodi di *login* impostano un identificativo di sessione (*JSESSIONID*) nell'header della risposta SOAP. Per utilizzare la sessione autenticata nelle successive chiamate ai servizi, l'identificativo di sessione deve essere impostato negli header delle richieste SOAP.

Esempio di risposta generata dal metodo ***login:***

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
X-Powered-By: Servlet 2.5; JBoss-5.0/JBossWeb-2.1
Set-Cookie: JSESSIONID=FF013E1775EC01CD786E0D4AC3C4C35A; Path=/csrpad-csrpad
Content-Type: text/xml;charset=UTF-8
Transfer-Encoding: chunked
Date: Wed, 17 Feb 2010 14:40:54 GMT

<env:Envelope xmlns:env='http://schemas.xmlsoap.org/soap/envelope/'>
  <env:Header>
    <seam:conversationId xmlns:seam='http://www.jboss.org/seam/webservice'>
    4
    </seam:conversationId>
  </env:Header>
  <env:Body>
    <ws:loginResponse xmlns:ws='http://www.csrpad.it/ws'>
      <return>4</return>
    </ws:loginResponse>
  </env:Body>
</env:Envelope>
```

Conversazione

La conversazione è un contesto, definito all'interno di una sessione di lavoro, che consente di mantenere memorizzato tra chiamate differenti lo stato che assumono gli oggetti utilizzati.

Si può aprire e chiudere la conversazione, senza che questo comporti la chiusura della sessione autenticata.

Per iniziare una nuova conversazione, occorre invocare il metodo *login*. Successivamente, per utilizzarla e mantenerla attiva, occorre impostare il *conversationId* nell'header delle richieste SOAP.

E' possibile ottenere il *conversationId* come valore restituito dal metodo *login*, oppure estrarlo dalla risposta SOAP dello stesso.

Esempio di utilizzo del conversationId:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:seam="http://seambay.example.seam.jboss.org/">
  <soapenv:Header>
    <seam:conversationId xmlns:seam='http://www.jboss.org/seam/webservice'>
      2
    </seam:conversationId>
  </soapenv:Header>
  <soapenv:Body>
    <seam:findValidKeys/>
  </soapenv:Body>
</soapenv:Envelope>
```

Specifiche del servizio di scaricamento delle chiavi

Note introduttive

Il termine ***ValidKeys*** nei metodi indica che le chiavi restituite sono attive e non scadute.

Il termine ***Updated*** nei metodi indica che le chiavi restituite sono tutte quelle modificate dalla data **startUpdateDate** alla data **endUpdateDate.**

Il termine ***UpdatedUntil*** nei metodi indica che le chiavi restituite sono tutte quelle modificate precedentemente rispetto alla data **endUpdateDate.**

La terna ***approvalNumber, creationDate, code*** rappresenta un identificativo univoco per la chiave.

Per consentire la paginazione dei risultati, tutti i metodi che restituiscono una lista di chiavi hanno i corrispondenti metodi che restituiscono solo una porzione della lista e un metodo che restituisce il numero di risultati totali.

Accesso al servizio

Il servizio per la gestione dello scaricamento delle chiavi è pubblicato all'indirizzo:

https://www.csrpad.it/csrpad-csrpad/KeyService?wsdl

Metodi implementati

Key[] findValidKeys()

Restituisce un Array di oggetti Key contenente le chiavi associate a tutte le omologazioni attualmente registrate nel Portale del C.S.R.P.A.D..

Key[] findUpdatedValidKeys(Date startUpdateDate, Date endUpdateDate)

Restituisce un Array di oggetti Key contenente le chiavi associate a tutte le omologazioni attualmente registrate nel Portale del C.S.R.P.A.D., filtrando per *updateDate* (data di ultima modifica della chiave inclusa nell'intervallo tra startUpdateDate e endUpdateDate comprese).

Key[] findUpdatedUntilValidKeys(Date endUpdateDate)

Restituisce un Array di oggetti Key contenente le chiavi associate a tutte le omologazioni attualmente registrate nel Portale del C.S.R.P.A.D., filtrando per *updateDate* (data di ultima modifica della chiave precedente o uguale a endUpdateDate).

Key[] findValidKeysByApprovalNumber(String approvalNumber)

Restituisce un Array di oggetti Key, filtrando per *approvalNumber* (numero di omologazione a cui sono associate le chiavi).

Key[] findUpdatedValidKeysByApprovalNumber(String approvalNumber, Date startUpdateDate, Date endUpdateDate)

Restituisce un Array di oggetti Key, filtrando per *approvalNumber*, (numero di omologazione a cui sono associate le chiavi) e *updateDate* (data di ultima modifica della chiave inclusa nell'intervallo tra startUpdateDate e endUpdateDate comprese).).

Key[] findUpdatedUntilValidKeysByApprovalNumber(String approvalNumber, Date endUpdateDate)

Restituisce un Array di oggetti Key, filtrando per *approvalNumber*, (numero di omologazione a cui sono associate le chiavi) e *updateDate* (data di ultima modifica della chiave precedente o uguale a endUpdateDate).

Key[] findValidKeysByType(String encryptionType)

Restituisce un Array di oggetti Key, filtrando per *encryptionType* (tipo crittografia, i cui valori consentiti sono RSA o RC4).

Key[] findUpdatedValidKeysByType(String encryptionType, Date startUpdateDate, Date endUpdateDate)

Restituisce un Array di oggetti Key, filtrando per *encryptionType* (tipo crittografia, i cui valori consentiti sono RSA o RC4) e *updateDate* (data di ultima modifica della chiave inclusa nell'intervallo tra startUpdateDate e endUpdateDate comprese).

Key[] findUpdatedUntilValidKeysByType(String encryptionType, Date endUpdateDate)

Restituisce un Array di oggetti Key, filtrando per *encryptionType* (tipo crittografia, i cui valori consentiti sono RSA o RC4) e *updateDate* (data di ultima modifica della chiave precedente o uguale a endUpdateDate).

Key[] findKeys()

Restituisce un Array di oggetti Key, contenente tutte le chiavi, anche disattivate o scadute, attualmente registrate nel Portale del C.S.R.P.A.D.

Key[] findUpdatedKeys(Date startUpdateDate, Date endUpdateDate)

Restituisce un Array di oggetti Key, contenente tutte le chiavi anche disattivate o scadute, attualmente registrate nel portale del C.S.R.P.A.D., filtrando per *updateDate* (data di ultima modifica della chiave inclusa nell'intervallo tra startUpdateDate e endUpdateDate comprese).

Key[] findUpdatedUntilKeys(Date endUpdateDate)

Restituisce un Array di oggetti Key, contenente tutte le chiavi anche disattivate o scadute, attualmente registrate nel portale del C.S.R.P.A.D., filtrando per *updateDate* (data di ultima modifica della chiave precedente o uguale a endUpdateDate).

Key[] findInhibitedKeys(Date inhibitionDate)

Restituisce un Array di oggetti Key, contenente le chiavi la cui data di disattivazione è uguale o successiva a *inhibitionDate*.

Key[] findInhibitedKeysByType (Date inhibitionDate, String encryptionType)

Restituisce un Array di oggetti Key, contenente le chiavi la cui data di disattivazione è uguale o successiva a *inhibitionDate* e il cui tipo corrisponde a *encryptionType* (tipo di crittografia, i cui valori consentiti sono RSA o RC4).

Key findKeyById(String approvalNumber, Date creationDate, String code)

Restituisce una singola chiave, cercando per identificativo della chiave (vedi note introduttive).

Date getSynchroDate()

Restituisce la data corrente del server.

Esempi

Autenticazione

Esempi di autenticazione con login, login plain e logout.

PseudoCodice

```
String conversationId = AuthenticatorService.login("wsuser@csrpad.it", "xxxxxxxx");
Print(conversationId);
```

Risposta

```
4
```

PseudoCodice

```
boolean loggedIn = AuthenticatorService.loginPlain("wsuser@csrpad.it", "xxxxxxxx");
if(loggedIn){
	Print(“true”);
}
```

Risposta

```
true
```

PseudoCodice

```
boolean loggedOut = AuthenticatorService.logout();
if(loggedOut){
	Print(“true”);
}
```

Risposta

```
true
```

Data di sincronizzazione

Esempio di ricezione della data del server.

PseudoCodice

```
Date now = KeyService.getSynchroDate();
Print(now);
```

Risposta

```
Thu Mar 11 10:05:17 CET 2010
```

Scaricamento chiavi di una omologazione

Esempio di scaricamento delle chiavi di una specifica omologazione con visualizzazione dei relativi dettagli.

PseudoCodice

```
Key[] keys = KeyService.findValidKeysByApprovalNumber("OMTESTRSA0001");

for (Key k : keys) {
	Print(k.getKey());
	Print(k.getCode());
	Print(k.getLinkType());
	Print(k.getEncryptionType());
	Print(k.getApprovalNumber());
	Print(k.getCreationDate());
	Print(k.getExpirationDate());
	if (k.getExpirationDate() > getCurrentDate()) {
		Print(“Chiave Scaduta”);
	}
	Print(k.getInhibitionDate());
	if (k.getInhibitionDate() != null) {
		Print(“Chiave disattivata”);
	}
	Print(k.getUpdateDate());
}
```

Risposta

<RSAKeyValue><Modulus>yPIZrZHeMPqVoRnclEwSUdRFkO2k4eEvjBfi6y9aVVpfzgJ34fRL6txizafmrKu8L4rAGklNX95ueMTGmgR84uo1F4sWHtgL4SIb2EE/SJWKMQeSKcEbHfCu+a17Xma40eTd23Ld/LoV5uhs5FSRlVBDlld1APZllcDdAeUKu/k=</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>
00001
RSCE
RSA
om1234T
Mon Feb 08 00:00:00 CET 2010
Fri Mar 12 00:00:00 CET 2010
null
Thu Feb 11 09:46:32 CET 2010

1287abcd
00002
RSCE
RC4
om1234T
Mon Feb 08 00:00:00 CET 2010
Fri Mar 12 00:00:00 CET 2010
null
Thu Feb 11 09:57:53 CET 2010

Scaricamento chiavi disattivate

Esempio di scaricamento delle chiavi la cui data di disattivazione è uguale o successiva al 02-07-2010, con visualizzazione dei relativi dettagli.

PseudoCodice

```
Key[] keys = KeyService.findInhibitedKeys(createDate(7, 2, 2010));

for (Key k : keys) {
        Print(k.getKey());
        Print(k.getCode());
        Print(k.getLinkType());
        Print(k.getEncryptionType());
        Print(k.getApprovalNumber());
        Print(k.getCreationDate());
        if (k.getExpirationDate() > getCurrentDate()){
                Print("Chiave Scaduta");
        }
        Print(k.getInhibitionDate());
        if (k.getInhibitionDate() != null){
                Print("Chiave disattivata");
        }
        Print(k.getUpdateDate());
}
```

Risposta

<RSAKeyValue><Modulus>yPIZrZHeMPqVoRnclEwSUdRFkO2k4eEvjBfi6y9aVVpfzgJ34fRL6txizafmrKu8L4rAGklNX95ueMTGmgR84uo1F4sWHtgL4SIb2EE/SJWKMQeSKcEbHfCu+a17Xma40eTd23Ld/LoV5uhs5FSRlVBDlld1APZllcDdAeUKu/k=</Modulus><Exponent>AQAB</Exponent></RSAKeyValue>
00003
RSCE
RSA
om1234T
Thu Feb 04 00:00:00 CET 2010
Fri Mar 05 00:00:00 CET 2010
Mon Feb 08 00:00:00 CET 2010
Chiave disattivata
Mon Feb 08 18:00:00 CET 2010

1245abcd
00004
RSSE
RC4
om1234T
Thu Feb 04 00:00:00 CET 2010
Fri Mar 05 00:00:00 CET 2010
Mon Feb 08 00:00:00 CET 2010
Chiave disattivata
Mon Feb 08 18:00:00 CET 2010

Scaricamento delle chiavi valide con filtro sulla data di ultima modifica

L'esempio seguente mostra lo scaricamento di tutte le chiavi valide utilizzando la data di ultima modifica come filtro. Verranno scaricate alla prima esecuzione tutte le chiavi attualmente valide e dallo scaricamento successivo solo quelle che saranno state modificate dalla data di ultimo scaricamento.

Per scaricare le chiavi in modalità paginata (modalità consigliata per evitare eventuali timeout nella ricezione della risposta dai web services) vedere l'esempio successivo.

PseudoCodice

```
function Date getLastUpdateDate(){
	// Restituisce la data di ultimo scaricamento delle chiavi (memorizzata in locale su database, file di configurazione o su un sistema analogo), null se è il primo scaricamento
}

function Date setLastUpdateDate(Date data){
	// Salva la data di ultimo scaricamento delle chiavi (memorizzata in locale su database, file di configurazione o su un sistema analogo)
}

function void updateKeys(Key[] keys){
	// Aggiorna le chiavi scaricate (memorizzate in locale su database, file di configurazione o su un sistema analogo)
}


function void main(){
	boolean loggedIn = AuthenticatorService.loginPlain("wsuser@csrpad.it", "xxxxxxxx");
	if(loggedIn){
		Date startUpdateDate = getLastUpdateDate();
		Date endUpdateDate = KeyService.getSynchroDate();

		if (startUpdateDate == null){
			Key[] keys = KeyService.findUpdatedUntilValidKeys(endUpdateDate);
			updateKeys(keys);
			setLastUpdateDate(endUpdateDate);
		}
		else {
			Key[] keys = KeyService.findUpdatedValidKeys(startUpdateDate, endUpdateDate);
			updateKeys(keys);
			setLastUpdateDate(endUpdateDate);
		}

		AuthenticatorService.logout();
	}

}
```

Scaricamento paginato delle chiavi valide con filtro sulla data di ultima modifica

L'esempio seguente mostra lo scaricamento di tutte le chiavi valide utilizzando la data di ultima modifica come filtro gestendone lo scaricamento in modalità paginata. Verranno scaricate alla prima esecuzione tutte le chiavi attualmente valide e dallo scaricamento successivo solo quelle che saranno state modificate dalla data di ultimo scaricamento.

Ogni singolo scaricamento avverrà in modalità paginata: verranno infatti in prima istanza contate il numero di chiavi da scaricare e in base a questo numero verranno scaricate a blocchi tutte le chiavi necessarie.

E' importante notare che nel caso in cui venga modificata una chiave durante lo scaricamento paginato, il risultato della sincronizzazione potrebbe non essere coerente: è quindi fondamentale verificare se il numero di chiavi da scaricare coincide con quelle effettivamente scaricate e in caso negativo sarà necessario effettuare una nuova sincronizzazione.

La modalità paginata è consigliata, soprattutto per evitare eventuali timeout nella ricezione della risposta dai web services

PseudoCodice

```
function Date getLastUpdateDate(){
	// Restituisce la data di ultimo scaricamento delle chiavi (memorizzata in locale su database, file di configurazione o su un sistema analogo), null se è il primo scaricamento
}

function Date setLastUpdateDate(Date data){
	// Salva la data di ultimo scaricamento delle chiavi (memorizzata in locale su database, file di configurazione o su un sistema analogo)
}

function void updateKeys(Key[] keys){
	// Aggiorna le chiavi scaricate (memorizzate in locale su database, file di configurazione o su un sistema analogo)
}

function key[] downloadUpdatedUntilValidKeysPaginated(startCounter,endUpdateDate){
	List listKeys = new List();

	long endCounter = 0;
	long pageSize = 100;
	while (startCounter != endCounter){

		startCounter = KeyService.countUpdatedUntilValidKeys(endUpdateDate);
		long numPage = (startCounter / pageSize) +1;
		if (startCounter % pageSize == 0) numPage--;

		for (int i=0; i< numPage ; i++){

	listKeys.addAll(KeyService.findUpdatedUntilValidKeysPaginated(endUpdateDate,i*pageSize, pageSize));
		}

		endCounter = listKeys.size();
	}

	return listKeys.toArray();
}

function key[] downloadUpdatedValidKeysPaginated(startCounter,startUpdateDate,endUpdateDate){
	List listKeys = new List();

	long endCounter = 0;
	long pageSize = 100;
	while (startCounter != endCounter){

		startCounter = KeyService.countUpdatedValidKeys(startUpdateDate,endUpdateDate);
		long numPage = (startCounter / pageSize) +1;
		if (startCounter % pageSize == 0) numPage--;

		for (int i=0; i< numPage ; i++){
			listKeys.addAll(KeyService.findUpdatedValidKeysPaginated(startUpdateDate,endUpdateDate,i*pageSize, pageSize));
		}

		endCounter = listKeys.size();
```

```
        }

        return listKeys.toArray();
}

function void main(){
        boolean loggedIn = AuthenticatorService.loginPlain("wsuser@csrpad.it", "xxxxxxxx");
        if(loggedIn){
                Date startUpdateDate = getLastUpdateDate();
                Date endUpdateDate = KeyService.getSynchroDate();

                if (startUpdateDate == null){

                        long count = KeyService.countUpdatedUntilValidKeys(endUpdateDate);
                        if (count > 0){
                                Key[] keys = downloadUpdatedUntilValidKeysPaginated(count,endUpdateDate);
                                updateKeys(keys);
                                setLastUpdateDate(endUpdateDate);
                        }
                }
                else {
                        long count = KeyService.countUpdatedValidKeys(startUpdateDate,endUpdateDate);
                        if (count > 0){
                                Key[] keys = downloadUpdatedValidKeysPaginated(count,startUpdateDate,
        endUpdateDate);
                                updateKeys(keys);
                                setLastUpdateDate(endUpdateDate);
                        }


                }

                AuthenticatorService.logout();
        }

}
```

Account di test

Il Portale del C.S.R.P.A.D. fornisce il servizio per lo scaricamento delle chiavi pubbliche relative agli strumenti utilizzati nel processo di revisione dei veicoli. Per poter verificare lo scaricamento delle chiavi tramite web services è stato predisposto un account di test:

USERNAME : utcsrpadws

PASSWORD: gf00g278

che permette di fruire del servizio e scaricare tutte le chiavi fittizie create esclusivamente a scopo di test. Come già anticipato, per poter fruire del servizio e scaricare le chiavi pubbliche ufficiali sarà invece necessario registrarsi nel portale attraverso l'apposita procedura per i costruttori e utilizzare l'account generato per effettuare la login nel servizio di autenticazione dei web services.

L'elenco delle chiavi fittizie è scaricabile al seguente indirizzo:

http://www.csrpad.it/csrpad/pages/public/index.seam?page=chiavi_test

Questo elenco verrà aggiornato nel caso in cui vengano introdotte, modificate o rimosse ulteriori chiavi di test.

Per qualsiasi problema tecnico inviare una email al seguente indirizzo: *supporto@csrpad.it*