راهنمای سرویس PKA-DSS

شرکت پندار کوشک ایمن

واحد امنیت اطلاعات و زیرساخت کلید عمومی

نسخه:	**۲,۱۳**
تاریخ:	**پاییز ۱۴۰۲**
شناسه:	**PKI-DSS-API-DG**
طبقه‌بندی:	**عمومی**

تاریخچه گزارش

نسخه	تاریخ	تهیه کننده/گان	توضیحات
۱,۰	۱۳۹۸/۱۰/۲۳	کارشناس پروژه	نسخه اول جهت انتشار عمومی
۲,۰	۱۳۹۹/۰۴/۲۵	کارشناس پروژه	اضافه شدن سرویس های VAClient, TSAClient ,DSClient
۲,۱	۱۳۹۹/۰۵/۲۵	کارشناس پروژه	اضافه شدن سرویس SignRSA
۲,۲	۱۳۹۹/۰۸/۰۲	کارشناس پروژه	اضافه شدن سرویس‌ها و کنترلرهای Crypto, Login, PKD
۲,۳	۱۴۰۱/۰۴/۲۶	کارشناس پروژه	تغییر نام توابع برخی توابع، حذف توابع اضافی، افزودن توابع جدید
۲,۴	۱۴۰۱/۰۵/۰۲	کارشناس پروژه	افزودن توابع ApiVersion
۲,۵	۱۴۰۱/۰۵/۰۴	کارشناس پروژه	افزودن توابع زیر CmsExtractAttachedMessage, CmsExtractAttachedMessageRaw
۲,۶	۱۴۰۱/۰۵/۱۹	کارشناس پروژه	افزودن توابع زیر: CertificateThumbprint CertificateThumbprintRaw PDFVerifyAndValidateCertificateByKeyUsages PDFVerifyAndValidateCertificateByKeyUsagesRaw ValidateBasicConstraints ValidateBasicConstraintsRaw CmsDigest CmsDigestRaw PutCMSSignature PutCMSSignatureRaw تغییر ترتیب بررسی Ocsp و Crl در تابع ValidateCertificateEntirley
۲,۷	۱۴۰۱/۰۶/۰۵	کارشناس پروژه	افزودن توابع زیر: ValidateCertificateByCRLFull, ValidateCertificateByCRLFullRaw, ValidateCertificateByOCSPFull, ValidateCertificateByOCSPFullRaw

			حذف توابع زیر: PDFVerifyAndValidateCertificateByKeyUsages PDFVerifyAndValidateCertificateByKeyUsagesRaw
۲,۸	۱۴۰۱/۰۶/۱۳	کارشناس پروژه	تغییر نام توابع ذیل: کلمه Full به Ex تغییر داده شد. ValidateCertificateByCRLEx, ValidateCertificateByCRLExRaw, ValidateCertificateByOCSPEx, ValidateCertificateByOCSPExRaw افزودن تابع ValidateCertificateEntirelyEx ValidateCertificateEntirelyExRaw
۲,۹	۱۴۰۱/۰۶/۱۵	کارشناس پروژه	افزودن توابع Sign , SignRaw
۲,۱۰	۱۴۰۱/۰۹/۱۹	کارشناس پروژه	افزودن مقادیر HashAlgorithm
۲,۱۱	۱۴۰۱/۰۹/۲۰	کارشناس پروژه	افزودن پارامتر AuthRequest به توابع کلاس TsaService , DsService برای اتصال به سرورها Sign
۲,۱۲	۱۴۰۲/۰۵/۱۵	کارشناس پروژه	افزودن ۴ api جدید ذیل به سرویس Crypto: - SignByP12Raw - SignByP12 - PdfMultiSignByP12Raw - PdfMultiSignByP12
۲,۱۳	۱۴۰۲/۰۷/۱۶	کارشناس پروژه	افزودن ۲ api جدید ذیل به سرویس Crypto: - MergeCmsAttachSign - MergeCmsAttachSignRaw

فهرست مطالب

۱ مقدمه

در این مستند راهنمایی‌های لازم جهت استفاده از توابع سرویس PKA به منظور انجام عملیات‌های اعتبارسنجی گواهی‌نامه‌ها و امضای اسناد و مهر زمانی و همچنین عملیات‌های صدور گواهینامه و همچنین ابطال گواهینامه و نیز تعلیق و فعال‌سازی گواهینامه ارائه شده است. خدماتی که این دسته از توابع ارائه می‌دهند، در ادامه مورد بررسی قرار خواهد گرفت. .

همچنین این راهنما به صورت اختصاصی برای برنامه‌نویسان و توسعه‌دهندگان نرم‌افزار تهیه شده است و قدر مسلم نیاز به دانش‌های اولیه برنامه‌نویسی دارد.

۲ سرویس اعتبارسنجی(VAService)

سرویس VAService به منظور دسته‌بندی توابع مربوط به بررسی صحت اعتبار گواهینامه در نظر گرفته شده است. توابع موجود در این سرویس، استانداردهای مختلف دریافت وضعیت یک گواهینامه را پیاده‌سازی می‌نماید. خدماتی که این دسته از توابع ارائه می‌دهد، در ادامه مورد بررسی قرار خواهد گرفت.

۲,۱ آدرس فراخوانی سرویس

امکان فراخوانی متدها به صورت REST FUL API در آدرس زیر مهیا شده است:

https://IP/API/VAService

۲,۲ بررسی اعتبار امضای دیجیتال و تاریخ اعتبار گواهینامه

تابعی که این خدمت را ارائه می‌کند، گواهی ارائه‌شده را از نظر اعتبار آن در زمان کنونی و همچنین امضای دیجیتال صادرشده بر روی آن از طرف CA صادرکننده‌ی آن، مورد ارزیابی قرار می‌دهد.

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس VAService به صورت زیر است:

نام تابع	API/VAService/ValidateCertificateRaw	
ورودی‌ها	`Byte[] Base64Certificate`	گواهی مورد نظر جهت بررسی صحت اعتبار در قالب byte[]
خروجی	`bool`	نتیجه‌ی صحت اعتبار گواهی

نام تابع	API/VAService/ValidateCertificate	
ورودی‌ها	`string Bs64Certificate`	گواهی مورد نظر جهت بررسی صحت اعتبار در قالب Base64
خروجی	`bool`	نتیجه‌ی صحت اعتبار گواهی

۲,۳ بررسی اعتبار گواهینامه(ها) با لیست گواهی باطله(CRL)

یکی از استانداردهایی که در زمینه‌ی بررسی اعتبار گواهینامه موجود است، استاندارد CRL یا Certificate Revocation List می‌باشد. این استاندارد به منظور بررسی اعتبار گواهی، به صورت آفلاین مورد استفاده قرار می‌گیرد.

در این متد، ابتدا آدرس Crl از فایل گواهی واکشی می‌شود، در صورتی‌که این آدرس در فایل گواهی وجود نداشته باشد یا دسترسی به آن با خطا مواجه شود، از آدرس مربوط به Crl در vaProfile استفاده می‌شود.

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس VAService به صورت زیر است:

نام تابع	API/VAService/ValidateCertificateByCRLExRaw		

ورودی‌ها
ValidateRequest<byte[]>
{
Certificate,
vaProfile
}
	Byte[]
	Base64Certificate
		گواهی موردنظر جهت بررسی صحت اعتبار در قالب byte[]	

	string vaProfile	نام پروفایل مورداستفاده(اختیاری)	
خروجی	CRLRevocationResult<byte[]>	نتیجه‌ی بازگشتی از بررسی Crl، شامل وضعیت صحت گواهی،زمان ابطال و فایل گواهی	

نام تابع	API/VAService/ValidateCertificateByCRLEx		

ورودی‌ها
ValidateRequest<string[]>
{
Certificate,
vaProfile
}
	string
	Base64Certificate
		گواهی موردنظر جهت بررسی صحت اعتبار	

	string vaProfile	نام پروفایل مورداستفاده(اختیاری)	
خروجی	CRLRevocationResult<string>	نتیجه‌ی بازگشتی از بررسی Crl، شامل وضعیت صحت گواهی،زمان ابطال و فایل گواهی	

دو متد قدیمی ذیل، جهت مطابقت با نسخه‌های قدیمی نگهداری می‌گردد، اما توصیه می‌شود از متدهای بالا جهت اعتبارسنجی گواهی‌ها به صورت آفلاین استفاده گردد:

نام تابع	API/VAService/ValidateCertificateByCRLRaw		

ورودی‌ها
ValidateRequest<byte[]>
{
Certificate,
vaProfile
}
	Byte[]
	Base64Certificate
		گواهی موردنظر جهت بررسی صحت اعتبار در قالب byte[]	

	string vaProfile	نام پروفایل مورداستفاده(اختیاری)	
خروجی	bool	نتیجه‌ی صحت گواهی	

نام تابع	API/VAService/ValidateCertificateByCRL		

ورودی‌ها
ValidateRequest<string>
{
Certificate,
vaProfile
}
	string Base64Certificate	گواهی موردنظر جهت بررسی صحت اعتبار در قالب Base64	

	string vaProfile	نام پروفایل مورداستفاده(اختیاری)	
خروجی	bool	نتیجه‌ی صحت گواهی	

جهت بررسی وضعیت اعتبار فهرستی از گواهی‌ها، از متدهای ذیل استفاده می‌شود، با توجه به این نکته که زنجیره‌ی فهرست گواهی‌ها باید یکسان باشد.

نام تابع	API/VAService/ValidateCertificateListByCRLRaw		
ورودی‌ها ChainValidationRequest<byte[]> { Certificates, vaProfile }	IEnumerable<byte[]> Base64CertificateList	فهرست گواهی‌های موردنظر جهت بررسی صحت اعتبار درقالب byte[]	
	string vaProfile	نام پروفایل مورداستفاده(اختیاری)	
خروجی	IEnumerable<CRLRevocationResult<byte[]>>	فهرستی از نتایج بازگشتی از بررسی Crl، شامل وضعیت صحت گواهی، زمان ابطال و فایل گواهی	

نام تابع	API/VAService/ValidateCertificateListByCRL		
ورودی‌ها ChainValidationRequest<string> { Certificates, vaProfile}	IEnumerable<string> Base64CertificateList	فهرست گواهی‌های موردنظر جهت بررسی صحت اعتبار درقالب Base64	
	string vaProfile	نام پروفایل مورد استفاده(اختیاری)	
خروجی	IEnumerable<CRLRevocationResult<string>>	فهرستی از نتایج بازگشتی از بررسی Crl، شامل وضعیت صحت گواهی، زمان ابطال و فایل گواهی	

۲,۴ بررسی اعتبار گواهینامه با سرویس استعلام آنلاین وضعیت گواهینامه (OCSP)

استاندارد دیگری که برای اعتبارسنجی گواهینامه‌ها مورد استفاده قرار می‌گیرد، استاندارد OCSP یا Online Certificate Status Protocol می‌باشد. این استاندارد وضعیت اعتبار گواهی را به صورت آنلاین مورد بررسی قرار می‌دهد.

در این متد ابتدا آدرس Ocsp از فایل گواهی واکشی می‌شود، در صورتی که این آدرس در فایل گواهی وجود نداشته باشد، اگر vaProfile ارسال شده باشد، از آدرس مربوط به Ocsp در vaProfile استفاده می‌شود.

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس VAService به صورت زیر است:

نام تابع	API/VAService/ValidateCertificateByOCSPExRaw	

ورودی‌ها
ValidateRequest<byte[]>
{
Certificate,
vaProfile
}
	byte[] Bs64Certificate	رشته‌ی گواهی مورد نظر جهت بررسی صحت اعتبار

	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
خروجی	OCSPResponseStatus<byte[]>	نتیجه بازگردانده شده از طرف سرور OCSP شامل وضعیت گواهی، زمان ابطال، علت ابطال و گواهی

نام تابع	API/VAService/ValidateCertificateByOCSPEx	

ورودی‌ها
ValidateRequest<string>
{
Certificate,
vaProfile
}
	string Bs64Certificate	رشته‌ی گواهی مورد نظر جهت بررسی صحت اعتبار در قالب base64

	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
خروجی	OCSPResponseStatus<string>	نتیجه بازگردانده شده از طرف سرور OCSP شامل وضعیت گواهی، زمان ابطال، علت ابطال و گواهی

دو متد قدیمی ذیل، جهت مطابقت با نسخه‌های قدیمی نگهداری می‌گردد، اما توصیه می‌شود از متدهای بالا جهت اعتبارسنجی گواهی‌ها به صورت آنلاین از متدهای بالا استفاده گردد:

تفاوت متدهای قدیمی ذیل، با متدهای بالا در نوع خروجی است که در متدهای ذیل فقط وضعیت گواهی، برگردانده می‌شود.

نام تابع	API/VAService/ValidateCertificateByOCSPRaw	
ورودی‌ها ValidateRequest<byte[]> { Certificate,	Byte[] Bs64Certificate	آرایه‌ی رشته‌ی گواهی مورد نظر جهت بررسی صحت اعتبار
	string vaProfile	نام پروفایل مورد استفاده(اختیاری)

		vaProfile }
خروجی	CertificateStatus	وضعیت بازگردانده شده از طرف سرور OCSP

نام تابع	API/VAService/ValidateCertificateByOCSP	

ورودی‌ها
ValidateRequest<string>
{
Certificate,
vaProfile
}
	string Bs64Certificate	گواهی مورد نظر جهت بررسی صحت اعتبار

	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
خروجی	CertificateStatus	وضعیت بازگردانده شده از طرف سرور OCSP

مقادیر نتیجه بازگشتی(CertificateStatus):

Good = 0,
Revoked = 1,
Unknown = 2

جهت بررسی وضعیت اعتبار فهرستی از گواهی‌ها از متدهای ذیل استفاده می‌گردد، با توجه به این نکته که زنجیره فهرست گواهی‌ها باید یکسان باشد.

نام تابع	API/VAService/ValidateCertificateListByOCSPRaw	
ورودی‌ها ChainValidationRequest<byte[]> { Certificates, vaProfile}	IEnumerable<byte[]> Base64CertificateList	فهرستی از گواهی های مورد نظر جهت بررسی صحت اعتبار در قالب Base64
	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
خروجی	IEnumerable<OCSPResponseStatus<byte[]>>	فهرستی از نتایج وضعیت‌های بازگردانده شده از طرف سرور OCSP، زمان ابطال، دلیل ابطال و فایل گواهی

نام تابع	API/VAService/ValidateCertificateListByOCSP	
ورودی‌ها ChainValidationRequest<string> { Certificates, vaProfile}	IEnumerable<string> Base64CertificateList	فهرستی از گواهی‌های مورد نظر جهت بررسی صحت اعتبار در قالب Base64
	string vaProfile	نام پروفایل مورد استفاده(اختیاری)

خروجی	IEnumerable<OCSPResponseStatus<string>>	فهرستی از وضعیت‌های بازگردانده شده از طرف سرور OCSP، زمان ابطال، دلیل ابطال و فایل گواهی

مقادیر نتیجه بازگشتی(CertificateStatus):

Good = 0,
Revoked = 1,
Unknown = 2

۲,۵ بررسی اعتبار گواهینامه با موارد استفاده گواهی(Key Usage)

با استفاده از این متد می‌توان کاربردهای یک گواهی را بررسی نمود.

در این متد با استفاده از فهرست Keyusages که در تگ مربوطه در vaProfile مشخص شده است، کاربردهای گواهی بررسی می‌گردد.

کاراکتر جداکننده در این لیست علامت | می‌باشد. به‌طور مثال:

KeyUsages="DIGITALSIGNATURE|NONREPUDIATION"

لازم به توضیح است که در فیلد Keyusage، با توجه به نیاز، می‌توان هریک یا چندین مورد از اقلام ذیل را قرار داد:

EncipherOnly ,
CrlSign ,
KeyCertSign ,
KeyAgreement ,
DataEncipherment ,
KeyEncipherment ,
NonRepudiation ,
DigitalSignature ,
DecipherOnly

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس VAService به صورت زیر است:

نام تابع	API/VAService/ValidateCertificateByKeyUsageRaw	
ورودی‌ها ValidateRequest<byte[]> { Certificate, vaProfile}	byte[] Bs64Certificate	گواهی مورد نظر جهت بررسی صحت اعتبار
	string vaProfile	نام پروفایل مورد استفاده
خروجی	bool	وضعیت تایید موارد استفاده (Keyusage) های موجود در گواهی

نام تابع	API/VAService/ValidateCertificateByKeyUsage	
ورودی‌ها ValidateRequest<string> { Certificate, vaProfile}	string Bs64Certificate	گواهی مورد نظر جهت بررسی صحت اعتبار
	string vaProfile	نام پروفایل مورد استفاده
خروجی	bool	وضعیت تایید موارد استفاده (Keyusage) های موجود در گواهی

با استفاده از این متد می‌توان کاربردهای توسعه‌یافته یک گواهی را بررسی نمود.

در این متد با استفاده از فهرست ExtendedKeyUsage هایی که در تگ مربوطه در vaProfile مشخص شده است، کاربردهای توسعه‌یافته گواهی بررسی می‌گردد.

کاراکتر جداکننده در این لیست علامت | می‌باشد. به‌طور مثال:

ExtendedKeyUsage="1.2.840.113583.1.1.5"

در فیلد ExtendedKeyUsage باید با توجه به نیاز، OID هر یک از کاربردها قرارداده شود.

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس VAService به صورت زیر است:

نام تابع	API/VAService/ValidateCertificateByKeyUsageExRaw	
ورودی‌ها ValidationByKeyUsageRequest<byte[]> { Certificate, KeyUsages, ExtendedKeyUsages}	Byte[] Bs64Certificate	گواهی مورد نظر جهت بررسی صحت اعتبار
	IEnumerable<KeyUsage> keyUsageList	موارد استفاده (keyUsage) های مورد نیاز برای بررسی
	IEnumerable<string> extendedKeyUsageList	لیستی از oId های مورد نیاز برای بررسی. این مقدار از فیلد Enhanced Key Usage گواهینامه استخراج می‌شود. بطور مثال: (1.3.6.1.4.1.311.20.2.2)
خروجی	bool	وضعیت تایید موارد استفاده(EnhancedKeyusage) های موجود در گواهی

نام تابع	API/VAService/ValidateCertificateByKeyUsageEx	

ورودی‌ها
ValidationByKeyUsageRequest<string>
{
Certificate,
KeyUsages,
ExtendedKeyUsages
}
	string
	Bs64Certificate
		گواهی مورد نظر جهت بررسی صحت اعتبار

	IEnumerable<KeyUsage> keyUsages	موارد استفاده (EnhancedKeyusage) های مورد نیاز برای بررسی
	IEnumerable<string> extendedKeyUsages	لیستی از oId های مورد نیاز برای بررسی. این مقدار از فیلد EnhancedKeyusage گواهینامه استخراج می‌شود. بطور مثال: (1.3.6.1.4.1.311.20.2.2)
خروجی	bool	وضعیت تایید موارد استفاده (EnhancedKeyusage) های موجود در گواهی

۲,۶ بررسی کلی اعتبار گواهینامه

برای این روش دو متد قدیمی و جدید پیاده‌سازی شده است. برای تطبیق‌پذیری با نسخه‌های قبلی، تابع قدیمی نیز وجود دارد اما توصیه می‌شود که از تابع جدید برای بررسی کلی اقلام گواهی‌نامه استفاده گردد.

متد جدید، به این صورت است که ابتدا کاربردهای گواهی، زنجیره گواهی، سپس وضعیت گواهی با استفاده از Ocsp , Crl بررسی می‌شود، در نهایت تاریخ انقضای گواهی بررسی می‌گردد. و در صورتی‌که در بررسی یکی از این اقلام، اعتبارسنجی گواهی‌نامه انجام نگیرد، علت رد شدن گواهی در خروجی تابع منعکس می‌گردد. همینطور اگر علت رد شدن گواهی، ابطال آن باشد، تاریخ ابطال نیز در خروجی متد قرار داده می‌شود.

لازم به ذکر است که ترتیب بررسی Ocsp و Crl به این صورت است که ابتدا با توجه به آدرس Ocsp داخل گواهی، بررسی صورت می‌گیرد و در صورت عدم وجود آدرس AIA در فایل گواهی، از آدرس Crl داخل فایل گواهی استفاده می‌شود. در صورتی‌که این آدرس نیز وجود نداشته باشد یا دسترسی به آن آدرس با خطا مواجه شود، اگر vaProfile ارسال شده باشد، ابتدا با توجه به آدرس تنظیم شده برای Ocp، و در صورت عدم موفقیت از آدرس تنظیم شده برای Crl استفاده می‌شود.

جزئیات متد جدید ارائه‌دهنده‌ی این خدمت در سرویس VAService به صورت‌های زیر است:

نام تابع	API/VAService/ValidateCertificateEntirelyEXRaw	

ورودی‌ها
ValidateEntirelyRequest
<byte[]>
{
Certificate,
vaProfile
}
	byte[] Base64Certificate	آرایه‌ی رشته‌ی Base64 گواهی مورد نظر جهت بررسی صحت اعتبار در قالب byte[]

	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
خروجی	CertificateValidationResult	وضعیت اعتبارسنجی گواهینامه و تاریخ ابطال گواهی

نام تابع	API/VAService/ValidateCertificateEntirelyEX	

ورودی‌ها
ValidateEntirelyRequest
<string>
{
Certificate,
vaProfile
}
	string Base64Certificate	رشته‌ی Base64 گواهی مورد نظر جهت بررسی صحت اعتبار

	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
خروجی	CertificateValidationResult	وضعیت اعتبارسنجی گواهینامه و تاریخ ابطال گواهی

متدهای قدیمی: در این متدها، فقط وضعیت گواهی در خروجی تابع منعکس می‌گردد.

نام تابع	API/VAService/ValidateCertificateEntirelyRaw	

ورودی‌ها
ValidateEntirelyRequest
<byte[]>
{
Certificate,
vaProfile
}
	byte[] Base64Certificate	گواهی مورد نظر جهت بررسی صحت اعتبار در قالب byte[]

	string vaProfile	نام پروفایل مورد استفاده
خروجی	CertificateValidationStatus	وضعیت اعتبارسنجی گواهینامه

نام تابع	API/VAService/ValidateCertificateEntirely	

ورودی‌ها
ValidateEntirelyRequest
<string>
{
Certificate,
vaProfile
}
	string Base64Certificate	گواهی مورد نظر جهت بررسی صحت اعتبار

	string vaProfile	نام پروفایل مورد استفاده
خروجی	CertificateValidationStatus	وضعیت اعتبارسنجی گواهینامه

مقادیر نتیجه بازگشتی(CertificateValidationResult)

CertificateValidationOK = 0,
PeriodValidationFailed = 1,
ChainValidationFailed = 2,
IntgrityValidationFailed = 3,
KeyUsageValidationFailed = 4,
OCSPValidationRevoked = 5,
OCSPValidationUnKnown = 6,
CRLValidationRevoked = 7,
CRLAndOCSPValidationError = 8,
OCSPValidationException = 9,
CRLValidationUnKnown = 10,
CRLAndOCSPValidationUnknown = 11

۲,۷ دانلود فایل CRL

استاندارد CRL یا Certificate Revocation List که در زمینه‌ی بررسی اعتبار گواهینامه موجود است، نیازمند بررسی سریال گواهینامه با فایل تولید شده توسط CA می‌باشد. متد DownloadCRL جهت دانلود فایل CRL پیاده‌سازی شده است .
جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس VAService به صورت‌های زیر است:

نام تابع	API/VAService/DownloadCRL	
ورودی‌ها	`string crlUrl`	آدرس فایل CRL
خروجی	`byte[]`	فایل دانلود شده CRL

نام تابع	API/VAService/DownloadCRLByCertificateRaw	
ورودی‌ها	`byte[] base64certificate`	گواهی مورد نظر جهت دریافت آدرس CRL
خروجی	`byte[]`	فایل دانلود شده CRL

نام تابع	API/VAService/DownloadCRLByCertificate	
ورودی‌ها	`string base64certificate`	گواهی مورد نظر جهت دریافت آدرس CRL
خروجی	`byte[]`	فایل دانلود شده CRL

۲,۸ بررسی اقلام استاندارد گواهی‌های میانی و ریشه

این متد همه استانداردهای موجود، جهت بررسی اعتبار گواهینامه‌های میانی و ریشه‌ی گواهی ارسال شده را بررسی می‌کند.

فرآیند بررسی این تابع، به این صورت است که صحت مقادیر BasicConstraint, Path Length, Subject Type=CA, keyUsages={CrlSign , KeyCerSign} های گواهی‌های میانی و ریشه‌ی یک گواهی بررسی می‌گردد.

نام تابع	API/VAService/ValidateBasicConstraintsRaw	
ورودی‌ها	byte[] base64certificate	آرایه‌ی رشته base64 گواهی مورد نظر جهت بررسی
خروجی	bool	نتیجه‌ی صحت بررسی گواهی

نام تابع	API/VAService/ValidateBasicConstraints	
ورودی‌ها	string base64certificate	رشته base64 گواهی مورد نظر جهت بررسی
خروجی	bool	نتیجه‌ی صحت بررسی گواهی

۳ سرویس امضای اسناد الکترونیک(DSService)

سرویس DSService توابع مورد نیاز جهت ارسال سند مورد نیاز به سرور امضاکننده‌ی اسناد و همچنین استخراج سند امضا شده از پاسخ دریافتی سرور را ارائه می‌دهند. خدماتی که این دسته از توابع ارائه می‌دهند در ادامه مورد بررسی قرار خواهد گرفت.

۳,۱ آدرس فراخوانی سرویس

امکان فراخوانی متدها به صورت REST FUL API در آدرس زیر مهیا شده است:

https://IP/API/DSService

۳,۲ درخواست امضای سند PDF

این متد فرایند مورد نیاز جهت امضای سند PDF توسط سرور امضای اسناد را پیاده‌سازی می‌کند.

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس DSService به صورت زیر است:

نام تابع	API/DSService/PDFSignRaw		

ورودی‌ها
SignRequest<byte[]>
{
Data,
DSProfile,
AuthRequest
}
	Byte[] PDFBase64	رشته Base64 شامل سند PDF درخواستی جهت امضا توسط سرور امضای اسناد	

	string DSProfile	نام پروفایل مورد استفاده	
	String AuthReuest	رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1	
خروجی	Byte[]	آرایه ای از رشته Base64 شامل سند PDF امضا شده توسط سرور امضای اسناد	

نام تابع	API/DSService/PDFSign		
ورودی‌ها SignRequest<string> { Data, DSProfile, AuthRequest	string PDFBase64	رشته Base64 شامل سند PDF درخواستی جهت امضا توسط سرور امضای اسناد	
	string DSProfile	نام پروفایل مورد استفاده	

}		
رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1	String AuthReuest	
رشته Base64 شامل سند PDF امضا شده توسط سرور امضای اسناد	string	خروجی

۳,۳ درخواست امضای سند CMS

این متد فرایند مورد نیاز جهت امضای سند CMS توسط سرور امضای اسناد را پیاده‌سازی می‌کند.

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس DSService به صورت زیر است:

نام تابع	API/DSService/CMSSignRaw	

ورودی‌ها
CMSSignRequest<byte[]>
{
Data,
DSProfile,
AttachData,
AuthRequest
}
	byte[] Data	آرایه ای از رشته Base64 شامل سند CMS درخواستی جهت امضا توسط سرور امضای اسناد

	string DSProfile	نام پروفایل مورد استفاده
	bool AttachData	مقدار دودویی جهت انتخاب قرارگیری خود اطلاعات در قالب CMS خروجی.
	String AuthReuest	رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1
خروجی	byte[]	رشته Base64 شامل سند CMS امضا شده توسط سرور امضای اسناد

نام تابع	API/DSService/CMSSign	
ورودی‌ها CMSSignRequest<string> { Data, DSProfile, AttachData,	string CMSBase64	رشته Base64 شامل سند CMS درخواستی جهت امضا توسط سرور امضای اسناد
	string DSProfile	نام پروفایل مورد استفاده

AuthRequest }		
رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1	String AuthReuest	
مقدار دودویی جهت انتخاب قرارگیری خود اطلاعات در قالب CMS خروجی.	bool AttachData	
رشته Base64 شامل سند CMS امضا شده توسط سرور امضای اسناد	string	خروجی

۳,۴ درخواست امضای RSA

این متد فرایند مورد نیاز جهت امضای RSA توسط سرور امضای اسناد را پیاده‌سازی می‌کند.

جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس DSService به صورت زیر است:

نام تابع	API/DSService/RSASignRaw	

ورودی‌ها
SignRequest<byte[]>
{
Data,
DSProfile,
AuthRequest
}
	Byte[] Data	رشته Base64 درخواستی جهت امضا توسط سرور امضای اسناد

	string DSProfile	نام پروفایل مورد استفاده
	String AuthReuest	رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1
خروجی	Byte[]	آرایه ای از رشته Base64 امضا شده توسط سرور امضای اسناد

نام تابع	API/DSService/RSASign	

ورودی‌ها
SignRequest<string>
{
Data,
DSProfile,
AuthRequest
}
	string Data	رشته Base64 درخواستی جهت امضا توسط سرور امضای اسناد

	string DSProfile	نام پروفایل مورد استفاده
	String AuthReuest	رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1
خروجی	string	رشته Base64 امضا شده توسط سرور امضای اسناد

۳,۵ درخواست امضای سند XML

این متد فرایند مورد نیاز جهت امضای سند XML توسط سرور امضای اسناد را پیاده‌سازی می‌کند.

جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس DSService به صورت زیر است:

نام تابع	API/DSService/XMLSign	
ورودی‌ها SignRequest<string>	string Data	سند XML درخواستی جهت امضا توسط سرور امضای اسناد

{
Data,
DSProfile,
AuthRequest
}
	string DSProfile	نام پروفایل مورد استفاده

	String AuthReuest	رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1
خروجی	string	رشته XMLBase64 امضا شده توسط سرور امضای اسناد

۴ سرویس مهر زمانی مطمئن (TSAServi ce)

سرویس TSAService توابع مورد نیاز جهت ارسال اطلاعات مورد نیاز جهت صدور مهر زمانی مطمئن بر روی آن به سرور TSA و همچنین دریافت پاسخ دریافتی از سرور را ارائه می‌دهند. خدماتی که این دسته از توابع ارائه می‌دهند در ادامه مورد بررسی قرار خواهد گرفت.

۴,۱ آدرس فراخوانی سرویس

امکان فراخوانی متدها به صورت REST FUL API در آدرس زیر مهیا شده است:

https://IP/API/ TSAService

۴,۲ درخواست مهر زمانی مطمئن

این متد فرآیند موردنیاز جهت صدور مهر زمانی مطمئن بر روی اطلاعات درخواستی توسط سرور TSA را پیاده‌سازی می‌کند. جزئیات متد ارائه‌دهنده‌ی این خدمت در سرویس TSAService به صورت زیر است:

نام تابع	API/TSAService/TstSignRaw	

ورودی‌ها
TSTSignRequest
<byte[]>
{
Message,
RequestCertificate,
TSAProfile,
AuthRequest
}
	Byte[] Message	آرایه ای از رشته Base64 شامل اطلاعات درخواستی جهت صدور مهر زمانی مطمئن بر روی آن توسط سرور TSA

	bool RequestCertificate	مقدار دودویی جهت انتخاب قرار دهی گواهی سرور صادرکننده‌ی مهر زمانی در خروجی دریافتی از سرور
	string TSAProfile	نام پروفایل مورد استفاده.
	string AuthRequest	رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1

خروجی	Byte[]	ارایه ای از رشته Base64 شامل مهر زمانی صادرشده بر روی اطلاعات

نام تابع	API/TSAService/TstSign	

ورودی‌ها
TSTSignRequest <string>
{
Message,
RequestCertificate,
TSAProfile,
AuthRequest
}
	string Message	رشته Base64 شامل اطلاعات درخواستی جهت صدور مهر زمانی مطمئن بر روی آن توسط سرور TSA

	bool RequestCertificate	مقدار دودویی جهت انتخاب قرار دهی گواهی سرور صادرکننده‌ی مهر زمانی در خروجی دریافتی از سرور
	string TSAProfile	نام پروفایل مورد استفاده.
	string AuthRequest	رشته Base64 شامل نام کاربری و رمز عبور سرور Signer با ساختار ذیل: User1:Password1
خروجی	string	رشته Base64 شامل مهر زمانی صادرشده بر روی اطلاعات

۵ سرویس رمزنگاری و رمزگشایی(CryptoService)

CryptoService توابع لازم برای رمزنگاری و رمزگشایی را در قالب یک وب سرویس ارائه می‌دهد. خدماتی که این دسته از توابع ارائه می‌دهند در ادامه مورد بررسی قرار خواهد گرفت.

۵,۱ آدرس فراخوانی سرویس

امکان فراخوانی متدها به صورت REST FUL API در آدرس زیر مهیا شده است:

https://IP/API/CryptoService

۵,۲ مبدل رشته Base۶۴ به Unicode

این تابع یک رشته را به فرمت Base64 دریافت کرده و به فرمت Unicode تبدیل می‌نماید.

نام تابع	Api/CryptoService/Base64ToUnicode	
ورودی‌ها	string b64Str	رشته به فرمت Base64 برای تبدیل به فرمت Unicode
خروجی	string	رشته به فرمت Unicode تولید شده از رشته ورودی

۵,۳ مبدل رشته Uni code به Base۶۴

این تابع یک رشته را به فرمت Unicode دریافت کرده و به فرمت Base64 تبدیل می‌نماید.

نام تابع	Api/CryptoService/UnicodeToBase64	
ورودی‌ها	`string unicodeStr`	رشته به فرمت Unicode برای تبدیل به فرمت Base64
خروجی	`string`	رشته به فرمت Base64 تولید شده از رشته ورودی

۵,۴ استخراج گواهی از قالب CMS

به منظور دستیابی به گواهی‌هایی که در یک قالب CMS موجود می‌باشند و عملیات رمزنگاری با استفاده از آن‌ها انجام شده است، در کلاس Crypto متد زیر در نظر گرفته شده است:

نام تابع	Api/CryptoService/CmsExtractCertificatesRaw	
ورودی‌ها	`Byte[] messgaeSignatureCertificate`	آرایه‌ی رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل گواهی استفاده شده جهت انجام عملیات رمزنگاری بر روی اطلاعات می‌باشد
خروجی	`IEnumerable<byte[]>`	آرایه‌ای از گواهی‌های استخراج‌شده از درون قالب CMS

نام تابع	Api/CryptoService/CmsExtractCertificates	
ورودی‌ها	`string messgaeSignatureCertificate`	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل گواهی استفاده شده جهت انجام عملیات رمزنگاری بر روی اطلاعات می‌باشد
خروجی	`IEnumerable<string>`	آرایه‌ای از گواهی‌های استخراج‌شده از درون قالب CMS

۵,۵ تصدیق امضای دیجیتال در قالب CMS

به منظور تصدیق امضای دیجیتالی که در قالب CMS قرار دارد، در کلاس Crypto دو متد در نظر گرفته شده است. در صورتی که قالب CMS علاوه بر مقدار امضای دیجیتال، حاوی اصل محتوای اطلاعات امضا شده نیز باشد، در این صورت وجود خود قالب CMS برای تصدیق امضای صادرشده بر روی اطلاعات کفایت می‌نماید. متدی که این خدمت را ارائه می‌دهد در جدول زیر آمده است:

نام تابع	API/CryptoService/CmsVerifyAttachRaw	
ورودی‌ها	Byte[] messgaeSignatureCertificate	آرایه ای از رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل اطلاعات امضا شده، مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.
خروجی	bool	مشخص‌کننده‌ی صحت امضای ارسال شده.

نام تابع	API/CryptoService/CmsVerifyAttach	
ورودی‌ها	string messgaeSignatureCertificate	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل اطلاعات امضا شده، مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.
خروجی	bool	مشخص‌کننده‌ی صحت امضای ارسال شده.

در صورتی که قالب CMS حاوی اطلاعات امضا شده نباشد، در این صورت به منظور تصدیق امضا موجود در این قالب، خود اطلاعات امضا شده نیز موردنیاز می‌باشد. به منظور تصدیق امضای دیجیتال در این شرایط، تابع زیر در کلاس Crypto در نظر گرفته شده است:

نام تابع	API/CryptoService/CmsVerifyRaw	

ورودی‌ها
CmsVerifyRequest <byte[]>
{
Signature,
Message
}
	Byte[] Signiture	آرایه‌ای از رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.

	Byte[] Message	آرایه‌ای از رشته‌ای که در قالب Base64 برای امضا ارسال شده است.
خروجی	bool	نتیجه‌ی صحت امضای ارسال شده.

نام تابع	API/CryptoService/CmsVerify	
ورودی‌ها CmsVerifyRequest <string> {	string Signiture	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.

Signature, Message }	string Message	رشته‌ای که در قالب Base64 برای امضا ارسال شده است.
خروجی	bool	نتیجه‌ی صحت امضای ارسال شده.

۵,۶ تصدیق امضای دیجیتال در قالب CMS و اعتبار سنجی گواهی

به منظور تصدیق امضای دیجیتالی که در قالب CMS قرار دارد و نیز اعتبار سنجی گواهی امضاکننده، در کلاس Crypto متدهایی در نظر گرفته شده است. متدی که این خدمت را ارائه می‌دهد در جدول زیر آمده است:

نام تابع	API/CryptoService/CmsVerifyAndValidateCertificateAttachRaw	

ورودی‌ها
PdfVerifyAnd
ValidateRequest
<byte[]>
{
SignedData,
VaProfile
}
	Byte[] SignedData	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل اطلاعات امضا شده، مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.

	string vaProfile	نام پروفایل مورد استفاده
خروجی	IEnumerable <VerificationResult>	آرایه‌ای از نوع VerificationResult که نتیجه تصدیق هر ردیف از امضا را برمی گرداند

نام تابع	API/CryptoService/CmsVerifyAndValidateCertificateAttach	

ورودی‌ها
PdfVerifyAnd
ValidateRequest
<string>
{
SignedData,
VaProfile
}
	string SignedData	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل اطلاعات امضا شده، مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.

	string vaProfile	نام پروفایل مورد استفاده

خروجی	IEnumerable<VerificationResult>	آرایه ای از نوع VerificationResult که نتیجه تصدیق هر ردیف از امضا را برمی گرداند

نتایج بازگشتی(VerificationResult)

VerificationOK = 0,
CertPeriodValidationFailed = 1,
CertChainValidationFailed = 2,
CertIntgrityValidationFailed = 3,
CertKeyUsageValidationFailed = 4,
CertOCSPValidationRevoked = 5,
CertOCSPValidationUnKnown = 6,
CertCRLValidationRevoked = 7,
CertCRLAndOCSPValidationFailed = 8,
VerificationFailed = 9,
CMSDataNotAttached = 10,
CMSFromatIncorrect = 11,
CertPeriodAndTimeMismatch = 12,
SignitureNotFound=13,
InvalidSignDateTime = 14

در صورتی که قالب CMS حاوی اطلاعات امضا شده نباشد، در این صورت به منظور تصدیق امضا موجود در این قالب، خود اطلاعات امضا شده نیز موردنیاز می‌باشد. به منظور تصدیق امضای دیجیتال در این شرایط، تابع زیر در کلاس Crypto در نظر گرفته شده است:

نام تابع	API/CryptoService/CmsVerifyAndValidateCertificateRaw	

ورودی‌ها
CmsVerifyAndValidateRequest<byte[]>{
Signature,
Message,
VaProfile
}
	Byte[] signature	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.

	Byte[] message	اطلاعات امضا شده به فرمت Base64 در ورودی تابع قرار میگیرد.
	string vaProfile	نام پروفایل مورداستفاده
خروجی	IEnumerable<VerificationResult>	آرایه ای از نوع VerificationResult که نتیجه تصدیق هر ردیف از امضا را برمی‌گرداند. مشخص‌کننده‌ی صحت امضای ارسال شده و اعتبار گواهی امضاکننده می‌باشد.

نام تابع	API/CryptoService/CmsVerifyAndValidateCertificate	

ورودی‌ها
CmsVerifyAndValidateRequest<string>{
Signature,
Message,
VaProfile
}
	string signature	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل مقدار امضا و گواهی استفاده شده جهت صدور امضا بر روی اطلاعات می‌باشد.

	string message	اطلاعات امضا شده به فرمت Base64 در ورودی تابع قرار میگیرد.
	string vaProfile	نام پروفایل مورداستفاده
خروجی	IEnumerable<VerificationResult>	آرایه ای از نوع VerificationResult که نتیجه تصدیق هر ردیف از امضا را برمی‌گرداند. مشخص‌کننده‌ی صحت امضای ارسال شده و اعتبار گواهی امضاکننده می‌باشد.

نتایج بازگشتی(VerificationResult)

VerificationOK = 0,
CertPeriodValidationFailed = 1,
CertChainValidationFailed = 2,
CertIntgrityValidationFailed = 3,
CertKeyUsageValidationFailed = 4,
CertOCSPValidationRevoked = 5,
CertOCSPValidationUnKnown = 6,
CertCRLValidationRevoked = 7,
CertCRLAndOCSPValidationFailed = 8,
VerificationFailed = 9,
CMSDataNotAttached = 10,
CMSFromatIncorrect = 11,
CertPeriodAndTimeMismatch = 12,
SignitureNotFound=13,
InvalidSignDateTime = 14

۵,۷ دریافت Policyهای گواهینامه

این متد تمام Policy های یک گواهی‌نامه را به صورت لیستی از رشته‌ها استخراج می کند. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/CertificatePoliciesRaw	
ورودی‌ها	`Byte[] Base64Certificate`	گواهینامه
خروجی	`string[]`	لیست policy های گواهینامه

نام تابع	API/CryptoService/CertificatePolicies	
ورودی‌ها	`string Base64Certificate`	گواهینامه
خروجی	`String[]`	لیست policy های گواهینامه

۵,۸ ایجاد Hash برای امضای یک پیام

به منظور امضای یک پیام، لازم است ابتدا Hash آن پیام به دست آید. در کلاس Crypto متدی برای این کار در نظر گرفته شده است:

نام تابع	API/CryptoService/DigestRaw	

ورودی‌ها
`DigestRequest<byte[]>`
`{`
`Message,`
`HashAlgorithm`
`}`
	`byte[] message`	متن پیام به فرمت Base64 در ورودی تابع قرار می‌گیرد.

	`HashAlgorithm hashAlgorithm`	الگوریتم درهم سازی مورد استفاده.
خروجی	`byte[]`	آرایه‌ی رشته Base64 حاوی Hash پیام

نام تابع	API/CryptoService/Digest	

ورودی‌ها
`DigestRequest<string>`
`{`
`Message,`
`HashAlgorithm}`
	`string message`	متن پیام به فرمت Base64 در ورودی تابع قرار می‌گیرد.

	`HashAlgorithm hashAlgorithm`	الگوریتم درهم سازی مورد استفاده.
خروجی	`string`	رشته Base64 حاوی Hash پیام

از توابع زیر برای گرفتن Hash پیام در قالب Cms استفاده می‌گردد:

نام تابع	API/CryptoService/CmsDigestRaw	

ورودی‌ها
CmsDigestRequest<byte[]>
{
Message,
Certificate,
HashAlgorithm,
SignDate
}
	byte[] message	آرایه‌ی متن پیام به فرمت Base64 در قالب CMS

	byte[] Certificate	گواهی موردنظر
	HashAlgorithm hashAlgorithm	الگوریتم درهم سازی مورد استفاده.
	DateTime SignDate	زمان امضا
خروجی	byte[]	آرایه‌ی رشته Base64 حاوی Hash پیام

نام تابع	API/CryptoService/CmsDigest	

ورودی‌ها
CmsDigestRequest<string>
{
Message,
Certificate,
HashAlgorithm,
SignDate
}
	string message	متن پیام به فرمت Base64

	string Certificate	گواهی موردنظر
	HashAlgorithm hashAlgorithm	الگوریتم درهم سازی مورد استفاده.
	DateTime SignDate	زمان امضا
خروجی	string	رشته Base64 حاوی Hash پیام

نکته مهم: تمامی ورودی‌های مشترک در دو تابع CmsDigest و PutCmsSignature باید دارای مقادیر یکسان باشند.

مقادیر HashAlgorithm:

```
SHA1 = 0,
SHA256 = 1,
SHA384 = 2,
SHA512 = 3
```

۵,۹ قرار دادن Digest امضا در محتوای Cms

از این متد به منظور قرار دادن مقدار Digest امضا شده در CMS استفاده می‌شود و محتوای امضا شده در قالب رشته Base64 بازگشت داده می‌شود.

نام تابع	API/CryptoService/PutCMSSignatureRaw	

ورودی‌ها
CMSSignature<byte[]>
{
Message,
Certificate,
HashAlgorithm,
SignDate,
Signature,
Encapsulate
}
	byte[] message	آرایه‌ی متن پیام به فرمت Base64 در قالب CMS

	byte[] Certificate	گواهی موردنظر
	HashAlgorithm hashAlgorithm	الگوریتم درهم سازی مورد استفاده.
	DateTime SignDate	زمان امضا
	byte[] Signature	آرایه‌ی رشته base64 امضا
	bool Encapsulate	محتوا در امضا کپسوله شود یا خیر
خروجی	byte[]	آرایه‌ی رشته Base64 حاوی Hash پیام

نام تابع	API/CryptoService/PutCMSSignature	

ورودی‌ها
CMSSignature<string>
{
Message,
Certificate,
HashAlgorithm,
SignDate,
Signature,
Encapsulate
}
	string message	متن پیام به فرمت Base64

	string Certificate	گواهی موردنظر
	HashAlgorithm hashAlgorithm	الگوریتم درهم سازی مورد استفاده.
	DateTime SignDate	زمان امضا
	string Signature	رشته base64 امضا
	bool Encapsulate	محتوا در امضا کپسوله شود یا خیر
خروجی	string	رشته Base64 حاوی Hash پیام

نکته مهم: تمامی ورودی‌های مشترک در دو تابع CmsDigest و PutCmsSignature باید دارای مقادیر یکسان باشند.

۵,۱۰ ایجاد Hash برای امضای یک سند PDF

به منظور امضای یک سند PDF، لازم است ابتدا Hash آن سند به دست آید. در کلاس Crypto متدی برای این کار در نظر گرفته شده است:

نکته: در این تابع از الگوریتم SHA1 استفاده شده است. بنابراین برای امضای نتیجه این تابع نیز باید از الگوریتم SHA1 استفاده گردد.

نام تابع	API/CryptoService/PdfDigestRaw	

ورودی‌ها
PDFDigestRequest<byte[]>
{
PdfData,
Certificate,
DateTime,
ImageDataUrl,
CrProfile,
}
	byte[] pdfData	فایل pdf در قالب Base64 در ورودی تابع قرار میگیرد.

	byte[] certificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال.
	DateTime datetime	زمان امضا
	string ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	string crProfile	نام پروفایل استفاده شده جهت امضای فایل pdf
خروجی	byte[]	آرایه‌ی رشته Base64 که حاوی hash فایل pdf

نام تابع	API/CryptoService/PdfDigest	

ورودی‌ها
PDFDigestRequest<byte[]>
{
PdfData,
Certificate,
DateTime,
ImageDataUrl,
CrProfile,
}
	string pdfData	فایل pdf در قالب Base64 در ورودی تابع قرار میگیرد.

	string certificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال.
	DateTime datetime	زمان امضا
	String ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	string crProfile	نام پروفایل استفاده شده جهت امضای فایل pdf
خروجی	string	رشته Base64 که حاوی hash فایل pdf

۵,۱۱ ایجاد Hash برای امضاهای متعدد در یک سند PDF

به منظور امضاهای متعدد یک سند PDF، لازم است ابتدا Hash آن سند به دست آید. در کلاس Crypto دو متد برای این کار در نظر گرفته شده است:

نام تابع	API/CryptoService/PDFDigestRaw	

ورودی‌ها
PDFDigestRequest<byte[]>
{
PdfData,
Certificate,
CrProfile,
DateTime,
ImageDataUrl
}
	byte[] pdfData	فایل pdf در قالب Base64 در ورودی تابع قرار میگیرد.

	byte[] certificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال.
	string crProfile	نام پروفایل مورداستفاده
	DateTime datetime	زمان گرفتن Hash
	string ImageDataUrl	تصویر مورد استفاده
خروجی	byte[]	آرایه‌ی رشته Base64 که حاوی محتویات فایل pdf به صورت Hash شده می‌باشد.

نام تابع	API/CryptoService/PDFDigest	

ورودی‌ها
PDFDigestRequest<string>
{
PdfData,
Certificate,
CrProfile,
DateTime,
ImageDataUrl
}
	string pdfData	فایل pdf در قالب Base64 در ورودی تابع قرار میگیرد.

	string certificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال.
	string crProfile	نام پروفایل مورداستفاده
	DateTime datetime	زمان گرفتن Hash
	string ImageDataUrl	تصویر مورد استفاده
خروجی	string	رشته Base64 که حاوی محتویات فایل pdf به صورت Hash شده می‌باشد.

نام تابع	API/CryptoService/PDFDigestForMultiSignRaw	
ورودی‌ها **MultiSignPdfDigestRequest <byte[]> { PdfData, HashAlgorithm, CertificationLevel, DateTime DateTime, SignerCertificate, ImageDataUrl, Location, LowerLeftX, LowerLeftY, UpperRightX, UpperRightY, Page, Reason, SignatureFieldName }**	byte[] pdfData	آرایه‌ی فایل pdf در قالب Base64 .
	HashAlgorithm HashAlgorithm	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
	CertificationLevel certificationLevel	نوع امضای برروی pdf NOT_CERTIFIED = 0, CERTIFIED_NO_CHANGES_ALLOWED = 1, CERTIFIED_FORM_FILLING = 2, CERTIFIED_FORM_FILLING_AND_ANNOTATION = 3,
	DateTime datetime	زمان Hash
	byte [] signatureCertificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال در قالب Base64
	string ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	string Location	نام موقعیت مکانی
	int LowerLeftX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int LowerLeftY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int Page	صفحه‌ای که تصویر امضا در آن قرار داده شود
	string Reason	متن دلیل امضا
	string signatureFieldName	نام فیلد امضا در pdf
خروجی	Byte[]	آرایه‌ی رشته Base64 که حاوی Hash فایل pdf

نام تابع	API/CryptoService/PDFDigestForMultiSign	
ورودی‌ها **MultiSignPdfDigestRequest <string> { PdfData, HashAlgorithm, CertificationLevel , DateTime DateTime, SignerCertificate, ImageDataUrl, Location, LowerLeftX, LowerLeftY, UpperRightX, UpperRightY, Page, Reason, SignatureFieldName }**	`string pdfData`	فایل pdf در قالب Base64 .
	`HashAlgorithm HashAlgorithm`	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
	`CertificationLevel certificationLevel`	نوع امضای برروی pdf NOT_CERTIFIED = 0, CERTIFIED_NO_CHANGES_ALLOWED = 1, CERTIFIED_FORM_FILLING = 2, CERTIFIED_FORM_FILLING_AND_ANNOTATION = 3,
	`DateTime datetime`	زمان Hash
	`string signatureCertificate`	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال در قالب Base64
	`string ImageDataUrl`	تصویر مورد استفاده در سند امضا شده
	`string Location`	نام موقعیت مکانی
	`int LowerLeftX`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int LowerLeftY`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int UpperRightX`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int UpperRightY`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int Page`	صفحه‌ای که تصویر امضا در آن قرار داده شود
	`string Reason`	متن دلیل امضا
	`string signatureFieldName`	نام فیلد امضا در pdf
خروجی	`string`	رشته Base64 که حاوی Hash فایل pdf

۵,۱۲ استخراج گواهینامه‌ها از فایل PDF

این متد تمامی گواهینامه‌هایی که به منظور امضای فایل PDF مورد استفاده قرار گرفته‌اند را از فایل استخراج می‌کند. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/PdfExtractCertificatesRaw	
ورودی‌ها	byte[] signedPdf	آرایه‌ی رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد
خروجی	IEnumerable<byte[]>	آرایه‌ای از گواهی‌های استخراج‌شده از درون فایل PDF

نام تابع	API/CryptoService/PdfExtractCertificates	
ورودی‌ها	string signedPdf	رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد
خروجی	IEnumerable<string>	آرایه‌ای از گواهی‌های استخراج‌شده از درون فایل PDF

۵,۱۳ استخراج اطلاعات امضاها از فایل PDF

این متد اطلاعات تمامی امضاهای موجود در فایل PDF را استخراج می‌کند. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/PDFExtractSignerInfoRaw	
ورودی‌ها	Byte[] signedPdf	آرایه‌ی رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد
خروجی	IEnumerable<SignerInfo<byte[]>>	آرایه‌ای از اطلاعات امضاهای موجود در گواهی

نام تابع	API/CryptoService/PDFExtractSignerInfo	
ورودی‌ها	string signedPdf	رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد
خروجی	IEnumerable<SignerInfo<string>>	آرایه‌ای از اطلاعات امضاهای موجود در گواهی

۵,۱۴ تصدیق امضای دیجیتال PDF

این متد به منظور تصدیق امضای دیجیتال صادرشده بر روی یک فایل PDF مورد استفاده قرار می گیرد. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/PDFVerifyRaw	
ورودی‌ها	`Byte[] signedPdf`	آرایه‌ی رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد.
خروجی	`bool`	مشخص‌کننده‌ی صحت امضای صادرشده بر روی فایل PDF

نام تابع	API/CryptoService/PDFVerify	
ورودی‌ها	`string signedPdf`	رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد.
خروجی	`bool`	مشخص‌کننده‌ی صحت امضای صادرشده بر روی فایل PDF

۵,۱۵ تصدیق امضای دیجیتال PDF و اعتبارسنجی گواهی‌های امضا

این متد به منظور تصدیق امضای دیجیتال صادرشده بر روی یک فایل PDF مورد استفاده قرار می‌گیرد. همچنین در این روش همه استانداردهای موجود جهت بررسی اعتبار گواهینامه‌های موجود در امضا بررسی می‌شود. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/PDFVerifyAndValidateCertificateRaw	
ورودی‌ها PdfVerifyAnd ValidateRequest <byte[]> { SignedData, VaProfile } خروجی	Byte[] signedData	آرایه‌ی رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد.
	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
	IEnumerable <VerificationResult>	خروجی شامل وضعیت اعتبارسنجی گواهینامه‌ها

نام تابع	API/CryptoService/PDFVerifyAndValidateCertificate	
ورودی‌ها PdfVerifyAnd ValidateRequest <string> { SignedData, VaProfile } خروجی	string signedData	رشته Base64 که حاوی یک فایل PDF امضا شده می‌باشد.
	string vaProfile	نام پروفایل مورد استفاده(اختیاری)
	IEnumerable <VerificationResult>	خروجی شامل وضعیت اعتبارسنجی گواهینامه‌ها

نتایج بازگشتی(VerificationResult)

VerificationOK = 0,
CertPeriodValidationFailed = 1,
CertChainValidationFailed = 2,
CertIntgrityValidationFailed = 3,
CertKeyUsageValidationFailed = 4,
CertOCSPValidationRevoked = 5,
CertOCSPValidationUnKnown = 6,
CertCRLValidationRevoked = 7,
CertCRLAndOCSPValidationFailed = 8,
VerificationFailed = 9,
CMSDataNotAttached = 10,
CMSFromatIncorrect = 11,
CertPeriodAndTimeMismatch = 12,

SignitureNotFound = 13,
InvalidSignDateTime = 14

۵,۱۶ قرار دادن امضا در سند PDF

از این متد به منظور قرار دادن مقدار Digest امضا شده در سند PDF (منطبق با استاندارد PKCS#7) استفاده می‌شود و سند امضا شده در قالب رشته Base64 بازگشت داده می‌شود.

نام تابع	API/CryptoService/PutPDFSignatureRaw	

ورودی‌ها
PDFSignature<byte[]>
{
PdfData,
Signature,
Certificate,
DateTime,
ImageDataUrl,
CrProfile
}
	byte[] pdfData	فایل pdf در قالب Base64 در ورودی تابع قرار میگیرد.

	byte[] Signature	مقدار Digest امضا شده
	byte[] certificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال.
	DateTime datetime	زمان امضا
	string ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	string crProfile	نام پروفایل استفاده شده جهت امضای فایل pdf
خروجی	byte[]	آرایه‌ی رشته Base64 حاوی محتویات فایل pdf

نام تابع	API/CryptoService/PutPDFSignature	

ورودی‌ها
PDFSignature<string>
{
PdfData,
Signature,
Certificate,
DateTime,
ImageDataUrl,
CrProfile
}
	string pdfData	فایل pdf در قالب Base64 در ورودی تابع قرار میگیرد.

	String Signature	مقدار Digest امضا شده
	string certificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال.
	DateTime datetime	زمان امضا
	string ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	string crProfile	نام پروفایل استفاده شده جهت امضای فایل pdf
خروجی	string	رشته Base64 حاوی محتویات فایل pdf

نکته مهم: تمامی ورودی‌های مشترک در دو تابع PdfDigest و PutPDFSignature باید دارای مقادیر یکسان باشند.

۵,۱۷ قرار دادن امضاهای متعدد در سند PDF

از این متد به منظور قرار دادن مقدار Digest امضا شده در یک سند PDF (منطبق با استاندارد PKCS#7) استفاده می‌شود که دارای امضاهای متعددی است. سند امضا شده در قالب رشته Base64 بازگشت داده می‌شود.

نام تابع	API/CryptoService/PutPDFSignatureForMultiSignRaw	

ورودی‌ها
MultiSignPdfDigest Request <byte[]>
{
PdfData,
HashAlgorithm,
CertificationLevel,
DateTime DateTime,
SignerCertificate,
ImageDataUrl,
Location,
LowerLeftX,
LowerLeftY,
UpperRightX,
UpperRightY,
Page,
Reason,
SignatureFieldName
}
	Byte[] pdfData	فایل pdf در قالب Base64 .

	HashAlgorithm HashAlgorithm	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
	CertificationLevel certificationLevel	نوع امضای برروی pdf NOT_CERTIFIED = 0, CERTIFIED_NO_CHANGES_ALLOWED = 1, CERTIFIED_FORM_FILLING = 2, CERTIFIED_FORM_FILLING_AND_ANNOTATION= 3,
	DateTime datetime	زمان Hash
	Byte[] signatureCertificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال در قالب Base64
	string ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	string Location	نام موقعیت مکانی
	int LowerLeftX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int LowerLeftY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int Page	صفحه‌ای که تصویر امضا در آن قرار داده شود
	string Reason	متن دلیل امضا
	string signatureFieldName	نام فیلد امضا در pdf
خروجی	Byte[]	رشته Base64 که حاوی محتویات فایل pdf به صورت امضا شده می‌باشد.

نام تابع	API/CryptoService/PutPDFSignatureForMultiSign	
ورودی‌ها **MultiSignPdfDigest Request <string> { PdfData, HashAlgorithm, CertificationLevel, DateTime DateTime, SignerCertificate, ImageDataUrl, Location, LowerLeftX, LowerLeftY, UpperRightX, UpperRightY, Page, Reason, SignatureFieldName }**	`string pdfData`	فایل pdf در قالب Base64 .
	`HashAlgorithm HashAlgorithm`	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
	`CertificationLevel certificationLevel`	نوع امضای برروی pdf NOT_CERTIFIED = 0, CERTIFIED_NO_CHANGES_ALLOWED = 1, CERTIFIED_FORM_FILLING = 2, CERTIFIED_FORM_FILLING_AND_ANNOTATION= 3,
	`DateTime datetime`	زمان Hash
	`string signatureCertificate`	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال در قالب Base64
	`string ImageDataUrl`	تصویر مورد استفاده در سند امضا شده
	`string Location`	نام موقعیت مکانی
	`int LowerLeftX`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int LowerLeftY`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int UpperRightX`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int UpperRightY`	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	`int Page`	صفحه‌ای که تصویر امضا در آن قرار داده شود
	`string Reason`	متن دلیل امضا
	`string signatureFieldName`	نام فیلد امضا در pdf
خروجی	`string`	رشته Base64 که حاوی محتویات فایل pdf به صورت امضا شده می‌باشد.

نکته مهم: تمامی ورودی های مشترک در تابع PDFDigestForMultiSign و در تابع PutPDFSignatureForMultiSign باید دارای مقادیر یکسان باشند.

۵,۱۸ استخراج گواهی از مهر زمانی

این متد به منظور استخراج گواهینامه از مهر زمانی صادرشده مورد استفاده قرار می‌گیرد. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/TstExtractCertificatesRaw	
ورودی‌ها	`Byte[] tst`	رشته Base64 که حاوی مهر زمانی صادرشده می‌باشد.
خروجی	`IEnumerable<string>`	آرایه‌ای از گواهی‌های استخراج‌شده از مهر زمانی.

نام تابع	API/CryptoService/TstExtractCertificates	
ورودی‌ها	`string tst`	رشته Base64 که حاوی مهر زمانی صادرشده می‌باشند.
خروجی	`IEnumerable<string>`	آرایه‌ای از گواهی‌های استخراج‌شده از مهر زمانی.

۵,۱۹ استخراج زمان از مهر زمانی

این متد به منظور استخراج زمان از مهر زمانی صادرشده مورد استفاده قرار می‌گیرد. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/TstExtractTimeRaw	
ورودی‌ها	`Byte[] tst`	آرایه‌ی رشته Base64 که حاوی مهر زمانی صادرشده می‌باشد.
خروجی	`DateTime`	زمان استخراج‌شده از درون مهر زمانی.

نام تابع	API/CryptoService/TstExtractTime	
ورودی‌ها	`string tst`	رشته Base64 که حاوی مهر زمانی صادرشده می‌باشند.
خروجی	`DateTime`	زمان استخراج‌شده از درون مهر زمانی.

۵,۲۰ تصدیق مهر زمانی

این متد به منظور تصدیق مهر زمانی صادرشده مورد استفاده قرار می‌گیرد. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس Crypto به صورت زیر است:

نام تابع	API/CryptoService/TstVerifyRaw	

ورودی‌ها
TimestampVerifyRequest<byte[]>
{
Message,
Certificate,
Timestamp
}
	Byte[] message	رشته Base64 که مهر زمانی بر روی آن صادرشده است.

	Byte[] cerificate	رشته Base64 گواهینامه که در صورتیکه ارسال نشود از گواهینامه‌ی موجود در رشته امضا شده حاوی مهر زمانی استفاده می‌کند.
	string tst	رشته Base64 که حاوی مهر زمانی صادرشده می‌باشند.
خروجی	bool	مشخص‌کننده‌ی صحت مهر زمانی

نام تابع	API/CryptoService/TstVerify	

ورودی‌ها
TimestampVerifyRequest<string>
{
Message,
Certificate,
Timestamp
}
	String message	رشته Base64 که مهر زمانی بر روی آن صادرشده است.

	String cerificate	رشته Base64 گواهینامه که در صورتیکه ارسال نشود از گواهینامه‌ی موجود در رشته امضا شده حاوی مهر زمانی استفاده می‌کند.
	string tst	رشته Base64 که حاوی مهر زمانی صادرشده می‌باشند.
خروجی	bool	مشخص‌کننده‌ی صحت مهر زمانی

۵,۲۱ امضای الکترونیک

این متد جهت امضای پیام استفاده می‌شود. تعریف متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/CryptoService/SignRaw	

ورودی‌ها
CryptoSignRequest<byte[]>
{
Data,
Certificate,
HashAlgorithm
}
	byte[] Data	آرایه‌ی حاوی رشته Base64 که برای انجام امضای الکترونیک

	Byte[] certificate	گواهی موردنظر جهت استفاده برای امضای دیجیتال
	HashAlgorithm hashAlgorithm	الگوریتم درهم‌سازی موردنظر برای امضای دیجیتال
خروجی	byte[]	آرایه‌ی رشته Base64 حاوی مقدار امضای الکترونیکی

نام تابع	API/CryptoService/Sign	

ورودی‌ها
CryptoSignRequest<string>
{
Data,
Certificate,
HashAlgorithm
}
	string Data	حاوی رشته Base64 که برای انجام امضای الکترونیک

	string certificate	گواهی مورد نظر جهت استفاده برای امضای دیجیتال
	HashAlgorithm hashAlgorithm	الگوریتم درهم‌سازی موردنظر برای امضای دیجیتال
خروجی	string	رشته Base64 حاوی مقدار امضای الکترونیکی

۵,۲۲ بررسی امضای الکترونیک

این متد جهت تصدیق امضای صادرشده بر روی اطلاعات مشخص شده استفاده می‌شود. تعریف متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/CryptoService/VerifyRaw	
ورودی‌ها VerifyRequest<byte[]> { Data, Certificate,	Byte[] data	رشته Base64 که امضای الکترونیک بر روی آن صورت گرفته است.
	Byte[] certificate	گواهی مورد نظر جهت استفاده برای عملیات تصدیق امضای دیجیتال.

Signature, HashAlgorithm }	string signature	رشته Base64 حاوی مقدار امضای صادرشده بر روی اطلاعات.
	HashAlgorithm hashAlgorithm	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
خروجی	bool	مشخص‌کننده‌ی صحت امضای ارسال شده.

نام تابع	API/CryptoService/Verify	

ورودی‌ها
VerifyRequest<string>
{
Data,
Certificate,
Signature,
HashAlgorithm
}
	string data	رشته Base64 که امضای الکترونیک بر روی آن صورت گرفته است.

	string certificate	گواهی مورد نظر جهت استفاده برای عملیات تصدیق امضای دیجیتال.
	string signature	رشته Base64 حاوی مقدار امضای صادرشده بر روی اطلاعات.
	HashAlgorithm hashAlgorithm	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
خروجی	bool	مشخص‌کننده‌ی صحت امضای ارسال شده.

۵,۲۳ اعتبار سنجی سند XML امضا شده

یک سند XML که امضا شده باشد را به کمک تابع XMLVerify می توان اعتبارسنجی کرد.

نام تابع	API/CryptoService/XMLVerify	
ورودی‌ها	`Stream signedXML`	سند XML امضا شده به صورت Stream
خروجی	`bool`	مشخص کننده صحت امضای سند XML

۵,۲۴ رمزگذاری

این متد جهت رمزگذاری اطلاعات مشخص شده استفاده می‌شود. تعریف متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/CryptoService/EncryptRaw		

ورودی‌ها
`CryptoRequest<byte[]>`
`{`
`Certificate,`
`Cipher`
`}`
	`Byte[] Certificate`	رشته Base64 گواهینامه برای عملیات رمزگذاری

	`Byte[] Cipher`	رشته موردنظر برای عملیات رمزگذاری
خروجی	`Byte[]`	رشته Base64 رمزگذاری شده

نام تابع	API/CryptoService/Encrypt		

ورودی‌ها
`CryptoRequest<string>`
`{`
`Certificate,`
`Cipher`
`}`
	`string Certificate`	رشته Base64 گواهینامه برای عملیات رمزگذاری

	`string Cipher`	رشته موردنظر برای عملیات رمزگذاری
خروجی	`Byte[]`	رشته Base64 رمزگذاری شده

۵,۲۵ رمزگشایی

این متد جهت رمزگشایی اطلاعات مشخص شده استفاده می‌شود. تعریف متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/CryptoService/DecryptRaw	
ورودی‌ها	`Byte[] message`	رشته Base64 رمزگذاری شده
خروجی	`Byte[]`	آرایه‌ی رشته Base64 رمزگشایی شده

نام تابع	API/CryptoService/Decrypt	
ورودی‌ها	`string message`	رشته Base64 رمزگذاری شده
خروجی	`string`	رشته Base64 رمزگشایی شده

۵,۲۶ رمزگذاری متقارن

این متد جهت رمزگذاری متقارن اطلاعات مشخص شده استفاده می‌شود. تعریف متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/CryptoService/SymmetricEncryptRaw	

ورودی‌ها
SymmetricDecryptRequest<byte[]>
{
Key,
IV,
SymmetricAlgorithm,
Data
}
	Byte[] key	کلید رمزگذاری

	Byte[] IV	بردار آغازین
	SymmetricAlgorithms SymmetricAlgorithm	الگوریتم استفاده شده در عملیات رمزگذاری AES_CBC128, AES_CBC256, AES_CBC192, AES_ECB128, AES_ECB256, AES_ECB192,
	Byte[] Data	رشته موردنظر برای عملیات رمزگذاری
خروجی	Byte[]	رشته Base64 رمزگذاری شده

نام تابع	API/CryptoService/SymmetricEncrypt	

ورودی‌ها
SymmetricDecryptRequest<string>
{
Key,
IV,
SymmetricAlgorithm,
Cipher
}
	strng key	کلید رمزگذاری

	string IV	بردار آغازین
	SymmetricAlgorithms SymmetricAlgorithm	الگوریتم استفاده شده در عملیات رمزگذاری AES_CBC128, AES_CBC256, AES_CBC192, AES_ECB128, AES_ECB256, AES_ECB192,
	string data	رشته موردنظر برای عملیات رمزگذاری
خروجی	string	رشته Base64 رمزگذاری شده

۵,۲۷ رمزگشایی متقارن

این متد جهت رمزگشایی متقارن اطلاعات مشخص شده استفاده می‌شود. تعریف متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/CryptoService/SymmetricDecryptRaw	

ورودی‌ها
SymmetricDecryptRequest<byte[]>
{
Key,
IV,
SymmetricAlgorithm,
Cipher
}
	Byte[] key	کلید رمزگشایی

	Byte[] IV	بردار آغازین
	SymmetricAlgorithms SymmetricAlgorithm	الگوریتم استفاده شده در عملیات رمزگذاری AES_CBC128, AES_CBC256, AES_CBC192, AES_ECB128, AES_ECB256, AES_ECB192,
	Byte[] cipher	رشته Base64 رمزگذاری شده
خروجی	Byte[]	رشته Base64 رمزگشایی شده

نام تابع	API/CryptoService/SymmetricDecrypt	

ورودی‌ها
SymmetricDecryptRequest<string>
{
Key,
IV,
SymmetricAlgorithm,
Cipher
}
	strng key	کلید رمزگشایی

	string IV	بردار آغازین
	SymmetricAlgorithms SymmetricAlgorithm	الگوریتم استفاده شده در عملیات رمزگذاری AES_CBC128, AES_CBC256, AES_CBC192, AES_ECB128, AES_ECB256, AES_ECB192,
	string cipher	رشته Base64 رمزگذاری شده
خروجی	string	رشته Base64 رمزگشایی شده

۵,۲۸ استخراج پیام از قالب CMSAttached

به منظور دستیابی به پیام که در یک قالب CMS موجود می‌باشد و عملیات رمزنگاری با استفاده از آن انجام شده است، در کلاس Crypto متد زیر در نظر گرفته شده است:

نام تابع	Api/CryptoService/CmsExtractAttachedMessageRaw	
ورودی‌ها	`Byte[] messgaeSignatureCertificate`	آرایه‌ی رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل پیام استفاده شده جهت انجام عملیات رمزنگاری بر روی اطلاعات می‌باشد
خروجی	`byte[]`	آرایه‌ی پیام استخراج‌شده از درون قالب CMS

نام تابع	Api/CryptoService/CmsExtractAttachedMessage	
ورودی‌ها	`string messgaeSignatureCertificate`	رشته Base64 که حاوی یک قالب CMS است. این قالب در درون خود شامل پیام استفاده شده جهت انجام عملیات رمزنگاری بر روی اطلاعات می‌باشد
خروجی	`string`	رشته Base64 پیام استخراج‌شده از درون قالب CMS

۵,۲۹ استخراج Thumbprint از فایل گواهی

به منظور دستیابی به فیلد Thumbprint که در فایل گواهی موجود می‌باشد، در کلاس Crypto متد زیر در نظر گرفته شده است:

نام تابع	Api/CryptoService/CertificateThumbprintRaw	
ورودی‌ها	`Byte[] Base64Certificate`	آرایه‌ی رشته Base64 که حاوی یک فایل گواهی است.
خروجی	`byte[]`	آرایه‌ی رشته Base64 از فیلد Thumbprint فایل گواهی

نام تابع	Api/CryptoService/CertificateThumbprint	
ورودی‌ها	`string Base64Certificate`	رشته Base64 که حاوی یک فایل گواهی است.
خروجی	`string`	رشته Base64 از فیلد Thumbprint فایل گواهی

۵,۳۰ امضای الکترونیک با گواهی P۱۲

به منظور امضای محتوا با فایل گواهی از جنس P12، در کلاس Crypto متد زیر در نظر گرفته شده است:

نام تابع	API/CryptoService/SignByP12Raw	

ورودی‌ها
CryptoSignByP12Request<byte[]>
{
Data,
Certificate,
HashAlgorithm,
Password
}
	byte[] Data	آرایه‌ی حاوی رشته Base64 برای انجام امضای الکترونیک

	Byte[] certificate	گواهی موردنظر جهت استفاده برای امضای دیجیتال
	HashAlgorithm hashAlgorithm	الگوریتم درهم‌سازی موردنظر برای امضای دیجیتال
	string password	رمز عبور فایل گواهی مورد استفاده
خروجی	byte[]	آرایه‌ی رشته Base64 حاوی مقدار امضای الکترونیکی

نام تابع	API/CryptoService/SignByP12	

ورودی‌ها
CryptoSignByP12Request<string>
{
Data,
Certificate,
HashAlgorithm,
Password
}
	string Data	حاوی رشته Base64 برای انجام امضای الکترونیک

	string certificate	گواهی مورد نظر جهت استفاده برای امضای دیجیتال
	HashAlgorithm hashAlgorithm	الگوریتم درهم‌سازی موردنظر برای امضای دیجیتال
	string password	رمز عبور فایل گواهی مورد استفاده
خروجی	string	رشته Base64 حاوی مقدار امضای الکترونیکی

۵,۳۱ امضای فایل PDF با P۱۲

به منظور امضاهای متعدد یک سند PDF، با استفاده از فایل P12 از این متد، در کلاس Crypto استفاده می‌گردد:

نام تابع	API/CryptoService/PdfMultiSignByP12Raw	

ورودی‌ها
PdfMultiSignByP12Request
<byte[]>
{
PdfData,
SignerCertificate,
CertificationLevel,
DateTime,
Reason,
Location,
ImageDataUrl,
Page,
LowerLeftX,
LowerLeftY,
UpperRightX,
UpperRightY,
SignatureFieldName,
HashAlgorithm,
Password
}
	byte[] pdfData	آرایه‌ی فایل pdf در قالب Base64 .

	byte [] signatureCertificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال در قالب Base64
	CertificationLevel certificationLevel	نوع امضای برروی pdf NOT_CERTIFIED = 0, CERTIFIED_NO_CHANGES_ALLOWED = 1, CERTIFIED_FORM_FILLING = 2, CERTIFIED_FORM_FILLING_AND_ANNOTATION = 3,
	DateTime datetime	زمان امضا
	string Reason	متن دلیل امضا
	string Location	نام موقعیت مکانی
	string ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	int Page	صفحه‌ای که تصویر امضا در آن قرار داده شود
	int LowerLeftX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int LowerLeftY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	string signatureFieldName	نام فیلد امضا در pdf
	HashAlgorithm HashAlgorithm	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
	string password	رمز عبور فایل گواهی مورد استفاده
خروجی	Byte[]	آرایه‌ی رشته Base64 که حاوی فایل pdfامضاشده

نام تابع	API/CryptoService/PdfMultiSignByP12	

ورودی‌ها
PdfMultiSignByP12Request<string>
{
PdfData,
SignerCertificate,
CertificationLevel,
DateTime,
Reason,
Location,
ImageDataUrl,
Page,
LowerLeftX,
LowerLeftY,
UpperRightX,
UpperRightY,
SignatureFieldName,
HashAlgorithm,
Password
}
	string pdfData	فایل pdf در قالب Base64

	string signatureCertificate	گواهی مورد نظر جهت استفاده برای عملیات امضای دیجیتال در قالب Base64
	CertificationLevel certificationLevel	نوع امضای برروی pdf NOT_CERTIFIED = 0, CERTIFIED_NO_CHANGES_ALLOWED = 1, CERTIFIED_FORM_FILLING = 2, CERTIFIED_FORM_FILLING_AND_ANNOTATION = 3,
	DateTime datetime	زمان امضا
	string Reason	متن دلیل امضا
	string Location	نام موقعیت مکانی
	string ImageDataUrl	تصویر مورد استفاده در سند امضا شده
	int Page	صفحه‌ای که تصویر امضا در آن قرار داده شود
	int LowerLeftX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int LowerLeftY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightX	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	int UpperRightY	مختصاتی که تصویر امضا در آنجا قرار داده می‌شود.
	string signatureFieldName	نام فیلد امضا در pdf
	HashAlgorithm HashAlgorithm	الگوریتم درهم سازی استفاده شده در عملیات امضای دیجیتال.
	string password	رمز عبور فایل گواهی مورد استفاده
خروجی	string	رشته Base64 که حاوی فایل pdf امضاشده

۵,۳۲ ادغام امضاهای CMS مختلف یک فایل

به منظور ادغام امضاهای متعدد یک محتوای CMS Attache، از این متد در کلاس Crypto استفاده می‌گردد:

نام تابع	API/CryptoService/MergeCmsAttachSignRaw		

ورودی‌ها
CMSAttachSignRequest
<byte[]>
{
CurrentMessgaeSignature
Certificate,
NextMessgaeSignature
Certificate
}
	byte[]	CurrentMessgae
		Signature
		Certificate
			آرایه‌ی حاوی رشته Base64 امضای CMSAttache جاری

	Byte[]	NextMessgae Signature Certificate	آرایه‌ی حاوی رشته Base64 امضای CMSAttache بعدی
خروجی	byte[]		آرایه‌ی رشته Base64 حاوی مقدار ادغام شده ی امضاهای الکترونیکی

نام تابع	API/CryptoService/MergeCmsAttachSign		

ورودی‌ها
CMSAttachSignRequest
<string>
{
CurrentMessgaeSignature
Certificate,
NextMessgaeSignature
Certificate
}
	string	CurrentMessgae
		Signature
		Certificate
			آرایه‌ی حاوی رشته Base64 امضای CMSAttache جاری

	string	NextMessgae Signature Certificate	آرایه‌ی حاوی رشته Base64 امضای CMSAttache بعدی
خروجی	string		رشته Base64 حاوی مقدار ادغام شده ی امضاهای الکترونیکی

۶ سرویس ورود به سیستم(LoginService)

سرویس Login به منظور دسته‌بندی توابع مورد نیاز جهت اجرای روال شناسایی کاربر و ورود به سیستم، ایجاد شده است. متدهای موجود در این سرویس ، توابع مورد نیاز جهت تولید رشته Challenge و تصدیق هویت کاربر می‌باشد.

در فرآیند تشخیص هویت کاربر، سرور با استفاده از متدLoginChallengeاز کلاس Login یک رشته تصادفی را در قالب Base64 تولید نموده و به سمت کاربر ارسال می‌نماید. در سمت کاربر رشته مربوطه با استفاده ازکلید خصوصی موجود در توکن کاربر امضا شده و رشته نهایی که امضای دیجیتال کاربر نیز در آن وجود دارد برای سرور ارسال خواهد شد. سپس سرور بر اساس کلید عمومی موجود در امضای الصاقی به رشته دریافت شده از سمت کاربر، هویت وی را تشخیص داده و در صورت تایید با توجه به سطوح دسترسی تعریف شده در سیستم، ورود کاربر را به سیستم میسر ساخته و منابع مورد درخواست را در اختیار وی قرار خواهد داد.

۶,۱ آدرس فراخوانی سرویس

امکان فراخوانی متدها به صورت REST FUL API در آدرس زیر مهیا شده است:

https://IP/API/LoginService

۶,۲ دریافت رشته کاراکتر تصادفی

این متد مسئول تولید یک رشته تصادفی در قالب Base64 می‌باشد.

نام تابع	API/LoginService/LoginChallengeRaw	
ورودی‌ها	ندارد	
خروجی	Byte[]	آرایه رشته تصادفی در قالب Base64

نام تابع	API/LoginService/LoginChallenge	
ورودی‌ها	ندارد	
خروجی	string	رشته تصادفی در قالب Base64

۶,۳ دریافت فایل تنظیمات XML

این متد مسئول دریافت فایل تنظیمات XML می‌باشد.

جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس LoginService به صورت زیر است:

نام تابع	API/LoginService/CurrentConfig	
ورودی‌ها	ندارد	
خروجی	object	فایل XML در قالب درخواست شده

نام تابع	API/LoginService/CurrentConfigAsXML	
ورودی‌ها	ندارد	
خروجی	string	رشته‌ی حاوی فایل XML

۶,۴ تصدیق هویت کاربر

این متد با استفاده از رشته تولید شده در تابع LoginChallenge و رشته امضا شده دریافتی از سمت کاربر، هویت کاربر را بررسی و نتیجه تصدیق هویت را به همراه گواهی استخراج شده از امضای کاربر در خروجی باز می‌گرداند.

مقدار برگردانده شده برای AuthenticationResult می‌تواند شامل موارد زیر باشد:

UnKnown = 0,
Athenticated = 1,
SignatureVerificationError = 2,
CertificateValidationError = 3,
CertificateCrlCheckError = 4,
CertificateOcspCheckError = 5,
CertificateKeyUsageError = 6,
CertificateEnhancedKeyUsageError = 7

نام تابع	API/LoginService/authenticateRaw		

ورودی‌ها
AuthRequest<byte[]>
{
Random,
CmsSignature,
LoginProfile
}
	Byte[] random	رشته تصادفی ارسال شده سمت کاربر در قالب Base64	

	Byte[] cmsSignature	امضای دریافت شده از سمت کاربر	
	string loginProfile	نام تنظیمات پیکربندی کلاس Login در فایل pktb.xml	
خروجی	AuthResponse<byte[]>	مقدار داده شمارشی مشخص کننده نتیجه تصدیق هویت کاربر	

نام تابع	API/LoginService/authenticateRaw	

ورودی‌ها
AuthRequest <string>
{
Random,
CmsSignature,
LoginProfile
}
	string random	رشته تصادفی ارسال شده سمت کاربر در قالب Base64

	string cmsSignature	امضای دریافت شده از سمت کاربر
	string loginProfile	نام تنظیمات پیکربندی کلاس Login در فایل pktb.xml
خروجی	AuthResponse<string>	مقدار داده شمارشی مشخص‌کننده نتیجه تصدیق هویت کاربر

۷ سرویس ارتباط با مخزن یا دایرکتوری کلید عمومی(PKDSrvice)

در PKDService متدهایی به منظور دسته‌بندی توابع موردنیاز جهت برقراری ارتباط با دایرکتوری کلید عمومی قرار داده شده است. متدهای موجود در این سرویس، توابع مورد نیاز جهت دریافت گواهی خاص از دایرکتوری کلید عمومی و یا قراردهی گواهینامه بر روی این دایرکتوری را ارائه می‌دهند.

۷,۱ آدرس فراخوانی سرویس

امکان فراخوانی متدها به صورت REST FUL API در آدرس زیر مهیا شده است:

https://IP/API/ PKDService

۷,۲ دریافت گواهینامه از مخزن

این متد به منظور دریافت گواهی خاص از دایرکتوری کلید عمومی مورد استفاده قرار می‌گیرد. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس PKDService به صورت زیر است:

نام تابع	API/PKDService/DownloadCertificate	
ورودی‌ها PKDClientRequest{ Path, PkdProfile}	string path	مقدار SubjectDN گواهینامه‌ی مورد درخواست
	string pkdProfile	نام پروفایل مورد استفاده
خروجی	IEnumerable<byte[]>	آرایه‌ای از گواهی‌های دریافت‌شده از دایرکتوری کلید عمومی

نام تابع	API/PKDService/DownloadCertificateAsBase64	
ورودی‌ها PKDClientRequest{ Path, PkdProfile}	string path	مقدار SubjectDN گواهینامه‌ی مورد درخواست
	string pkdProfile	نام پروفایل مورد استفاده
خروجی	IEnumerable<string>	آرایه‌ای از گواهی‌های دریافت‌شده از دایرکتوری کلید عمومی در قالب base64

۷,۳ دریافت CRL از مخزن

این متد به منظور دریافت CRL یا Certificate Revocation List از دایرکتوری کلید عمومی مورد استفاده قرار می گیرد. جزئیات متد ارائه‌دهنده‌ی این خدمت در کلاس PKDService به صورت زیر است:

نام تابع	API/PKDService/DownloadCRL	
ورودی‌ها PKDClientRequest{ Path, PkdProfile}	string path	مقدار SubjectDN گواهینامه‌ی crl مورد درخواست
	string pkdProfile	نام پروفایل مورداستفاده
خروجی	byte[]	آرایه‌ی فایل CRL دریافت شده از مخزن PKD

نام تابع	API/PKDService/DownloadCRLAsBase64	
ورودی‌ها PKDClientRequest{ Path, PkdProfile}	string path	مقدار SubjectDN گواهینامه‌ی crl مورد درخواست
	string pkdProfile	نام پروفایل مورداستفاده
خروجی	string	رشته Base64 فایل CRL دریافت شده از مخزن PKD

۷,۴ دریافت لیست دایرکتوری

این متد به منظور دریافت لیست دایرکتوری‌های موجود در یک مسیر LDAP پیاده‌سازی شده است. جزئیات متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/PKDService/SubDirectoryList	

ورودی‌ها
PKDClientRequest{
Path,
PkdProfile
}
	string rootDirectoryPath	مقدار SubjectDN گواهینامه‌ی مورد درخواست

	string pkdProfile	نام پروفایل مورداستفاده
خروجی	String[]	آرایه ای از مسیر های موجود در دایرکتوری آرایه ای از مسیر های موجود در دایرکتوری

۷,۵ دریافت فایل تنظیمات

جزئیات متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

نام تابع	API/PKDService/CurrentConfig	
ورودی‌ها	ندارد	
خروجی	Configuration	فایل تنظیمات در قالب Tag base configuration

نام تابع	API/PKDService/CurrentConfigAsXML	
ورودی‌ها	ندارد	
خروجی	string	رشته فایل تنظیمات در قالب XML

۸ سرویس دریافت نسخه(Api Version)

۸,۱ آدرس فراخوانی سرویس

امکان فراخوانی متدها به صورت REST FUL API در آدرس زیر مهیا شده است:

https://IP/API/ApiVersion

۸,۲ دریافت نسخه

جزئیات متد ارائه‌دهنده‌ی این خدمت به صورت زیر است:

این متد به صورت Get عمل کرده و روی مرورگر هم قابل اجراست.

نام تابع	API/ApiVersion/GetVersion	
ورودی‌ها	ندارد	
خروجی	string	رشته حاوی نسخه‌ی اپلیکیشن

این متد به صورت Post عمل می‌کند.

نام تابع	API/ApiVersion/Version	
ورودی‌ها	ندارد	
خروجی	string	رشته حاوی نسخه‌ی اپلیکیشن