سازمان امور مالیاتی کشور

دفتر توسعه خوداظهاری و تمکین داوطلبانه

سند

«دستورالعمل فنی اتصال به سامانه مودیان»

شناسه سند:

RC_TICS.IS_v1.6

شهریور ۱۴۰۴

مقدمه

زیرسامانه‌ی جمع‌آوری صورتحساب یکی از زیرسامانه‌های سامانه‌ی مودیان است که وظیفه‌ی دریافت صورتحساب، ارسال به هسته، گرفتن نتیجه اعتبارسنجی صورتحساب و ذخیره‌کردن آن و پاسخ به استعلام‌های صورتحساب‌های ارسالی از سمت مودی را برعهده دارد. این زیرسامانه دارای یک وب‌سرویس می‌باشد که تمامی درخواست‌ها از طریق این وب‌سرویس به سامانه ارسال شده و پاسخ داده می‌شوند. این وب‌سرویس در چهارچوب REST API پیاده سازی شده و فراخوانی آن نیازمند احراز هویت[1] مودی از طریق امضای دیجیتال می‌باشد.

در این سند منابع موجود در این وب سرویس و نحوه‌ی فراخوانی و پاسخ دهی هر کدام شرح داده خواهد شد و در نهایت در قالب یک مثال فرآیند اتصال به سامانه‌ی مودیان و ارسال صورتحساب به طور کامل انجام می‌گیرد.

[1] Authentication

فهرست مطالب

۱ - تعاریف ۴
۲ - دریافت شناسه یکتای حافظه مالیاتی ۵
۳ - مراحل ارسال صورتحساب و استعلام وضعیت آن ۶
۴ - منابع در دسترس و کاربردهای هر کدام ۶
۵ - دریافت چالش تصادفی و احراز هویت ۹
۶ - دریافت اطلاعات سرور ۱۷
۷ - ارسال صورتحساب ۱۸
۸ - استعلام وضعیت صورتحساب‌های ارسالی ۳۳
۹ - استعلام اطلاعات حافظه و مودی ۴۲
۱۰- استعلام وضعیت صورتحساب در کارپوشه ۴۴
۱۱- ارسال پرداخت صورتحساب ۴۵
۱۲-پیوست‌ها ۴۷

۱ – تعاریف

عنوان	تعریف
امضای دیجیتال[۲]	فرآیندی که طی آن فرستنده پیام به وسیله‌ی یک کلید خصوصی، رشته‌ای به نام امضا تولید می‌کند و آن را در کنار اصل پیام برای گیرنده می‌فرستد. گیرنده با در اختیار داشتن کلید عمومی فرستنده می‌تواند نسبت به اصالت فرستنده اطمینان حاصل کند و مطمئن شود محتوای پیام در طول مسیر تغییری نکرده است.
گواهی امضا[۳]	گواهی الکترونیکی امضا برای اشخاص حقیقی یا گواهی الکترونیکی مهر سازمانی برای اشخاص حقوقی صادر شده توسط مراکز میانی صدور گواهی الکترونیکی، شامل کلید عمومی امضا، تاریخ انقضا و اطلاعات شناسایی هویتی افراد که در قالب یک فایل crt یا cer می‌باشد و برای بررسی صحت امضای بسته‌های ارسالی مورد استفاده قرار می‌گیرد.
صورتحساب الکترونیک	صورتحسابی است دارای شماره منحصر به فرد مالیاتی که اطلاعات مندرج در آن، در حافظه مالیاتی فروشنده ذخیره می شود. مشخصات و اقلام اطلاعاتی صورتحساب الکترونیکی، متناسب با نوع کسب و کار توسط سازمان تعیین و اعلام می شود. در مواردی که از دستگاه کارتخوان بانکی یا درگاه پرداخت الکترونیکی به عنوان پایانه فروشگاهی استفاده می شود، رسید یا گزارش الکترونیکی پرداخت خرید صادره در حکم صورتحساب الکترونیکی است.
شماره مالیاتی	شماره منحصر به فرد مشخص‌کننده یک صورتحساب شامل شناسه حافظه مالیاتی صادرکننده صورتحساب، تاریخ صدور، سریال صورتحساب و یک رقم کنترلی برای جلوگیری از صدور شماره مالیاتی‌های نامعتبر.
شناسه یکتای حافظه مالیاتی	شناسه‌ی یکتایی که توسط سازمان اختصاص یافته می‌شود و مودی می‌تواند از طریق کارپوشه آن را دریافت کند و از آن به منظور صدور صورتحساب استفاده می‌شود.

[۲] Digital Signature

[۳] Digital Signature Certificate

۲ – دریافت شناسه یکتای حافظه مالیاتی

مودی جهت صدور و ارسال صورتحساب الکترونیکی نیاز به دریافت شناسه یکتا حافظه مالیاتی دارد. بنابراین می‌بایست به بخش عضویت و ثبت نام کارپوشه خود مراجعه نموده و مراحل زیر را طی نماید:

۱. به ازای هر شناسه یکتا حافظه مالیاتی، یکی از سه حالت ارسال اطلاعات صورتحساب را به شرح ذیل انتخاب کند:

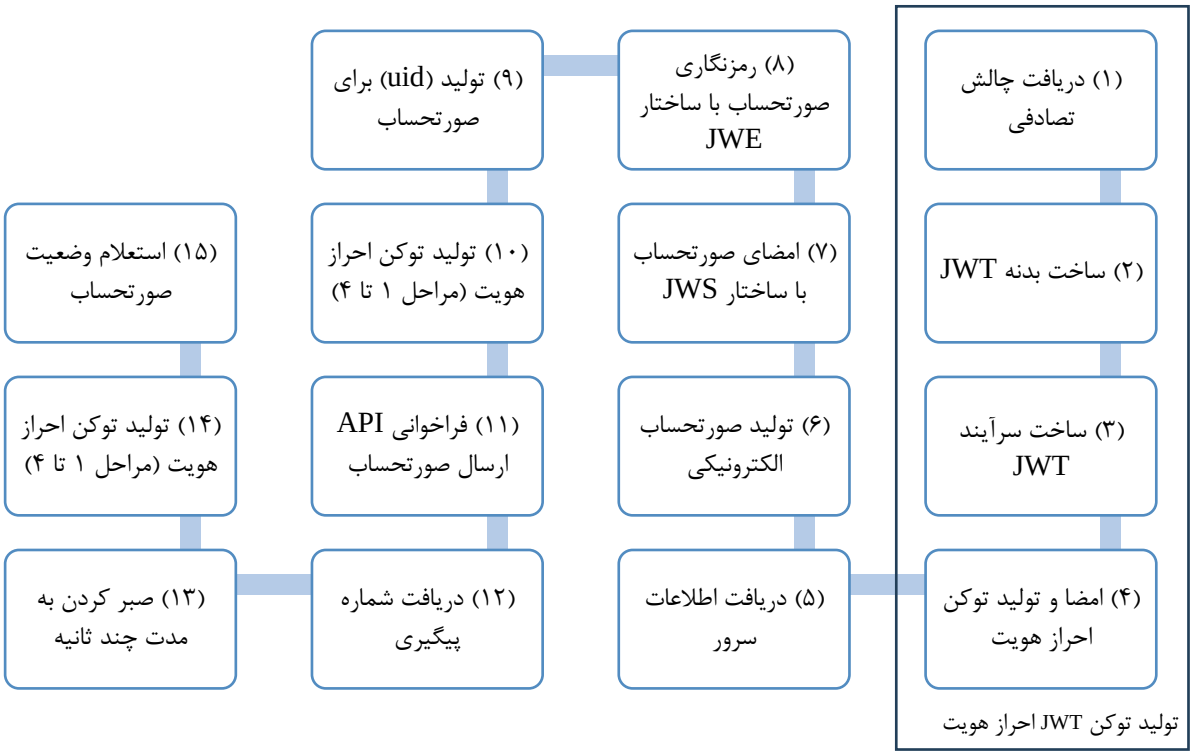
- توسط مودی
- توسط شرکت معتمد / سامانه های دولتی – با کلید مودی
- توسط شرکت معتمد / سامانه های دولتی – با کلید شرکت معتمد / سامانه های دولتی

۲. کلید عمومی/گواهی امضاء دریافتی از مراکز میانی معتبر با طول کلید ۲۰۴۸ بیت را بارگذاری نماید. در این نسخه امکان بارگذاری گواهی امضاء نیز در کارپوشه افزوده شده است.

نکته: در صورتی‌که ارسال غیرمستقیم باشد و شرکت معتمد ارائه کننده خدمات مالیاتی صدور، رمزگذاری و ارسال صورتحساب را به عهده داشته باشد، بارگذاری کلید عمومی/گواهی امضاء توسط مودی ضرورتی ندارد. در این حالت شرکت معتمد ارائه کننده خدمات مالیاتی باید از طریق کارپوشه خود، کلید عمومی/گواهی امضاء مربوطه را به سازمان معرفی نماید.

<u>توجه</u>: این سند با هدف استفاده از نسخه دوم APIهای سامانه مودیان و جهت تسهیل در فرآیند ارسال صورتحساب و اتصال به سامانه مودیان منتشر شده است. در این نسخه قابلیت استفاده از گواهی امضاء برای احراز هویت و ارسال صورتحساب ایجاد شده است. لازم به ذکر است تا اطلاع ثانوی امکان استفاده از نسخه قبلی APIهای سامانه مودیان نیز فراهم می‌باشد که راهنمای استفاده از آن به پیوست این سند بارگذاری گردیده است.

۳ – مراحل ارسال صورتحساب و استعلام وضعیت آن



۴ – منابع در دسترس و کاربردهای هر کدام

وب سرویس جمع‌آوری سامانه‌ی مودیان در آدرس https://tp.tax.gov.ir/requestsmanager در دسترس می‌باشد. منابع موجود در این وب سرویس شامل موارد زیر است (جدول ۱) که توضیحات کامل هر کدام به همراه مثال در ادامه به شکل کامل داده خواهد شد فراخوانی همه‌ی منابع موجود در وب‌سرویس جمع‌آوری سامانه‌ی مودیان نیازمند احراز هویت فراخوانی کننده می‌باشند. به جز منبع دریافت چالش تصادفی که به منظور احراز هویت به کار می‌رود که جزئیات آن در ادامه توضیح داده خواهد شد.

توضیحات	ساختار خروجی	درخواست	آدرس

دریافت چالش تصادفی
راهنما در صفحه ۱۰
	{
	"nonce": "string",
	"expDate": "string"
	}
		curl -X 'GET' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/nonce?timeToLive=20' \
		-H 'accept: */*'
			GET
			https://tp.tax.gov.ir/requestsmanager/api/v2/nonce

ارسال صورتحساب
راهنما در صفحه ۱۸
	{
	"timestamp": 0,
	"result": [
	{
	"uid": "string",
	"packetType": "string",
	"referenceNumber": "string",
	"data": "string"
	}
	]
	}
		curl -X 'POST' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/invoice' \
		-H 'accept: */*' \
		-H 'Authorization: Bearer eyJhbGc [JWT Token] dLcdPeI_9Q' \
		-H 'Content-Type: application/json' \
		-d '[
		{
		"payload": "eyJhbGciOiJ...[JWE]...EEzZe9mxIxw",
		"header": {
		"requestTraceId": "cf019c26-f235-11ed-a05b-0242ac120003",
		"fiscalId": "A11216"
		}
		}
		]'
			POST
			https://tp.tax.gov.ir/requestsmanager/api/v2/invoice

دریافت اطلاعات
پرونده‌ی مودی
راهنما در صفحه ۴۳
	{
	"nameTrade": "string",
	"taxpayerStatus": "string",
	"nationalId": "string"
	}
		curl -X 'GET' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/taxpayer?economicCode=14003778990' \
		-H 'accept: */*' \
		-H 'Authorization: Bearer eyJhbG...[JWT Token]...klQXOuA'
			GET
			https://tp.tax.gov.ir/requestsmanager/api/v2/taxpayer

دریافت اطلاعات
حافظه
راهنما در صفحه ۴۲
	{
	"nameTrade": "string",
	"fiscalStatus": "string",
	"nationalId": "string",
	"economicCode": "string"
	}
		curl -X 'GET' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/fiscal-information?memoryId=A11216' \
		-H 'accept: */*' \
		-H 'Authorization: Bearer eyJhbGc...[JWT Token]...OFh9zw'
			GET
			https://tp.tax.gov.ir/requestsmanager/api/v2/fiscal-information

دریافت اطلاعات
سرور
راهنما در صفحه ۱۷
	{
	"serverTime": 0,
	"publicKeys": [
	{
	"key": "string",
	"id": "string",
	"algorithm": "string",
	"purpose": 0
	}
	]
	}
		curl -X 'GET' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/server-information' \
		-H 'accept: */*' \
		-H 'Authorization: Bearer eyJhbGci...[JWT]...Jvl8fvHm0PKVA'
			GET
			https://tp.tax.gov.ir/requestsmanager/api/v2/server-information

استعلام صورتحساب با شماره پیگیری
راهنما در صفحه ۳۴
	[
	{
	"referenceNumber": "string",
	"uid": "string",
	"status": "string",
	"data": {},
	"packetType": "string",
	"fiscalId": "string",
	"sign": "string"
	}
	]
		curl -X 'GET' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-reference-id?referenceIds=f9173085-2316-4ca6-918e-e41aaf7ef8dd&referenceIds=93367b02-23dd-4568-90e1-2b47d799f361&start=2023-05-14T10%3A00%3A00.000000000%2B03%3A30&end=2023-05-14T21%3A00%3A00.000000000%2B03%3A30' \
		-H 'accept: */*' \
		-H 'Authorization: Bearer eyJhbGc...[JWT TOKEN]...q4RcXogA'
			GET
			https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-reference-id

استعلام با شناسه درخواست (uid)
راهنما در صفحه ۳۷
	[
	{
	"referenceNumber": "string",
	"uid": "string",
	"status": "string",
	"data": {},
	"packetType": "string",
	"fiscalId": "string",
	"sign": "string"
	}
	]
		curl -X 'GET' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-uid?uidList=cb080c58-e36f-4bb0-a932-90f672109fb6&uidList=b3bd6327-1c57-4cae-85ed-5c88de28aea3&fiscalId=A111YO&start=2023-06-10T00%3A00%3A00.000000000%2B03%3A30&end=2023-06-10T23%3A59%3A59.999999999%2B03%3A30' \
		-H 'accept: */*' \
		-H 'Authorization: Bearer eyJhbGciO...[JWT TOKEN]...ski8e-A'
			GET
			https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-uid

استعلام براساس بازه زمانی
راهنما در صفحه ۴۰
	[
	{
	"referenceNumber": "string",
	"uid": "string",
	"status": "string",
	"data": {},
	"packetType": "string",
	"fiscalId": "string",
	"sign": "string"
	}
	]
		curl -X 'GET' \
		'https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry?start=2023-06-10T00%3A00%3A00.000000000%2B03%3A30&end=2023-06-10T23%3A59%3A59.999999999%2B03%3A30&pageNumber=1&pageSize=10' \
		-H 'accept: */*' \
		-H 'Authorization: Bearer eyJhbGci...[JWT TOKEN]...KXzBjRZw'
			GET
			https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry

جدول ۱

۵ – دریافت چالش تصادفی و احراز هویت

مکانیزم احراز هویت در وب‌سرویس جمع‌آوری سامانه‌ی مودیان بر اساس استاندارد «پروتکل احراز هویت در زیرساخت کلید عمومی ایران» منتشر شده توسط «مرکز دولتی صدور گواهی الکترونیک ریشه» می‌باشد که بر مبنای امضای دیجیتال طراحی شده. جهت اتصال به این وب سرویس و پیاده‌سازی این پروتکل لازم است که یک امضای دیجیتال –شامل کلید خصوصی امضاء یا توکن امنیتی، به همراه گواهی امضای الکترونیک معتبر، صادر شده توسط مراکز صدور گواهی الکترونیک میانی– در اختیار داشته باشید.

۵–۱– مراحل احراز هویت هر درخواست

مراحل احراز هویت درخواست به شکل زیر است که در ادامه جزئیات هر یک شرح داده می‌شود:

۱. دریافت چالش تصادفی[۴] (رشته‌ی تصادفی)

۲. قرار دادن clientId و nonce و ساختن payload توکن JWT

۳. ایجاد header توکن JWT براساس ساختار مشخص و قراردادن گواهی امضا در آن

۴. امضای header و payload براساس استاندارد JWS و ساخت توکن

۵. فراخوانی درخواست بعدی با قرار دادن توکن در سرآیند درخواست

فرآیند احراز هویت بدین صورت است که ابتدا شما باید API چالش تصادفی را فراخوانی کنید. این API نیاز به اعتبارسنجی ندارد و یک رشته تصادفی یکبار مصرف و دارای مهلت استفاده محدود تولید کرده و به شما می‌دهد. شما باید قبل از منقضی شدن این رشته آن را به همراه گواهی امضای خود امضا کرده و در درخواست بعدی در بخش Http Header به عنوان توکن ارسال نمائید.

توجه شود که فرآیند گرفتن Nonce و تولید توکن، برای انجام هر درخواست الزامی است و توکنی که برای یک درخواست ساخته می‌شود تنها یکبار قابل استفاده خواهد بود.

---

[۴] Nonce

۵-۱-۱- دریافت چالش تصادفی

این Api یک ورودی timeToLive دارد که زمان اعتبار چالش تصادفی را مشخص می‌کند. در صورتی که این ورودی خالی باشد به طور پیشفرض چالش ۳۰ ثانیه معتبر خواهد بود. در پاسخ رشته تصادفی تولید شده برگردانده می‌شود.

Get Nonce – چالش تصادفی	
آدرس	https://tp.tax.gov.ir/requestsmanager/api/v2/nonce
Method	GET
ورودی	• timeToLive: مدت زمان اعتبار رشته تصادفی به ثانیه ○ نوع ورودی: اختیاری ○ محل قرارگیری: Request Params ○ مقدار پیش‌فرض: ۳۰ ○ مقادیر مجاز: ۱۰ تا ۲۰۰
خروجی	• nonce: رشته تصادفی تولید شده • expDate: زمان انقضای رشته تصادفی

نمونه درخواست دریافت چالش تصادفی:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/nonce?timeToLive=20' \
  -H 'accept: */*'
```

خروجی برابر است با:

```
{
  "nonce": "ab202a55-e106-445c-b2a3-5a7364991a66",
  "expDate": "2023-08-22T16:07:18.277824208Z"
}
```

۵-۱-۲- امضای Nonce و تولید توکن

پس از دریافت رشته تصادفی Nonce، این رشته باید تبدیل به توکن JWS شود و امضا شود. فرمت header و payload توکن JWS تولید شده به شکل زیر است:

ساختار توکن JWS	

Header	یک شیء JSON دارای فیلدهای زیر:
	• alg: الگوریتم امضا: RS256
	• x5c: لیستی که شامل گواهی امضای مودی باشد. کد شده به فرمت Base64
	• sigT: زمان امضای توکن
	○ فرمت:
	`yyyy-MM-dd'T'HH:mm:ss'Z'`
	○ نمونه:
	`2023-05-13T10:44:47Z`
	• crit: لیستی از فیلدهای ضروری در قسمت Header. تنها شامل رشته‌ی "sigT"
	توجه کنید که مقدار فیلد sigT باید طبق استاندارد ISO_8601 باشد و در صورتی که با کاراکتر Z ختم شود باید به منطقه زمانی UTC باشد. در صورتی که بخواهید زمان محلی را در امضای صورتحساب ارسال نمایید می‌توانید در انتهای آن عبارت "+0330" را اضافه کنید:
	`2023-05-13T14:14:47Z+0330`
	همچنین در فیلد x5c که به صورت لیست است، می‌توانید گواهی امضای مرکز میانی صادرکننده‌ی گواهی خود را نیز ارسال نمایید که به عنوان زنجیره گواهی‌های تایید کننده‌ی بسته شناخته می‌شوند تا به یک گواهی مورد اعتماد برسد (Trusted cert). البته این مورد ضرورتی ندارد زیرا گواهی‌های معتبر تمامی مراکز میانی فعال و گواهی مرکز ریشه همگی به عنوان گواهی‌های مورد اعتماد شناخته می‌شوند.

Payload	یک شیء Json دارای دو فیلد زیر: • nonce: رشته تصادفی یکبارمصرف دریافت شده • clientId: شناسه ارسال کننده صورتحساب ○ برای مودیان همان شناسه حافظه صادر کننده صورتحساب ○ برای شرکت‌های معتمد شناسه شرکت معتمد
Signature	ورودی الگوریتم امضا برابر است با: `ASCII(BASE64URL(UTF8(JWS Protected Header)) || '.' || BASE64URL(JWS Payload))`

	الگوریتم مورد استفاده در امضای توکن، RSASSA-PKCS1-v1_5 using SHA-256 می‌باشد. کلید خصوصی امضا باید کلید خصوصی معادل با گواهی فرستاده شده در بخش header باشد که امضای توکن توسط آن گواهی Verify شود.

مطابق استاندارد RFC7515 قالب چیدمان توکن JWS به صورت زیر است:

```
BASE64URL(UTF8(JWS Protected Header))|| '.' ||
BASE64URL(UTF8(JWS Payload))|| '.' ||
BASE64URL(JWS Signature)
```

توجه کنید که فرمت Header بسته‌ی JWS و محتویات بسته (Payload) باید حتما utf-8 باشد و رشته‌ی Header و Payload برای امضا و تولید توکن به فرمت Base64URL کد می‌شود. برای اطلاعات بیشتر سند RFC7515 را مشاهده کنید: https://www.rfc-editor.org/rfc/rfc7515

لازم به ذکر است گواهی استفاده شده در فرآیند امضای توکن احراز هویت می‌بایست معتبر بوده و کد ملی/شناسه ملی موجود در گواهی[۵] با مشخصات ارسال کننده درخواست (صاحب clientId) منطبق باشد.

بعنوان مثال فرض کنید Header توکن ما یک Json به شکل زیر باشد:


```
{"crit":["sigT"],"sigT":"2024-03-
06T13:05:50Z","x5c":["MIIDejCCAmKgAwIBAgIUV27QXqJjK2EgFy9zeYkpsX+ISPswDQYJK
oZIhvcNAQELBQAwcTELMAkGA1UEBhMCSVIxDDAKBgNVBAgMA1RlaDEMMAoGA1UEBwwDVGVoMREw
DwYDVQQKDAhNb2hheW1lbjEMMAoGA1UECwwDVGF4MSUwIwYJKoZIhvcNAQkBFhZtLm1hbHZlcmR
pQG1vaGF5bWVuLmlyMB4XDTIzMDMyNDEzMjgyM1oXDTI0MDMyMzEzMjgyM1owTTEPMA0GA1UEAw
wGQW56YWxpMRcwFQYDVQQKDA5RdW9WYWRpcyBHcm91cDELMAkGA1UEBhMCSVIxFDASBgNVBAUTC
zE0MDAzNzc4OTkwMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzLgyk5KO6+j9d1ud
0ilJArrZ3Whw/w9wEzHB9yXwENRa5fm5AbRukMF5b6VGeKzD6LZuL9+tfdFfWyPcjI++gGNywzR
mEHKpTnzP1t6NyuXKfm4nBVAblsugSw8Y5DEXRfTqILgBWN/pZ4zGlifEALMcGAs7ADcjnv7b/t
M2wxr9rHxsCvW4HvlzQasK8Qr1CrKgT0EI66rSXCHep/uIONDWp0W2OelMlZtM6AAjWXRLGcshP
IHuK+ZLfAFxWtoGonf6qN9ypos2B18D/EFa8WHON62eYKT0kW3jBVa3yPEkRwkdDjDu/3CPzymh
f3WFYwxpb4t35oWb/qUXGVIdvwIDAQABoy4wLDAfBgNVHSMEGDAWgBSx+Oq+RO3x/FmyCp+jcmf
OH+Fn9TAJBgNVHRMEAjAAMA0GCSqGSIb3DQEBCwUAA4IBAQAgKATXlnS+pPtAiRIYGtydVU5Vi7
Aq+D6QW07uFqcB7vBhddN3yX2lVVcwpTNJzhv8UCM+mDMvlmsRVKVtMoo5fHfII92/Wo8rUz1RP
+yhyCk0Vz8I11v+bjLwVur/agC/s5Rf0m66pNNjFZ9J3S2N3lChXYwz2vvA8pdAYvWTu9g5u4FM
FqlsaLwMGC+WaA0g3KYzRkdWRy1vd23hLTUcVsWM8wpgZ11wEGE1khca/Sd0mCU2HG5vIbqFfTj
A6to0fY07CE5fD8aR3UcXjNduosVO52ZqCX5SabrhFS3AGHFRjpFnI5LZespiCXSA8Sv3kOSCSR
QKqFbiwSFM8Zjg"],"alg":"RS256"}
```


همچنین Payload درخواست ما به فرمت utf-8 شیء زیر است:

[۵] فیلد SERIALNUMBER در قسمت SUBJECT گواهی امضاء

```
{
  "nonce": "91dc28af-4c95-47f0-9913-87f8162b1708-1709717749862",
  "clientId": "A11226"
}
```

با امضای رشته با الگوریتم RS256 و الصاق امضای تولید شده به رشته‌ی اصلی (به وسیله‌ی یک ".")، توکن JWT ساخته می‌شود:

eyJjcml0IjpbInNpZ1QiXSwic2lnVCI6IjIwMjQtMDMtMDZUMTM6MDU6NTBaIiwieDVjIjpbIk1
JSURlakNDQW1LZ0F3SUJBZ0lVVjI3UVhxSmpLMkVnRnk5emVZa3BzWCtJU1Bzd0RRWUpLb1pJaH
ZjTkFRRUxCUUF3Y1RFTE1Ba0dBMVVFQmhNQ1NWSXhEREFLQmdOVkJBZ01BMVJsYURFTU1Bb0dBM
VVFQnd3RFZHVm9NUkV3RHdZRFZRUUtEQWhOYjJoaGVXMWxiakVNTUFvR0ExVUVDd3dEVkdGNE1T
VXdJd1lKS29aSWh2Y05BUWtCRmhadExtMWhiSFpsY21ScFFHMXZhR0Y1YldWdUxtbHlNQjRYRFR
Jek1ETXlOREV6TWpneU0xb1hEVEkwTURNeU16RXpNamd5TTFvd1RURVBNQTBHQTFVRUF3d0dRVz
U2WVd4cE1SY3dGUVlEVlFRS0RBNVJkVzlXWVdScGN5QkhjbTkxY0RFTE1Ba0dBMVVFQmhNQ1NWS
XhGREFTQmdOVkJBVVRDekUwTURBek56YzRPVGt3TUlJQklqQU5CZ2txaGtpRzl3MEJBUUVGQUFP
Q0FROEFNSUlCQ2dLQ0FRRUF6TGd5azVLTzYrajlkMXVkMGlsSkFyclozV2h3L3c5d0V6SEI5eVh
3RU5SYTVmbTVBYlJ1a01GNWI2VkdlS3pENkxadUw5K3RmZEZmV3lQY2pJKytnR055d3pSbUVIS3
BUbnpQMXQ2Tnl1WEtmbTRuQlZBYmxzdWdTdzhZNURFWFJmVHFJTGdCV04vcFo0ekdsaWZFQUxNY
0dBczdBRGNqbnY3Yi90TTJ3eHI5ckh4c0N2VzRIdmx6UWFzSzhRcjFDcktnVDBFSTY2clNYQ0hl
cC91SU9ORFdwMFcyT2VsTWxadE02QUFqV1hSTEdjc2hQSUh1SytaTGZBRnhXdG9Hb25mNnFOOXl
wb3MyQjE4RC9FRmE4V0hPTjYyZVlLVDBrVzNqQlZhM3lQRWtSd2tkRGpEdS8zQ1B6eW1oZjNXRl
l3eHBiNHQzNW9XYi9xVVhHVklkdndJREFRQUJveTR3TERBZkJnTlZIU01FR0RBV2dCU3grT3ErU
k8zeC9GbXlDcCtqY21mT0grRm45VEFKQmdOVkhSTUVBakFBTUEwR0NTcUdTSWIzRFFFQkN3VUFB
NElCQVFBZ0tBVFhsblMrcFB0QWlSSVlHdHlkVlU1Vmk3QXErRDZRVzA3dUZxY0I3dkJoZGROM3l
YMmxWVmN3cFROSnpodjhVQ00rbURNdmxtc1JWS1Z0TW9vNWZIZklJOTIvV284clV6MVJQK3loeU
NrMFZ6OEkxMXYrYmpMd1Z1ci9hZ0MvczVSZjBtNjZwTk5qRlo5SjNTMk4zbENoWFl3ejJ2dkE4c
GRBWXZXVHU5ZzV1NEZNRnFsc2FMd01HQytXYUEwZzNLWXpSa2RXUnkxdmQyM2hMVFVjVnNXTTh3
cGdaMTF3RUdFMWtoY2EvU2QwbUNVMkhHNXZJYnFGZlRqQTZ0bzBmWTA3Q0U1ZkQ4YVIzVWNYak5
kdW9zVk81MlpxQ1g1U2FicmhGUzNBR0hGUmpwRm5JNUxaZXNwaUNYU0E4U3Yza09TQ1NSUUtxRm
Jpd1NGTThaamciXSwiYWxnIjoiUlMyNTYifQ.ew0KICAibm9uY2UiOiAiOTFkYzI4YWYtNGM5NS
00N2YwLTk5MTMtODdmODE2MmIxNzA4LTE3MDk3MTc3NDk4NjIiLA0KICAiY2xpZW50SWQiOiAiQ
TExMjI2Ig0KfQ.qPsK_S-76JrYRerFsbl8SqVSNQTcUAd1o496mE_5XOf-
bw1BgLd3bC33Vxrfqs0lpxSm5hyiKpMYkaz-
GPWXt9meoHozRNIFAGimWukE26pH7VJD4vJqfWN784k82XwHRLv1UVoPkzfmtRiFYTwI2hGc1B7
AfY_I6cP5uGMBOY3Kqok0hhlfQyBM_EIoSRNSW3JBAzHQea-
EIm55oXkvXo_2lwsmGHXngFA6qhDbzuHJ2mkItor2HJjZTbND9sxN-
X1ZgSuM7i6izvxxTuSaQMWAfTrEv9d2BzG1ZT1zcFBZVXLAza13Bh74P5DhWsoOR8q0EJvgMvbK
nWaX-H-c_A

تکه‌کد زیر به زبان جاوا عملیات ساخت توکن از روی Nonce را انجام می‌دهد. لازم‌به‌ذکر است تمامی این فرآیندها در کیت توسعه نرم‌افزاری جاوا و .net پیاده‌سازی شده‌اند و این تکه کد صرفا جهت آشنایی بهتر با فرآیند تولید توکن JWT قرار داده شده است. همچنین کلید خصوصی‌ای که فرآیند امضا با آن انجام شده در قسمت پیوست قرار گرفته است:

```
/** Loading Signature Private Key in PKCS#8 format **/
String privateKeyPath = "path/to/private-key.pem";
KeyFactory keyFactory = KeyFactory.getInstance("RSA");
final PEMParser pemParser = new PEMParser(new FileReader(privateKeyPath));
final PrivateKeyInfo pemKeyPair = (PrivateKeyInfo) pemParser.readObject();
```

```
final byte[] encoded = pemKeyPair.getEncoded();
PrivateKey privateKey = keyFactory.generatePrivate(new
PKCS8EncodedKeySpec(encoded));

/** Loading Certificate **/
String certificatePath = "path/to/certificate.crt";
CertificateFactory certificateFactory =
CertificateFactory.getInstance("X.509");
X509Certificate certificate = (X509Certificate)
certificateFactory.generateCertificate(new
FileInputStream(certificatePath));

/** Generate Signature Time **/
String dateFormat = "yyyy-MM-dd'T'HH:mm:ss'Z'";
DateTimeFormatter dateTimeFormatter =
DateTimeFormatter.ofPattern(dateFormat, Locale.ROOT);
String signatureTime =
LocalDateTime.now(ZoneOffset.UTC).format(dateTimeFormatter);

String payload = "{\"nonce\":\"91dc28af-4c95-47f0-9913-87f8162b1708-
1709717749862\",\"clientId\":\"A11226\"}";

/** Generating JWT **/
final JsonWebSignature jws = new JsonWebSignature();

jws.setPayload(payload);
jws.setAlgorithmHeaderValue(AlgorithmIdentifiers.RSA_USING_SHA256);
jws.setKey(privateKey);
jws.setCertificateChainHeaderValue(certificate);
jws.setHeader("sigT", signatureTime);
jws.setHeader("crit", new String[]{"sigT"});

jws.sign();

String jwt = jws.getCompactSerialization();
```

در ابتدا کلید خصوصی امضا به فرمت PKCS#8 در شیء privateKey بارگزاری می‌شود. سپس گواهی امضای مودی در قالب یک شیء از نوع X509Certificate باز می‌شود. سپس تاریخ امضا به فرمت مشخص شده تولید می‌شود. در نهایت با ساخت header درخواست و قرادادن آن در کنار payload داده شده و با در اختیار قرار داشتن کلید خصوصی امضا، توکن jwt ساخته می‌شود. جهت امضای رشته و تولید jwt در این تکه‌کد از کتابخانه‌ی jose-4j نسخه‌ی ۰.۹.۳ استفاده شده است. همچنین تکه کد زیر به زبان NET. همین عملیات را انجام می‌دهد:

```
namespace TaxCollectData.Sample;
using System.Globalization;
using System.Text.Json;
using System.Text.Json.Nodes;
using JWT.Algorithms;
using JWT.Builder;
```

```
using Org.BouncyCastle.Crypto.Parameters;
using Org.BouncyCastle.OpenSsl;
using Org.BouncyCastle.Security;

internal class SignTest
{
    public static void Main(string[] args)
    {
        // Loading Signature Private Key in PKCS#8 format
        var privateKeyPemReader = new
PemReader(File.OpenText("path/to/private-key.pem"));
        var privateKey = DotNetUtilities.ToRSA((RsaPrivateCrtKeyParameters)
privateKeyPemReader.ReadObject());

        // Loading Certificate
        var certPemReader = new PemReader(new
StreamReader("path/to/certificate.crt"));
        var certificate =
DotNetUtilities.ToX509Certificate((Org.BouncyCastle.X509.X509Certificate)
certPemReader.ReadObject());
        var publicKey = DotNetUtilities.ToRSA((RsaKeyParameters)
DotNetUtilities.FromX509Certificate(certificate).GetPublicKey());

        var payload = "{\"nonce\":\"91dc28af-4c95-47f0-9913-87f8162b1708-
1709717749862\",\"clientId\":\"A11226\"}";

        // Generating JWT
        var jws = JwtBuilder.Create()
            .WithAlgorithm(new RS256Algorithm(publicKey, privateKey))
            .AddHeader(HeaderName.X5c, new[]
{Convert.ToBase64String(certificate.GetRawCertData())})
            .AddHeader("sigT", DateTime.UtcNow.ToString("yyyy-MM-
dd'T'HH:mm:ss'Z'", CultureInfo.InvariantCulture))
            .AddHeader("crit", new[] {"sigT"})
            .Encode(JsonSerializer.Deserialize<JsonNode>(payload));

        Console.WriteLine(jws);
    }
}
```

این کد از پکیج‌های زیر برای امضای بسته استفاده می‌نماید:

```
<PackageReference Include="jose-jwt" Version="4.1.0" />
<PackageReference Include="JWT" Version="10.0.2" />
<PackageReference Include="System.Security.Cryptography.Cng"
Version="6.0.0-preview.4.21253.7" />
<PackageReference Include="System.Security.Cryptography.X509Certificates"
Version="4.3.2" />
<PackageReference Include="Portable.BouncyCastle" Version="1.9.0" />
```

۵-۱-۳- استفاده از توکن تولید شده در درخواست بعدی

حال باید با استفاده از این توکن هنگام ارائه‌ی درخواست بعدی خود را به سرور معرفی نمائیم. بدین منظور باید در سرآیند[۶] درخواست HTTP ارسالی، فیلد `Authorization` را برابر با `" Bearer [jwt token] "` قرار دهیم. مثلا فرض کنید می‌خواهیم درخواستی برای گرفتن اطلاعات سرور انجام دهیم. این API از طریق https://tp.tax.gov.ir/requestsmanager/api/v2/server-information در دسترس است.

```
curl --location 'https://tp.tax.gov.ir/requestsmanager/api/v2/server-information' --header 'Authorization: Bearer eyJhbGciO....'
```

در صورتی که در `Authorization` مقدار توکن به درستی وارد شده باشد، امضای توکن صحیح باشد، گواهی امضا معتبر باشد، Nonce هنوز معتبر باشد و تاکنون استفاده نشده باشد، پاسخ به درخواست به شکل زیر خواهد بود:

```
{
  "serverTime": 1683985068801,
  "publicKeys": [
    {
      "key": "MIICIjANBgkq...",
      "id": "6a2bcd88-a871-4245-a393-2843eafe6e02",
      "algorithm": "RSA",
      "purpose": 1
    }
  ]
}
```

بررسی امضای توکن JWS بدین صورت است که هر یک از شرایط زیر برای صحت امضای بسته بررسی می‌شوند و در صورت وجود خطا در هر بخش صحت امضای بسته زیر سوال می‌رود و توکن فرستاده شده مورد قبول واقع نمی‌شود:

۱. ساختار کلی بسته‌ی JWS (header، payload و signature) ارسال شده صحیح باشد.

۲. امضای موجود در بسته‌ی JWS با گواهی امضای فرستاده شده در قسمت header توکن ارسالی Verify شود.

---

[۶] Header

۳. گواهی امضای فرستاده شده در header توسط یکی از مراکز میانی مورد اعتماد زیرسامانه‌ی جمع‌آوری صادر شده باشد و امضای گواهی با یکی از گواهی‌های موجود در مخزن Trusted Cert اعتبارسنجی و Verify شود.

۴. تاریخ انقضای گواهی امضا سپری نشده باشد.

۵. در صورت وجود فیلد CRL Distribution Point در فیلدهای موجود در گواهی (قسمت Extension) استعلام لیست گواهی‌های ابطال شده پیش از موعد از مرکز میانی مورد نظر گرفته می‌شود و بررسی می‌شود گواهی مورد نظر جزو گواهی‌های ابطال شده‌ی پیش از موعد نباشد.

۶. در صورت وجود فیلد Authority Information Access در فیلدهای موجود در گواهی (قسمت Extension) و وجود Access Method برابر با OCSP، از آدرس OCSP موجود در گواهی که آدرس سرور OCSP مرکزمیانی صادر کننده‌ی گواهی می‌باشد، وضعیت گواهی استعلام گرفته می‌شود که گواهی دچار ابطال پیش از موعد نشده باشد.

۶ – دریافت اطلاعات سرور

این وب‌سرویس اطلاعات سرور مانند timestamp و کلیدهای عمومی رمزگذاری سامانه مودیان را برمی‌گرداند. همانطور که گفته شد فراخوانی این API نیاز به احراز هویت دارد و جهت فراخوانی آن باید توکن JWT در قسمت header درخواست در فیلد Authorization فرستاده شود. نحوه‌ی کار این API به صورت زیر است:

دریافت اطلاعات سرور – Get server-information	
آدرس	https://tp.tax.gov.ir/requestsmanager/api/v2/server-information
Method	GET
ورودی	ندارد
خروجی	شیء Json دارای فیلدهای زیر: • serverTime: برچسب زمانی سرور • publicKeys: آرایه Json از کلیدهای عمومی رمزگذاری سرور هر یک شامل:

- key: کلید کد شده به فرمت Base64
- id: شناسه کلید
- algorithm: فعلا برابر با "RSA"
- purpose: مقدار ثابت ۱

نمونه درخواست ارسالی و پاسخ آن:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/server-information' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGciOi...[JWT]...Jvl8fvHm0PKVA'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:


```
{
  "serverTime": 1684055518885,
  "publicKeys": [
    {
      "key":
"MIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEAxdzREOEfk3vBQogDPGTMqdDQ7t0oDh
uKMZkA+Wm1lhzjjhAGfSUOuDvOKRoUEQwP8oUcXRmYzcvCUgcfoRT5iz7HbovqH+bIeJwT4rmLm
FcbfPke+E3DLUxOtIZifEXrKXWgSVPkRnhMgym6UiAtnzwA1rmKstJoWpk9Nv34CYgTk8DKQN5j
QJqb9L/Ng0zOEEtI3zA424tsd9zv/kP4/SaSnbbnj0evqsZ29X6aBypvnTnwH9t3gbWM4I9eAVQ
hPYClawHTqvdaz/O/feqfm06QBFnCgL+CBdjLs30xQSLsPICjnlV1jMzoTZnAabWP6FRzzj6C2s
xw9a/WwlXrKn3gldZ7Ctv6Jso72cEeCeUI1tzHMDJPU3Qy12RQzaXujpMhCz1DVa47RvqiumpTN
yK9HfFIdhgoupFkxT14XLDl65S55MF6HuQvo/RHSbBJ93FQ+2/x/Q2MNGB3BXOjNwM2pj3ojbDv
3pj9CHzvaYQUYM1yOcFmIJqJ72uvVf9Jx9iTObaNNF6pl52ADmh85GTAH1hz+4pR/E9IAXUIl/Y
iUneYu0G4tiDY4ZXykYNknNfhSgxmn/gPHT+7kL31nyxgjiEEhK0B0vagWvdRCNJSNGWpLtlq4F
lCWTAnPI5ctiFgq925e+sySjNaORCoHraBXNEwyiHT2hu5ZipIW2cCAwEAAQ==",
      "id": "6a2bcd88-a871-4245-a393-2843eafe6e02",
      "algorithm": "RSA",
      "purpose": 1
    }
  ]
}
```


۷ – ارسال صورتحساب

فرآیند ارسال صورتحساب هم مانند همه‌ی فرآیندها در وب‌سرویس جمع‌آوری نیاز به احراز هویت فراخوانی کننده دارد. بنابراین از تکرار این مرحله صرف‌نظر می‌کنیم. فرض می‌کنیم عملیات گرفتن nonce، امضای آن و ساخت توکن برای هر درخواست انجام می‌شود. فرض کنید یک صورتحساب الکترونیک در اختیار داریم. برای آشنایی با صورتحساب الکترونیکی و فیلدهای آن به سند «دستورالعمل صدور صورتحساب

الکترونیکی» مراجعه نمائید. صورتحساب به فرم یک JSON دارای header (اطلاعات مربوط به خریدار و فروشنده و اطلاعات کلی صورتحساب)، body (لیست اقلام موجود در صورتحساب) و payments (اطلاعات مراحل پرداخت صورتحساب) می‌باشد.

در ابتدا باید با استفاده از شماره منحصربه‌فرد مالیاتی، صورتحساب را به شکل یکتا مشخص نمائیم. سپس باید صورتحساب را امضا کنیم و بسته‌ی JWS تولید نمائیم. در مرحله‌ی بعدی باید از صورتحساب امضا شده، به وسیله کلید عمومی سازمان (که از طریق GET server-information دریافت می‌گردد) بسته‌ی رمزشده‌ی JWE ساخته شود و سپس برای وب‌سرویس جمع‌آوری ارسال شود.

۷-۱- جزئیات فرآیند ارسال صورتحساب

۷-۱-۱- تولید شماره‌ی منحصربه‌فرد مالیاتی TaxId

برای آشنایی با چگونگی تولید شماره مالیاتی به سند «قالب شناسه یکتای حافظه مالیاتی و شماره منحصر به فرد مالیاتی» مراجعه نمایید. همچنین در کیت توسعه نرم‌افزاری (SDK) به زبان‌های جاوا و .net توابعی جهت تولید شماره مالیاتی مطابق با الگوریتم‌های گفته شده پیاده‌سازی شده که نحوه‌ی استفاده از آن‌ها شرح داده خواهد شد.

۷-۱-۲- امضای صورتحساب

امضای صورتحساب نیز مانند امضای توکن JWT می‌باشد بدین صورت که برای امضای صورتحساب باید یک بسته‌ی JWS ساخته شود که ساختار آن مانند جدول زیر باشد:

امضای صورتحساب – JWS	
Header	یک شیء JSON دارای فیلدهای زیر: • alg: الگوریتم امضا: RS256 • x5c: لیستی که شامل گواهی امضای مودی باشد. کد شده به فرمت Base64 • sigT: زمان امضای توکن Format: yyyy-MM-dd'T'HH:mm:ss'Z' Example: 2023-05-13T10:44:47Z

	• crit: لیستی از فیلدهای ضروری در قسمت Header. تنها شامل رشته‌ی "sigT"
Payload	شیء JSON صورتحساب
Signature	ورودی الگوریتم امضا: ASCII(BASE64URL(UTF8(JWS Protected Header)) || '.' || BASE64URL(JWS Payload)) الگوریتم: RSASSA-PKCS1-v1_5 using SHA-256 کلید خصوصی: کلید خصوصی متناظر با گواهی فرستاده شده در بخش Header

ساختار بسته‌ی JWS صورتحساب دقیقا مانند توکن JWS می‌باشد. با این تفاوت که payload آن JSON صورتحساب می‌باشد. توجه کنید که رشته‌های Header و Payload باید به فرمت utf8 بوده و برای امضا و تولید بسته JWS به فرمت Base64URL کد شوند.

به عنوان نمونه فرض کنید Json صورتحساب ما بعد از تولید شماره مالیاتی به شکل زیر است:


```
{"header":{"taxid":"A1121604C220002F095011","inno":"49321217","indatim":168
3997837988,"inty":1,"inp":1,"ins":1,"tins":"14003778990","tob":2,"bid":"101
00302746","tinb":"10100302746","tprdis":20000,"tdis":500,"tadis":19500,"tva
m":1755,"todam":0,"tbill":21255,"setm":2},"body":[{"sstid":"2710000138624",
"sstt":"فولاد صنعت قطعات سرسیلندر
سازی","mu":"164","am":2,"fee":10000,"prdis":20000,"dis":500,"adis":19500,"v
ra":9,"vam":1755,"tsstam":21255}],"payments":[]}
```


حال یک شیء JWS می‌سازیم که Header آن مقدار زیر باشد:


```
{"crit":["sigT"],"sigT":"2024-03-
06T13:16:11Z","x5c":["MIIE7DCCA9SgAwIBAgISAM4ybFjIYlezw/6Ly2CQYYNBMA0GCSqGS
Ib3DQEBCwUAMIG7MQswCQYDVQQGEwJJUjEZMBcGA1UECgwQTm9uLUdvdmVybm1lbnRhbDEkMCIG
A1UECwwbQW1uYWZ6YXIgR29zdGFyLWUgU2hhcmlmIENvMTQwMgYDVQQLDCtQYXJzU2lnbiBJbnR
lcm1lZGlhdGUgQ2VydGlmaWNhdGUgQXV0aG9yaXR5MTUwMwYDVQQDDCxQYXJzU2lnbiBQcml2YX
RlIEludGVybWVkaWF0ZSBTaWx2ZXIgQ0EgLSBHMzAeFw0yMzEwMTgwNjQyMTZaFw0yNDEwMTIwN
jQyMTZaMIGyMSAwHgYDVQQDDBdBbGkgU2hpcm1vaGFtYWRpIFtTaWduXTELMAkGA1UEBhMCSVIx
DzANBgNVBCoMBti52YTbjDETMBEGA1UEBwwK2KrZh9ix2KfZhjEVMBMGA1UECgwMVW5hZmZpbGl
hdGVkMRMwEQYDVQQFEwowMDIzNDU3NzA4MRMwEQYDVQQIDArYqtmH2LHYp9mGMRowGAYDVQQEDB
HYtNuM2LEg2YXYrdmF2K/bjDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAIm7YPJLA
bIyHftOB9SsMps1OeFMJ5+bRZLa4dE5cf404Qb2K6m+ITxT8sDO09mDLqbfxTCq1+ehlbAl/yug
XdF5+rGYQYX/yaJf+vkaAq162vqdRvbVvkhNr1XI2rI+sXfGJBcGnceskoVf7rm+RTciOmAA3Zh
cjdU3jFmU1/Am8VBJ83kK8Qf8I6ZtFZhHL1Yp051xLVs5QI4/lwmKxkEkmG+VgGV22Mkve3Yw1P
oeMyc0O110BvFd3iiEnAUtHuIKF6cFwWia0hIJLvd+s/W7LuVQMNVYeu+J64zDqYBwDaSt4L+eo
fT6ksg6aslQSZdf/7VGyDGIqcjTet1E/C8CAwEAAaOB8DCB7TAdBgNVHQ4EFgQUepvjaOKnzm8N
4C/+g1F4neodzaMwHwYDVR0jBBgwFoAUFMcOIHyJ0RZ/yrlp6IfJLEaC+lUwNAYDVR0fBC0wKzA
poCegJYYjaHR0cDovL2wyLnBhcnNzaWduY2EuaXIvUFNfTDJDQS5jcmwwDgYDVR0PAQH/BAQDAg
bAMBYGA1UdJQEB/wQMMAoGCCsGAQUFBwMCMDcGCCsGAQUFBwEBBCswKTAnBggrBgEFBQcwAYYba
HR0cDovL2wyLnBhcnNzaWduY2EuaXI6ODEvMBQGA1UdIAQNMAswCQYHYIJsZQEBAjANBgkqhkiG
9w0BAQsFAAOCAQEAcw2hdV1YN8xHXsQq4X6YWtgyUsK92CTILb/j2jYfIGXlwLCYxR6zPFuoqmj
VG9Txs3NEPQirsVwqoIUzljTZnBU/WlPjeqPyWadQooRvg/64Ahd88A8YJf3Yvv1Dax8rUANX+A
Qd093OxAr7hfCISTHFmhhw5GoJ153vh0idhekeK6//iwvLugPA9k0Qu0WObWsJ2Ctj2zWhCXMdz
```

IajqkpEmyBJxe4yj9ZGyezjSmPb+khDgvEsbrOPcWy3WTsx80S3Shphl8ywNQTRd6OximDXs14+lL0BlZav6o0DpTlkF40OmoPkoZwQkJNxBrNrKWDeK39vRR56yPP4p+04Ww=="],"alg":"RS256"}

و سپس آن را با استفاده از کلید خصوصی که در اختیار داریم امضا می‌کنیم. حال صورتحساب امضا شده به صورت یک JWS در می‌آید:

eyJjcml0IjpbInNpZ1QiXSwic2lnVCI6IjIwMjQtMDMtMDZUMTM6MTY6MTFaIiwieDVjIjpbIk1JSUU3RENDQTlTZ0F3SUJBZ0lTQU00eWJGaklZbGV6dy82THkyQ1FZWU5CTUEwR0NTcUdTSWIzRFFQkN3VUFNSUc3TVFzd0NRWURWUVFHRXdKSlVqRVpNQmNHQTFVRUNnd1FUbTl1TFVkdmRtVnlibTFsYm5SaGJERWtNQ0lHQTFVRUN3d2JRVzF1WVdaNllYSWdSMjl6ZEdGeUxXVWdVMmhoY21sbUlFTnZNVFF3TWdZRFZRUUxEQ3RRWVhKelUybG5iaUJKYm5SbGNtMWxaR2xoZEdVZ1EyVnlkR2xtYVdOaGRHVWdRWFYwYUc5eWFYUjVNVFV3TXdZRFZRUUREQ3hRWVhKelUybG5iaUJRY21sMllYUmxJRWx1ZEdWeWJXVmthV0YwWlNCVGFXeDJaWElnUTBFZ0xTQkhNekFlRncweU16RXdNVGd3TmpReU1UWmFGdzB5TkRFd01USXdOalF5TVRaYU1JR3lNU0F3SGdZRFZRUUREQmRCYkdrZ1UyaHBjbTF2YUdGdFlXUnBJRnRUYVdkdVhURUxNQWtHQTFVRUJoTUNTVkl4RHpBTkJnTlZCQ29NQnRpNTJZVGJqREVUTUJFR0ExVUVCd3dLMktyWmg5aXgyS2ZaaGpFVk1CTUdBMVVFQ2d3TVZXNWhabVpwYkdsaGRHVmtNUk13RVFZRFZRUUZFd293TURJek5EVTNOekE0TVJNd0VRWURWUVFJREFyWXF0bUgyTEhZcDltR01Sb3dHQVlEVlFRRURCSFl0TnVNMkxFZzJZWFlyZG1GMksvYmpEQ0NBU0l3RFFZSktvWklodmNOQVFFQkJRQURnZ0VQQURDQ0FRb0NnZ0VCQUltN1lQSkxBYkl5SGZ0T0I5U3NNcHMxT2VGTUo1K2JSWkxhNGRFNWNmNDA0UWIySzZtK0lUeFQ4c0RPMDltRExxYmZ4VENxMStlaGxiQWwveXVnWGRGNStyR1lRWVgveWFKZit2a2FBcTE2MnZxZFJ2YlZ2a2hOcjFYSTJySStzWGZHSkJjR25jZXNrb1ZmN3JtK1JUY2lPbUFBM1poY2pkVTNqRm1VMS9BbThWQko4M2tLOFFmOEk2WnRGWmhITDFZcDA1MXhMVnM1UUk0L2x3bUt4a0VrbUcrVmdHVjIyTWt2ZTNZdzFQb2VNeWMwTzExMEJ2RmQzaWlFbkFVdEh1SUtGNmNGd1dpYTBoSUpMdmQrcy9XN0x1VlFNTlZZZXUrSjY0ekRxWUJ3RGFTdDRMK2VvZlQ2a3NnNmFzbFFTWmRmLzdWR3lER0lxY2pUZXQxRS9DOENBd0VBQWFPQjhEQ0I3VEFkQmdOVkhRNEVGZ1FVZXB2amFPS256bThONEMvK2cxRjRuZW9kemFNd0h3WURWUjBqQkJnd0ZvQVVGTWNPSUh5SjBSWi95cmxwNklmSkxFYUMrbFV3TkFZRFZSMGZCQzB3S3pBcG9DZWdKWVlqYUhSMGNEb3ZMMnd5TG5CaGNuTnphV2R1WTJFdWFYSXZVRk5mVERKRFFTNWpjbXd3RGdZRFZSMFBBUUgvQkFRREFnYkFNQllHQTFVZEpRRUIvd1FNTUFvR0NDc0dBUVVGQndNQ01EY0dDQ3NHQVFVRkJ3RUJCQ3N3S1RBbkJnZ3JCZ0VGQlFjd0FZWWJhSFIwY0Rvdkwyd3lMbkJoY25OemFXZHVZMkV1YVhJNk9ERXZNQlFHQTFVZElBUU5NQXN3Q1FZSFlJSnNaUUVCQWpBTkJna3Foa2lHOXcwQkFRc0ZBQU9DQVFFQWN3MmhkVjFZTjh4SFhzUXE0WDZZV3RneVVzSzkyQ1RJTGIvajJqWWZJR1hsd0xDWXhSNnpQRnVvcW1qVkc5VHhzM05FUFFpcnNWd3FvSVV6bGpUWm5CVS9XbFBqZXFQeVdhZFFvb1J2Zy82NEFoZDg4QThZSmYzWXZ2MURheDhyVUFOWCtBUWQwOTNPeEFyN2hmQ0lTVEhGbWhodzVHb0oxNTN2aDBpZGhla2VLNi8vaXd2THVnUEE5azBRdTBXT2JXc0oyQ3RqMnpXaENYTWR6SWFqcWtwRW15Qkp4ZTR5ajlaR3llempTbVBiK2toRGd2RXNick9QY1d5M1dUc3g4MFMzU2hwaGw4eXdOUVRSZDZPeGltRFhzMTQrbEwwQmxaYXY2bzBEcFRsa0Y0ME9tb1Brb1p3UWtKTnhCck5yS1dEZUszOXZSUjU2eVBQNHArMDRXdz09Il0sImFsZyI6IlJTMjU2In0.eyJoZWFkZXIiOnsidGF4aWQiOiJBMTEyMTYwNEMyMjAwMDJGMDk1MDExIiwiaW5ubyI6IjQ5MzIxMjE3IiwiaW5kYXRpbSI6MTY4Mzk5NzgzNzk4OCwiaW50eSI6MSwiaW5wIjoxLCJpbnMiOjEsInRpbnMiOiIxNDAwMzc3ODk5MCIsInRvYiI6MiwiYmlkIjoiMTAxMDAzMDI3NDYiLCJ0aW5iIjoiMTAxMDAzMDI3NDYiLCJ0cHJkaXMiOjIwMDAwLCJ0ZGlzIjo1MDAsInRhZGlzIjoxOTUwMCwidHZhbSI6MTc1NSwidG9kYW0iOjAsInRiaWxsIjoyMTI1NSwic2V0bSI6Mn0sImJvZHkiOlt7InNzdGlkIjoiMjcxMDAwMDEzODYyNCIsInNzdHQiOiLYs9ix2LPbjNmE2YbYr9ixINmC2LfYudin2Kog2LXZhti52Kog2YHZiNmE2KfYryDYs9in2LLbjCIsIm11IjoiMTY0IiwiYW0iOjIsImZlZSI6MTAwMDAsInByZGlzIjoyMDAwMCwiZGlzIjo1MDAsImFkaXMiOjE5NTAwLCJ2cmEiOjksInZhbSI6MTc1NSwidHNzdGFtIjoyMTI1NX1dLCJwYXltZW50cyI6W119.AVPw0w2IG7BJ5zglQYrP_vvBT1wX0eB4ZP77vvQkTN2At-zb8hpFeqG-uqui99cvR4i27GVnKcMpLBNVmMrL4FODTXqKn053bJ8oPJhbnUFhIkOAcoRmPUOtmO5jsWvocAL07Bh-m8B7nQG7d-RLEXViwbaCmMI7hzMCBzMm3P0Sr2cBVDrK8FUR_8VHb2l9gUm52A4Wy8i4moHc1Gi_65O3vrNX9bN3DiJSS_me5WOisE0GjglB3zHTMzE6MAZMpBbVJYfJOK-h-RKJaWLAlIvZYbYmhVMbfuxFB5Q9DlqDsvc0ie4Nn3Qs0sMVPcNO335ANSMu7wgFPqYmEK5aEw

توجه کنید که بررسی اعتبار امضا و گواهی امضا کننده‌ی صورتحساب نیز عینا همانند بررسی توکن JWS می‌باشد و در کنار بررسی گواهی امضای صادرکننده‌ی صورتحساب و تاریخ انقضای گواهی، از سرور OCSP و CRL صادرکننده‌ی گواهی استعلامات لازم برای اطمینان نسبت به اعتبار و صحت گواهی گرفته می‌شود.

لازم به ذکر است گواهی استفاده شده در فرآیند امضای صورتحساب می‌بایست معتبر بوده و کد ملی/شناسه ملی موجود در گواهی[۷] دسترسی ارسال صورتحساب برای شناسه یکتای حافظه مالیاتی مربوطه را داشته باشد.

۷- ۱- ۳- رمزگذاری صورتحساب

در مرحله‌ی بعد باید صورتحساب امضاشده را رمزگذاری کنیم. این کار باعث می‌شود هیچ شخص دیگری به جز سرور سامانه‌ی مودیان توانایی بازکردن و مشاهده محتویات صورتحساب را نداشته باشد. الگوریتم رمزگذاری مورد استفاده در این مرحله AES256-GCM (AES/GCM/NoPadding) می‌باشد و کلید AES نیز از طریق الگوریتم RSA-OAEP-256 (RSAES OAEP using SHA-256 and MGF1 with SHA-256) رمز می‌شود که این فرآیند بر اساس استاندارد JWE انجام می‌شود. الگوریتم رمزگذاری AES256-GCM دارای ۴ ورودی و ۲ خروجی است. ورودی‌های الگوریتم عبارتند از محتوای بسته یا همان payload، کلید متقارن، iv و داده اضافی احراز هویت AAD (Additional Authentication Data). خروجی‌های الگوریتم عبارتند از بسته‌ی رمزشده و تگ احراز هویت (Authentication tag). ابتدا یک کلید ۲۵۶ بیتی AES به همراه ۹۶ بیت iv (یا همان Initial Vector) تولید می‌شود. داده اولیه AAD هم از روی header ساخته می‌شود و در کنار محتوای بسته یا همان Payload (رشته‌ی JWS صورتحساب امضا شده) به عنوان ورودی به الگوریتم داده می‌شود تا داده رمز شده و Authentication Tag محاسبه شود. سپس خود کلید با الگوریتم RSA و به وسیله کلید عمومی سازمان رمز می‌شود. در نهایت صورتحساب رمزشده در کنار کلید AES رمز شده، شناسه کلید عمومی سازمان، iv و تگ شناسایی، یک بسته JWE را تشکیل می‌دهند که می‌توان آن را برای وب‌سرویس سامانه‌ی مودیان ارسال نمود.

---

[۷] فیلد SERIALNUMBER در قسمت SUBJECT گواهی امضاء

ساختار شیء JWE:

بسته‌ی رمز شده – JWE	
Header **سرآیند**	یک شیء Json دارای فیلدهای زیر: • alg: مقدار "RSA-OAEP-256" • enc: مقدار "A256GCM" • kid: شناسه کلید عمومی رمزگذاری سامانه مودیان نمونه: {"alg":"RSA-OAEP-256","enc":"A256GCM","kid":"6a2bcd88-a871-4245-a393-2843eafe6e02"}
Encrypted Symmetric Key **کلید متقارن رمز شده**	کلید متقارن رمزگذاری AES به طول ۳۲ بایت (۲۵۶ بیت) را با استفاده از الگوریتم RSA-OAEP-256 (RSAES OAEP using SHA-256 and MGF1 with SHA-256) با کلید عمومی رمزگذاری سامانه مودیان به طول ۴۰۹۶ بیت، رمز می‌کنیم. کلید رمز شده ۵۱۲ بایت طول دارد که Base64URL Encoded آن را در این قسمت قرار می‌دهیم.
Initial Vector	Encode شده‌ی ۱۲ بایت (۹۶ بیت) iv به فرمت Base64URL که رشته‌ای به طول ۱۶ کاراکتر می‌شود.
Payload **محتوای رمز شده**	رشته‌ی JWS صورتحساب امضا شده که با استفاده از کلید متقارن و با الگوریتم AES256-GCM رمز می‌شود.
Authentication Tag **تگ شناسایی**	تگی که به منظور اطمینان از دستکاری نشدن محتوای پیام رمز شده و کلیدها استفاده می‌شود.

برای شناخت بیشتر نسبت به ساختار JWE سند RFC7516 را مشاهده کند:

https://www.rfc- editor.org/rfc/rfc7516

مطابق این استاندارد قالب چیدمان بسته JWE به صورت زیر است:

```
BASE64URL(UTF8(JWE Protected Header))|| '.' ||
BASE64URL(JWE Encrypted Key)|| '.' ||
BASE64URL(JWE Initialization Vector)|| '.' ||
BASE64URL(JWE Ciphertext)|| '.' ||
BASE64URL(JWE Authentication Tag)
```

مراحل رمزگذاری صورتحساب:

سرآیند JWE به فرمت utf-8 برابر است با:


```
{"alg":"RSA-OAEP-256","enc":"A256GCM","kid":"6a2bcd88-a871-4245-a393-
2843eafe6e02"}
```


که Encode شده آن به فرمت Base64URL می‌شود:


```
eyJhbGciOiJSU0EtT0FFUC0yNTYiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiNmEyYmNkODgtYTg
3MS00MjQ1LWEzOTMtMjg0M2VhZmU2ZTAyIn0
```


توجه کنید که رشته‌ی سرآیند (header) حتما باید به فرمت utf-8 باشد.

حال یک کلید متقارن تولید می‌کنیم که Encode شده آن به فرمت Base64URL برابر است با:


```
_hm__fdwWlcK2FiFuuOHcRHqvKeYrd9_tzAdUI9IK5E
```


سپس این کلید را با کلید عمومی رمزگذاری سامانه مودیان به طول ۴۰۹۶ بیت، رمز می‌کنیم. این کلید از API دریافت اطلاعات سرور دریافت می‌شود و مقدار آن به فرمت Base64 برابر است با:


```
MIICIjANBgkqhkiG9w0BAQEFAAOCAg8AMIICCgKCAgEAxdzREOEfk3vBQogDPGTMqdDQ7t0oDhu
KMZkA+Wm1lhzjjhAGfSUOuDvOKRoUEQwP8oUcXRmYzcvCUgcfoRT5iz7HbovqH+bIeJwT4rmLmF
cbfPke+E3DLUxOtIZifEXrKXWgSVPkRnhMgym6UiAtnzwA1rmKstJoWpk9Nv34CYgTk8DKQN5jQ
Jqb9L/Ng0zOEEtI3zA424tsd9zv/kP4/SaSnbbnj0evqsZ29X6aBypvnTnwH9t3gbWM4I9eAVQh
PYClawHTqvdaz/O/feqfm06QBFnCgL+CBdjLs30xQSLsPICjnlV1jMzoTZnAabWP6FRzzj6C2sx
w9a/WwlXrKn3gldZ7Ctv6Jso72cEeCeUI1tzHMDJPU3Qy12RQzaXujpMhCz1DVa47RvqiumpTNy
K9HfFIdhgoupFkxT14XLDl65S55MF6HuQvo/RHSbBJ93FQ+2/x/Q2MNGB3BXOjNwM2pj3ojbDv3
pj9CHzvaYQUYM1yOcFmIJqJ72uvVf9Jx9iTObaNNF6pl52ADmh85GTAH1hz+4pR/E9IAXUIl/Yi
UneYu0G4tiDY4ZXykYNknNfhSgxmn/gPHT+7kL31nyxgjiEEhK0B0vagWvdRCNJSNGWpLtlq4Fl
CWTAnPI5ctiFgq925e+sySjNaORCoHraBXNEwyiHT2hu5ZipIW2cCAwEAAQ==
```


و حاصل رمز شده‌ی کلید متقارن با کلید عمومی به فرمت Base64URL سازمان می‌شود. قابل توجه است که خروجی رمز شده الگوریتم RSA-OAEP-۲۵۶ با ورودهای یکسان، متفاوت خواهد بود:


```
lRs_vcZlauJs4VwED3wZGr3wJDbg8It9zBStuJgqf7TiGs3kUCSulPwA6jRA-
8xuHTpNtKr1w0hsNbUhsx-
3BL5zGaDzsSekA1AZVevVy2qOrfId6OxLOb81SpVnSRR_K3jRHDWfH_ZDgZqMoM2mv6qFvb8GgG
FV59_jIkhctX73Y6iqe3NCask2CT0Ine3SUFVxt_OBqnBCwgDLrauGyisQbpVN1JJRjsWPeyXA0
0WvlECnRSLS93-prnMrfLTU_JHUyU304-cq7gVu7OATP1NB78L8m0-
MiIF_tr0mpbaEneOLDbimNYt2UVgn1thJsvh5Na30GKY5t-ISHoWhX-
DvVHp3KTEpp0bkC8IczZt-jg-
2vdYhyM7Sz1H0c9w2EvSSryYsAVonOojl4yWeaNhP_7mk09hgu1MUkr91Tfr-
U1k3zFHyCClfJzHj3t3uebS2mFDEyuao7K4omC335_VYsCi6kRSU421AW1_8V21RxaS2VDBQbgo
Cqr3UQQxsoSRIG3v1-OBgTgZ9slEGZnEihaptab-
7IeqnXIbhZNA8Onzy0wKW2S6r8M2eDd01QhGaDPGXcdWtOpC-Q-
```

cvu9MJTp4MRDx5dYe2rRQaI_A7WS5vJGkjwdRRpfhaJhGoXA7Wc6eUmB_UFOvPfsGQdFXPBUB3vh6jPfsER6hnXjU

مقدار IV تولید شده برای رمزگذاری محتوای اصلی با کلید AES به فرمت Base64URL برابر است با:

75ZoCO0VRqIWRyAs

همچنین برای ایجاد تگ شناسایی نیاز به داده اضافی احراز هویت یا همان AAD (Additional Authenticated Data) داریم که مقدار آن از طریق محاسبه مقدار ASCII سرآیند (Header) Encode شده به فرمت Base64URL به دست می‌آید:

Header: {"alg":"RSA-OAEP-256","enc":"A256GCM","kid":"6a2bcd88-a871-4245-a393-2843eafe6e02"}

Base64URL(Header):eyJhbGciOiJSU0EtT0FFUC0yNTYiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiNmEyYmNkODgtYTg3MS00MjQ1LWEzOTMtMjg0M2VhZmU2ZTAyIn0

AAD = ASCII(Base64URL(Header)): [101, 121, 74, 104, 98, 71, 99, 105, 79, 105, 74, 83, 85, 48, 69, 116, 84, 48, 70, 70, 85, 67, 48, 121, 78, 84, 89, 105, 76, 67, 74, 108, 98, 109, 77, 105, 79, 105, 74, 66, 77, 106, 85, 50, 82, 48, 78, 78, 73, 105, 119, 105, 97, 50, 108, 107, 73, 106, 111, 105, 78, 109, 69, 121, 89, 109, 78, 107, 79, 68, 103, 116, 89, 84, 103, 51, 77, 83, 48, 48, 77, 106, 81, 49, 76, 87, 69, 122, 79, 84, 77, 116, 77, 106, 103, 48, 77, 50, 86, 104, 90, 109, 85, 50, 90, 84, 64, 121, 73, 110, 48]

محتوای بسته که همان صورتحساب امضا شده است را با کلید متقارن تولید شده و الگوریتم AES256-GCM رمز می‌کنیم و حاصل به فرمت Base64URL برابر است با:

GiB15WaorEeK77gX1Tr-14lX0520tRcvrhp8D9cHCfYLTIonEo3MXYQ46Z8QTPv2ueCdiKztQDngTuczGGY3lve1cz-tHqeAJZ92UG9cjouMVE7IY29ML7OrzPy66GJsVb3BZnmIm6vDNEMf9AL8LH3y-H4rk9JmZjQbrRIU2AbmTGwRK0_FnQD660b-4nEMKfc51h9bk-dc8as3_OxFH2b0KxaiZDki8pDLumCFPQ9U9pfU7KUmkFlO9FGF3FpryGX5DzEE_YprotK6lcEr9f8U5ooHS_PF06pprnL_CkXtimdJEnJgghk-5XdZfdIMlR4ghYPDT_rfE0GjmroEaWOaoM3qXwDhdUFs1nb_48KAUOQY1ebgd9Ngi_ed-rh5l7TAj1wPCmqnPEY9SYIPeorurSwlMPPpI9NN-bjjBUhW1a4RxkCt5UeHDS--bvRkPqD5DaiOQg2RD7n8cm6pzHKcKO16WpXSSFhIjK1-l_kevoprXPIEHtWik3PBSXYWDzl5Gc_ET-aw2CF6xgWkUi-vw512_ySvs66grxGqubBQkOOxLLyUEmcxedPSCSs6g4eNqf-SES0yZ9NR4uBs3WBsqs2h0Ftrn5ZKniMDQKtoU0LJvscjXBso6ckAVuvjBilY4zKgr07FjJ9zn-9xkStotxms0czBo-fOsVunTuKypLBvTiwV4uykqUnsxbyOtuUvr10kLK1dma4F9q1gixRqfoMyn1o7seRDTRrTkBh_DkrF-AzNhZG3o1TC5bHUtwN5crc2dw3FkS4UMe1unQkusFL44MRqY3qbn_iP4NY8X3bWTWtp3lkiFxKtATTiZG_lvi5om1QwcA7a_mYQw47EjjvGTiSJQsfDM8d2WBlVDOV65xgcqy7OQVRPxfTo6jJu-X1MGd-RN2Lb_crLsnJGJ6kpdhqyF7ts_3gbWvryzKLJyDe2HM3izN6Kw0U4Ba-__YOMPV-mJECsVC5KUw9jRYhw6tdYSU41LXKRDnMyh-

cjRyuv4V347DOR280aGWsbqpqaDp4mdtkaABARhtO5BhbzzeSnHnEjTVZmdn8H5WifzGvqHCiGA
z752CNNO00OwCbpVoW2Jcq78XndubsxsXzkcq0XvsDxnKzdBdRkoYyTEd5tMOdxUZw_tpbXNQZF
FwAB4ph_FtoYC0j5OkGhEola6h7jguoFJu0CewloGSkvvqu2Ht4G9IfDkFs0tMTvdjHJJGWpDj9
nQ_Mzn0QkKBPhGLM50yUtosIVd1FvrHCJuhkt3qsAP6OTHvgDswELnLMcOsoRB2qdVp6YSAOT3Z
uRc6Bawwd5S78wd_o4ISW9pNMxNbHPueiXI_N48S9Zcb2ktwH6dB3rbkRwgXCEWcPheX8s5ik32
5PbY5RAs29fj45yISSZ-
6xlg9BwBXjg81TkNg18_6j0aI8ZZmDjL1EyjK0ojb9UpQUfjreakNAGhxSa5S_ethmI63Bs6_En
XyzOKDP5sPmlyBgJ6RjXRSmVI8zqA1ETazPY93PYql_PqADQm48_Ocs1h_djjNNBHeA8ucYelX1
T8tcD-ENwrvJw8_2u1A1DWX-
FucPGQQLZc4PkwO4OCUeBlxDUsJFzedhX0cKEQr_b9zxhVT3C09yr3q6FyhqrYF892Kql06PmXB
VEZrb-Q0piVZyozeordNJGRE0l6r8Al0hnc-KgUlkdZrxee_zzk9aV08CUOl-
fvBRCGOR12v_EAcY17pnyfnCKmZH3u3Iq5D7-
T3JHJSZz2NuUFrCok81L87X0YbRCscW164VAEsF1QolakktKt8cNRQwj_95rtt1Ka06Fdg6Q-
VK9NskncKRKnlWcIQT6Ec6eTmr1Ng7JaQ7dqkEh_zThM8ifE-
Ak6jVdpMiHZ4NvdlOndCPbi1r7zFNrUeRI0BK5PWfOJA90YAnMJ3B7DkrAMyLhZMkKXp4UjEZSQ
hBhe_lnD9VxL3rZcmRqEctFVmsTnyRZY-
pZlSx8gS__99WXrtNqD1lR3njeHWJsCi35GoJJwWh3BFCJvsyeQcmsIkkN4RdBrwj_B-
ZpUKJn1gvho2E7PPkvuMpMPSWr1c1zptCpzamAY2pTjBgmQ5NWDnxN87fUZN9YaJYMcqD7kb-
kooJuNWw7TcF-YPo_31UHHWd3mKzBVFUeYEqqJBA-
9KroMQzfC7hc5QQkaA_oGVn0_8_AexX7fNDdXc6yRzM_msfGsbO6pMYegbEijS5F3DaopvpAmwd
BbFlDRUIGSKpfODn9DNGB2BoqNn1mNPCu_BsqGuPvF3RzgxP-1MoCI3Xf7RxG1DYr-
bTPsV6pyzinV5HaGSNRorYkXchaXhl19k4d9pfu_4uXeASFVG6WxzdpVwb_w769VQdVYL7KG0u4
uC_MhWDH0w1DVEFySffBNPpZ6Z_lLYkbc8epocBH9HjJVFByP4gqVieVuDlX_tkK6QUwSPZTl36
Be4uVf9luAJXZfn3IkMbj8KuRfLjwBE3N4MnjXbz4RHN5dZwaSJftRgYo_DaJPiqcGz1H3dB5B4
i0A7bLA4H1TEIKLPom-
A1GnpRruGdoF7dD3dhx8HOKUt7ny7zMrYQWI86bmRzubnMZmi87GguvCoCP7o7ATMJIGprSDmg-
3TJ5FGyeXaNRZOlEP1Dhc6uh19Pm_PYP7l8tKg5Z_dGgnFWwJp2SDweU7kvWgEUePJVa-
HKd7iiRGP-
2cttS8ak4zlq_OjKkFRHi6A2uZ95Yg0dkWgBbHgCHbbuM5oBmIztFixmnHeoTrV3HPbBPw2jEX_
XINhFVcYnfc1efs0Gh-
jKA4wrCgStq5bXRhMC5degaoQ0Sb5eY_S_Ehpz1yoA_JYDHp6t8FfnSMTNbM9LfoBTgZDdlrRmw
d2dBVJxzkbhTpXo4dpShFRsL1vs0GV3wdELBuzMb3UaTyu-
rJy_DOblfPE8pmlC1NS2_WHrVXB0gJf91MaaIup_Fx7tJzSbqiNG9mFj6SQvMk0HvAyI195NpyF
NX33x5NLFnF6l25mZdn6MY6nLPeNu9P1yEadZ8uXnMtwfrAOuN83yS2XZDWKHPYSJOaVcsGAjRr
cGLpulp-dcx1g0XVQgNR0KzaH7J3zbRJ-Lqf7V6w9aYT1JnHk21bof2oGp9NaWN1-
TqlOEvrAk3-
EqzyCigOG5A2YZdoRR_1rlCxfNCXz0dtRwM07IX_kEEEKHTs4lk970YTSa_SdhYNMUN0QPBLkjT
ts_YMnx8Y2wFumFy9Ji3K5o2GHB5nc0a1DioSi9gNEpFjv6pT_Tk9NDIFUBAwv8oV_qp41qFqSp
A9g6G51madn0K6sonvAsRlDWNjxSBropg1stgWWCRe3Ucp_b9PhZht1rSuJsZynEqLCf9ZLn2lx
fugLbIOdvaNZxkI2NtNHCv_CsiMFo76rC5SwCJ5uTtD64z53s2txbOPmhWpIGE63OKi3fQSZz5H
ZdBbidIRcM3vCEseMNw1A27F7_BCIN6o9tZd-lX0Zkvh08Wts2d-c1Fs-
x5ssLWeoMPpGvnFjZ9pQBD8EyK34JsA5qPTVjLOD62ZRph8PdkiJIoE0es4QMyleVbUe9rbscIN
F2m5YSHiRWWM5MFSpDFmT67EtriseeWQnVNQTU31_chDRut92yv2G3seSa8HHMLW0Gmq7W7hUUG
uxLbUSVv74U7_55vpQM0fuRhJ7khTbZioN-
wBp32Zyy_phgMi5sOqV35lPoCzpZ7ueQwbsS6kvABCj2PjoeQL2trqrjcfkPfLPUKsETgkVrgvi
jt55bzKmUG624fv1zJbgSt7t3rrPiVQyKflW0YJcx-
39sKDICQeEFpQp_7gnf7LoKKVPAISDLSSCHJro6R5Vm74INsQQjGn-
fH55R2sS4GVw74DfWF1k7wp9Upw3JjlgQ9MhRpVjUd98CprL-
jjdn4k4CuHxZPXEoXEfkvC8PO4HmDN1Od-
3ShVrilhJ1gLFpuM18l4OylUju9bhHEM8R28YL7ftgWvtzhMkdw3BPM05Uzo29xAsS_GZeDikFl

db2E7CoZu3evhnrMsk9KWGbbrxcAZDFR1zhNJHJBFVPpjpxgoXy6Br3LAtZBnu2l7Zt5mfAF08h
EG0baz-MS0hPahcfoE3S7c5xPcV4wTmmX_r9pE_eg-
6tyiylj5I9hOpgUMtogAgRSNA8VP5YLvoDv5D3eMhK_ZzTQtQGEQulpA86uPKjImusrAe4XAbFE
fWIPZ--S6xkYacgCzGjqfAQOInmRLg_fOcmuy2atMbCX9Rf66rbGhOTwIY-HEBUAeIA9cY_AL-
rb78ErzcKhRodnAN3XXq-rpA9709UjBiFm_SEPGEsA6HlmxDoPFubEt1g-
36CxG41OLaKSEAXPUm105xNWUHS_xOaT55mq4DgaX_3cn8q6za9729x3PiGwtw0LhlhH-
sezEb7wkSW0HjJoXkRKAMKD_HcaV7cAcwiMEvvE3sX_JQ

همچنین خروجی Authentication Tag عملیات رمزگذاری بسته با iv و aad داده شده به فرمت Base64URL برابر است با:

m0uVEEEqSLHJRbG-xLOsyA

از کنار هم گذاشتن این مقادیر به ترتیب header، کلید متقارن رمز شده با کلید سازمان، iv، بسته‌ی رمز شده با کلید متقارن تولید شده و تگ شناسایی بسته‌ی JWE ساخته می‌شود:

eyJhbGciOiJSU0EtT0FFUC0yNTYiLCJlbmMiOiJBMjU2R0NNIiwia2lkIjoiNmEyYmNkO
DgtYTg3MS00MjQ1LWEzOTMtMjg0M2VhZmU2ZTAyIn0.lRs_vcZlauJs4VwED3wZGr3wJDbg8It9
zBStuJgqf7TiGs3kUCSulPwA6jRA-8xuHTpNtKr1w0hsNbUhsx-
3BL5zGaDzsSekA1AZVevVy2qOrfId6OxLOb81SpVnSRR_K3jRHDWfH_ZDgZqMoM2mv6qFvb8GgG
FV59_jIkhctX73Y6iqe3NCask2CT0Ine3SUFVxt_OBqnBCwgDLrauGyisQbpVN1JJRjsWPeyXA0
0WvlECnRSLS93-prnMrfLTU_JHUyU304-cq7gVu7OATP1NB78L8m0-
MiIF_tr0mpbaEneOLDbimNYt2UVgn1thJsvh5Na30GKY5t-ISHoWhX-
DvVHp3KTEpp0bkC8IczZt-jg-
2vdYhyM7Sz1H0c9w2EvSSryYsAVonOojl4yWeaNhP_7mk09hgu1MUkr91Tfr-
U1k3zFHyCClfJzHj3t3uebS2mFDEyuao7K4omC335_VYsCi6kRSU421AW1_8V21RxaS2VDBQbgo
Cqr3UQQxsoSRIG3v1-OBgTgZ9slEGZnEihaptab-
7IeqnXIbhZNA8Onzy0wKW2S6r8M2eDd01QhGaDPGXcdWtOpC-Q-
cvu9MJTp4MRDx5dYe2rRQaI_A7WS5vJGkjwdRRpfhaJhGoXA7Wc6eUmB_UFOvPfsGQdFXPBUB3v
h6jPfsER6hnXjU.75ZoCO0VRqIWRyAs.GiB15WaorEeK77gX1Tr-
14lX0520tRcvrhp8D9cHCfYLTIonEo3MXYQ46Z8QTPv2ueCdiKztQDngTuczGGY3lve1cz-
tHqeAJZ92UG9cjouMVE7IY29ML7OrzPy66GJsVb3BZnmIm6vDNEMf9AL8LH3y-
H4rk9JmZjQbrRIU2AbmTGwRK0_FnQD660b-4nEMKfc51h9bk-
dc8as3_OxFH2b0KxaiZDki8pDLumCFPQ9U9pfU7KUmkFlO9FGF3FpryGX5DzEE_YprotK6lcEr9
f8U5ooHS_PF06pprnL_CkXtimdJEnJgghk-
5XdZfdIMlR4ghYPDT_rfE0GjmroEaWOaoM3qXwDhdUFs1nb_48KAUOQY1ebgd9Ngi_ed-
rh5l7TAj1wPCmqnPEY9SYIPeorurSwlMPPpI9NN-bjjBUhW1a4RxkCt5UeHDS--
bvRkPqD5DaiOQg2RD7n8cm6pzHKcKO16WpXSSFhIjK1-
l_kevoprXPIEHtWik3PBSXYWDzl5Gc_ET-aw2CF6xgWkUi-
vw512_ySvs66grxGqubBQkOOxLLyUEmcxedPSCSs6g4eNqf-
SES0yZ9NR4uBs3WBsqs2h0Ftrn5ZKniMDQKtoU0LJvscjXBso6ckAVuvjBilY4zKgr07FjJ9zn-
9xkStotxms0czBo-
fOsVunTuKypLBvTiwV4uykqUnsxbyOtuUvr10kLK1dma4F9q1gixRqfoMyn1o7seRDTRrTkBh_D
krF-
AzNhZG3o1TC5bHUtwN5crc2dw3FkS4UMe1unQkusFL44MRqY3qbn_iP4NY8X3bWTWtp3lkiFxKt
ATTiZG_lvi5om1QwcA7a_mYQw47EjjvGTiSJQsfDM8d2WBlVDOV65xgcqy7OQVRPxfTo6jJu-
X1MGd-RN2Lb_crLsnJGJ6kpdhqyF7ts_3gbWvryzKLJyDe2HM3izN6Kw0U4Ba-__YOMPV-
mJECsVC5KUw9jRYhw6tdYSU41LXKRDnMyh-
cjRyuv4V347DOR280aGWsbqpqaDp4mdtkaABARhtO5BhbzzeSnHnEjTVZmdn8H5WifzGvqHCiGA
z752CNNO00OwCbpVoW2Jcq78XndubsxsXzkcq0XvsDxnKzdBdRkoYyTEd5tMOdxUZw_tpbXNQZF

FwAB4ph_FtoYC0j5OkGhEola6h7jguoFJu0CewloGSkvvqu2Ht4G9IfDkFs0tMTvdjHJJGWpDj9
nQ_Mzn0QkKBPhGLM50yUtosIVd1FvrHCJuhkt3qsAP6OTHvgDswELnLMcOsoRB2qdVp6YSAOT3Z
uRc6Bawwd5S78wd_o4ISW9pNMxNbHPueiXI_N48S9Zcb2ktwH6dB3rbkRwgXCEWcPheX8s5ik32
5PbY5RAs29fj45yISSZ-
6xlg9BwBXjg81TkNg18_6j0aI8ZZmDjL1EyjK0ojb9UpQUfjreakNAGhxSa5S_ethmI63Bs6_En
XyzOKDP5sPmlyBgJ6RjXRSmVI8zqA1ETazPY93PYql_PqADQm48_Ocs1h_djjNNBHeA8ucYelX1
T8tcD-ENwrvJw8_2u1A1DWX-
FucPGQQLZc4PkwO4OCUeBlxDUsJFzedhX0cKEQr_b9zxhVT3C09yr3q6FyhqrYF892Kql06PmXB
VEZrb-Q0piVZyozeordNJGRE0l6r8Al0hnc-KgUlkdZrxee_zzk9aV08CUOl-
fvBRCGOR12v_EAcY17pnyfnCKmZH3u3Iq5D7-
T3JHJSZz2NuUFrCok81L87X0YbRCscW164VAEsF1QolakktKt8cNRQwj_95rtt1Ka06Fdg6Q-
VK9NskncKRKnlWcIQT6Ec6eTmr1Ng7JaQ7dqkEh_zThM8ifE-
Ak6jVdpMiHZ4NvdlOndCPbi1r7zFNrUeRI0BK5PWfOJA90YAnMJ3B7DkrAMyLhZMkKXp4UjEZSQ
hBhe_lnD9VxL3rZcmRqEctFVmsTnyRZY-
pZlSx8gS__99WXrtNqD1lR3njeHWJsCi35GoJJwWh3BFCJvsyeQcmsIkkN4RdBrwj_B-
ZpUKJn1gvho2E7PPkvuMpMPSWr1c1zptCpzamAY2pTjBgmQ5NWDnxN87fUZN9YaJYMcqD7kb-
kooJuNWw7TcF-YPo_31UHHWd3mKzBVFUeYEqqJBA-
9KroMQzfC7hc5QQkaA_oGVn0_8_AexX7fNDdXc6yRzM_msfGsbO6pMYegbEijS5F3DaopvpAmwd
BbFlDRUIGSKpfODn9DNGB2BoqNn1mNPCu_BsqGuPvF3RzgxP-1MoCI3Xf7RxG1DYr-
bTPsV6pyzinV5HaGSNRorYkXchaXhl19k4d9pfu_4uXeASFVG6WxzdpVwb_w769VQdVYL7KG0u4
uC_MhWDH0w1DVEFySffBNPpZ6Z_lLYkbc8epocBH9HjJVFByP4gqVieVuDlX_tkK6QUwSPZTl36
Be4uVf9luAJXZfn3IkMbj8KuRfLjwBE3N4MnjXbz4RHN5dZwaSJftRgYo_DaJPiqcGz1H3dB5B4
i0A7bLA4H1TEIKLPom-
A1GnpRruGdoF7dD3dhx8HOKUt7ny7zMrYQWI86bmRzubnMZmi87GguvCoCP7o7ATMJIGprSDmg-
3TJ5FGyeXaNRZOlEP1Dhc6uh19Pm_PYP7l8tKg5Z_dGgnFWwJp2SDweU7kvWgEUePJVa-
HKd7iiRGP-
2cttS8ak4zlq_OjKkFRHi6A2uZ95Yg0dkWgBbHgCHbbuM5oBmIztFixmnHeoTrV3HPbBPw2jEX_
XINhFVcYnfc1efs0Gh-
jKA4wrCgStq5bXRhMC5degaoQ0Sb5eY_S_Ehpz1yoA_JYDHp6t8FfnSMTNbM9LfoBTgZDdlrRmw
d2dBVJxzkbhTpXo4dpShFRsL1vs0GV3wdELBuzMb3UaTyu-
rJy_DOblfPE8pmlC1NS2_WHrVXB0gJf91MaaIup_Fx7tJzSbqiNG9mFj6SQvMk0HvAyI195NpyF
NX33x5NLFnF6l25mZdn6MY6nLPeNu9P1yEadZ8uXnMtwfrAOuN83yS2XZDWKHPYSJOaVcsGAjRr
cGLpulp-dcx1g0XVQgNR0KzaH7J3zbRJ-Lqf7V6w9aYT1JnHk21bof2oGp9NaWN1-
TqlOEvrAk3-
EqzyCigOG5A2YZdoRR_1rlCxfNCXz0dtRwM07IX_kEEEKHTs4lk970YTSa_SdhYNMUN0QPBLkjT
ts_YMnx8Y2wFumFy9Ji3K5o2GHB5nc0a1DioSi9gNEpFjv6pT_Tk9NDIFUBAwv8oV_qp41qFqSp
A9g6G51madn0K6sonvAsRlDWNjxSBropg1stgWWCRe3Ucp_b9PhZht1rSuJsZynEqLCf9ZLn2lx
fugLbIOdvaNZxkI2NtNHCv_CsiMFo76rC5SwCJ5uTtD64z53s2txbOPmhWpIGE63OKi3fQSZz5H
ZdBbidIRcM3vCEseMNw1A27F7_BCIN6o9tZd-lX0Zkvh08Wts2d-c1Fs-
x5ssLWeoMPpGvnFjZ9pQBD8EyK34JsA5qPTVjLOD62ZRph8PdkiJIoE0es4QMyleVbUe9rbscIN
F2m5YSHiRWWM5MFSpDFmT67EtriseeWQnVNQTU31_chDRut92yv2G3seSa8HHMLW0Gmq7W7hUUG
uxLbUSVv74U7_55vpQM0fuRhJ7khTbZioN-
wBp32Zyy_phgMi5sOqV35lPoCzpZ7ueQwbsS6kvABCj2PjoeQL2trqrjcfkPfLPUKsETgkVrgvi
jt55bzKmUG624fv1zJbgSt7t3rrPiVQyKflW0YJcx-
39sKDICQeEFpQp_7gnf7LoKKVPAISDLSSCHJro6R5Vm74INsQQjGn-
fH55R2sS4GVw74DfWF1k7wp9Upw3JjlgQ9MhRpVjUd98CprL-
jjdn4k4CuHxZPXEoXEfkvC8PO4HmDN1Od-
3ShVrilhJ1gLFpuM18l4OylUju9bhHEM8R28YL7ftgWvtzhMkdw3BPM05Uzo29xAsS_GZeDikFl
db2E7CoZu3evhnrMsk9KWGbbrxcAZDFR1zhNJHJBFVPpjpxgoXy6Br3LAtZBnu2l7Zt5mfAF08h
EG0baz-MS0hPahcfoE3S7c5xPcV4wTmmX_r9pE_eg-
6tyiylj5I9hOpgUMtogAgRSNA8VP5YLvoDv5D3eMhK_ZzTQtQGEQulpA86uPKjImusrAe4XAbFE
fWIPZ--S6xkYacgCzGjqfAQOInmRLg_fOcmuy2atMbCX9Rf66rbGhOTwIY-HEBUAeIA9cY_AL-
rb78ErzcKhRodnAN3XXq-rpA9709UjBiFm_SEPGEsA6HlmxDoPFubEt1g-

36CxG41OLaKSEAXPUm105xNWUHS_xOaT55mq4DgaX_3cn8q6za9729x3PiGwtw0LhlhH-sezEb7wkSW0HjJoXkRKAMKD_HcaV7cAcwiMEvvE3sX_JQ.m0uVEEEqSLHJRbG-xLOsyA

تکه کد زیر به زبان جاوا با در اختیار داشتن فایل کلید خصوصی امضا و گواهی امضای مودی و همچنین کلید عمومی سرور و شناسه‌ی آن کلید، از روی صورتحساب داده شده، بسته‌ی امضاشده و رمزشده را تولید می‌نماید:

```
/** Loading Signature Private Key in PKCS#8 format **/
String privateKeyPath = "path/to/private-key.pem";
KeyFactory keyFactory = KeyFactory.getInstance("RSA");
final PEMParser pemParser = new PEMParser(new FileReader(privateKeyPath));
final PrivateKeyInfo pemKeyPair = (PrivateKeyInfo) pemParser.readObject();
final byte[] encoded = pemKeyPair.getEncoded();
PrivateKey privateKey = keyFactory.generatePrivate(new
PKCS8EncodedKeySpec(encoded));

/** Loading Certificate **/
String certificatePath = "path/to/certificate.crt";
CertificateFactory certificateFactory =
CertificateFactory.getInstance("X.509");
X509Certificate certificate = (X509Certificate)
certificateFactory.generateCertificate(new
FileInputStream(certificatePath));

/** Loading server Key - Taken from server-information **/
String serverPublicKeyString = "...[Server public key in Base64
Format]...";
String serverPublicKeyId = "...[Server Encryption Key Id]...";
final byte[] byteKey = Base64.getDecoder().decode(serverPublicKeyString);
final X509EncodedKeySpec X509publicKey = new X509EncodedKeySpec(byteKey);
final KeyFactory kf = KeyFactory.getInstance("RSA");
PublicKey publicKey = kf.generatePublic(X509publicKey);

String signatureTime = "2023-05-13T15:32:01Z";

String invoiceJson = "{ INVOICE JSON }";

/** Sign Invoice **/
final JsonWebSignature jws = new JsonWebSignature();

jws.setPayload(invoiceJson);
jws.setAlgorithmHeaderValue(AlgorithmIdentifiers.RSA_USING_SHA256);
jws.setKey(privateKey);
jws.setCertificateChainHeaderValue(certificate);
jws.setHeader("sigT", signatureTime);
jws.setHeader("crit", new String[]{"sigT"});

jws.sign();

String signedJson = jws.getCompactSerialization();
```

```
/** Encrypt Signed Json **/
final JsonWebEncryption jwe = new JsonWebEncryption();

jwe.setAlgorithmHeaderValue(KeyManagementAlgorithmIdentifiers.RSA_OAEP_256)
;
jwe.setEncryptionMethodHeaderParameter(ContentEncryptionAlgorithmIdentifier
s.AES_256_GCM);
jwe.setPayload(signedJson);
jwe.setKey(publicKey);
jwe.setKeyIdHeaderValue(serverPublicKeyId);

String encryptedInvoice = jwe.getCompactSerialization();
System.out.println(encryptedInvoice);
```

توجه کنید که این موارد به طور کامل در SDK پیاده‌سازی شده‌اند و تنها با فراخوانی یک تابع می‌توان از آن استفاده نمود. تکه کد ارائه شده صرفا به منظور آشنایی دقیق‌تر با فرآیند تولید بسته‌ی امضاشده و رمزشده‌ی صورتحساب می‌باشد. در این بخش هم برای عملیات تولید JWS و JWE از کتابخانه jose-4j نسخه‌ی ۰.۹.۳ استفاده شده است. همچنین تکه کد زیر به زبان NET. همین عملیات را پیاده سازی می‌کند:

```
namespace TaxCollectData.Sample;

using System.Globalization;
using System.Security.Cryptography;
using System.Text.Json;
using System.Text.Json.Nodes;
using JWT.Algorithms;
using JWT.Builder;
using Org.BouncyCastle.Crypto.Parameters;
using Org.BouncyCastle.OpenSsl;
using Org.BouncyCastle.Security;
using Jose;
using Org.BouncyCastle.Utilities.Encoders;

internal class InvoiceSignAndEncryptionTest
{
    public static void Main(string[] args)
    {
        // Loading Signature Private Key in PKCS#8 format
        var privateKeyPemReader = new
PemReader(File.OpenText("path/to/private-key.pem"));
        var privateKey = DotNetUtilities.ToRSA((RsaPrivateCrtKeyParameters)
privateKeyPemReader.ReadObject());

        // Loading Certificate
        var certPemReader = new PemReader(new
StreamReader("path/to/certificate.crt"));
        var certificate =
DotNetUtilities.ToX509Certificate((Org.BouncyCastle.X509.X509Certificate)
certPemReader.ReadObject());
        var publicKey = DotNetUtilities.ToRSA((RsaKeyParameters)
DotNetUtilities.FromX509Certificate(certificate).GetPublicKey());
```

```
        var invoiceJson = "{ INVOICE JSON }";

        // Generating JWS
        var signedJson = JwtBuilder.Create()
            .WithAlgorithm(new RS256Algorithm(publicKey, privateKey))
            .AddHeader(HeaderName.X5c, new[]
{Convert.ToBase64String(certificate.GetRawCertData())})
            .AddHeader("sigT", DateTime.UtcNow.ToString("yyyy-MM-
dd'T'HH:mm:ss'Z'", CultureInfo.InvariantCulture))
            .AddHeader("crit", new[] {"sigT"})
            .Encode(JsonSerializer.Deserialize<JsonNode>(invoiceJson));

        // Encrypt jws
        var serverPublicKeyString = "...[Server public key in Base64
Format]...";
        var serverPublicKeyId = "...[Server Encryption Key Id]...";
        var decoded = Base64.Decode(serverPublicKeyString);
        var asymmetricKeyParameter = PublicKeyFactory.CreateKey(decoded);
        var rsaParams = DotNetUtilities.ToRSAParameters((RsaKeyParameters)
asymmetricKeyParameter);
        var rsa = RSA.Create();
        rsa.ImportParameters(rsaParams);
        var header = new Dictionary<string, object>
        {
            {
                "kid", serverPublicKeyId
            }
        };
        var recipient = new JweRecipient(JweAlgorithm.RSA_OAEP_256, rsa,
header);
        var encryptedInvoice = JWE.Encrypt(signedJson, new[] {recipient},
JweEncryption.A256GCM, mode: SerializationMode.Compact);

        Console.WriteLine(encryptedInvoice);
    }
}
```

این کد نیز از همان پکیج‌های قبلی برای امضا و رمزنگاری صورتحساب استفاده می‌نماید:

```
<PackageReference Include="jose-jwt" Version="4.1.0" />
<PackageReference Include="JWT" Version="10.0.2" />
<PackageReference Include="System.Security.Cryptography.Cng"
Version="6.0.0-preview.4.21253.7" />
<PackageReference Include="System.Security.Cryptography.X509Certificates"
Version="4.3.2" />
<PackageReference Include="Portable.BouncyCastle" Version="1.9.0" />
```

۷-۱-۴- ارسال صورتحساب

ارسال صورتحساب از طریق فراخوانی این API به شیوه‌ی POST امکان پذیر است و از طریق آن می‌توان لیستی از صورتحساب‌ها را با هم ارسال نمود. نحوه‌ی کار این API به صورت زیر است:

Send Invoices – ارسال صورتحساب	
آدرس	https://tp.tax.gov.ir/requestsmanager/api/v2/invoice
Method	POST
ورودی	لیستی از InvoicePacketها هر یک شامل موارد زیر: • header: آبجکت Json ○ requestTraceId: شناسه منحصر به فرد درخواست از نوع uuid ○ fiscalId: از نوع رشته، شناسه حافظه صادرکننده‌ی صورتحساب • payload: از نوع رشته، بسته‌ی JWE صورتحساب (امضا شده و رمز شده) محل قرارگیری: Request Body در یک درخواست ارسال صورتحساب امکان ارسال حداکثر ۱۰۰۰ صورتحساب وجود دارد.
خروجی	آبجکت Json دارای فیلدهای زیر: • timestamp: برچسب زمانی لحظه ایجاد جواب • result: آرایه‌ی Json هر یک شامل شیئی که دارای فیلدهای زیر است: ○ uid: همان requestTraceId ارسال شده به ازای هر صورتحساب ○ packetType: در اینجا `null` است. ○ referenceNumber: شماره پیگیری صورتحساب ارسالی ○ data: در اینجا `null` است.

با استفاده از این API شرکت معتمد می‌تواند همه‌ی صورتحساب‌های مودیان مختلفی که دسترسی ارسال صورتحساب را به شرکت معتمد واگذار کرده‌اند ارسال نماید و نیازی نیست همه‌ی صورتحساب‌های موجود در یک درخواست متعلق به یک شناسه یکتای حافظه مالیاتی باشد. همچنین مودی می‌تواند در یک درخواست ارسال صورتحساب، صورتحساب‌های مربوط به شناسه یکتاهای مختلف خود را ارسال نماید. البته شناسه یکتاهایی که اجازه‌ی ارسال صورتحساب از طریق آن‌ها را به شرکت معتمد واگذار نکرده باشد.

همچنین توجه نمایید که مقدار فیلد requestTraceId برای هر صورتحساب در یک درخواست باید منحصر به فرد باشد و درصورتی که در یک درخواست دو صورتحساب دارای requestTraceId یکسان باشند درخواست در همان ابتدا با خطا مواجه می‌شود. همچنین در صورتی که ارسال کننده درخواست برای یک شناسه

یکتا برای دو صورتحساب از requestTraceIdهای تکراری استفاده نماید هنگام استعلام وضعیت صورتحساب با uid تنها پاسخ وضعیت یکی از آن‌ها به عنوان جواب برگردانده می‌شود.

نمونه درخواست cURL ارسال صورتحساب و پاسخ آن:

```
curl -X 'POST' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/invoice' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGc [ JWT Token ] dLcdPeI_9Q' \
  -H 'Content-Type: application/json' \
  -d '[
  {
    "payload": "eyJhbGciOiJ...[ JWE ]...EEzZe9mxIxw",
    "header": {
      "requestTraceId": "cf019c26-f235-11ed-a05b-0242ac120003",
      "fiscalId": "A11216"
    }
  }
]'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
{
  "timestamp": 1684054900556,
  "result": [
    {
      "uid": "cf019c26-f235-11ed-a05b-0242ac120003",
      "packetType": null,
      "referenceNumber": "3645b684-2c1e-400c-8584-f739c09d99fb",
      "data": null
    }
  ]
}
```

۸ – استعلام وضعیت صورتحساب‌های ارسالی

پس از تولید یک بسته صورتحساب امضاشده و رمزشده و ارسال آن از طریق API به سرور، وب سرویس جمع‌آوری به شما یک شماره پیگیری می‌دهد و از طریق آن (یا از طریق شناسه درخواست تولیدشده توسط خودتان) و یا از طریق زمان ارسال صورتحساب می‌توانید نسبت به وضعیت پردازش صورتحساب و خطاهای احتمالی موجود در آن مطلع شوید. جزئیات خطاهای وب سرویس جمع‌آوری و لایه محتوا، در سند کدخطاهای سامانه مودیان آورده شده است. به عنوان مثال فرض کنید خروجی درخواست ارسال دو صورتحساب به شکل زیر می‌باشد:

```
{
  "timestamp": 1684073047333,
  "result": [
    {
      "uid": "c5352f85-9322-41bc-a5b2-9abb130fe622",
      "packetType": null,
      "referenceNumber": "f9173085-2316-4ca6-918e-e41aaf7ef8dd",
      "data": null
    },
    {
      "uid": "2b982bfd-9a60-47cd-9da7-30fc0dabd37d",
      "packetType": null,
      "referenceNumber": "93367b02-23dd-4568-90e1-2b47d799f361",
      "data": null
    }
  ]
}
```


برای استعلام وضعیت صورتحساب سه روش وجود دارد که در ادامه شرح داده می‌شوند.

۸– ۱– استعلام صورتحساب به وسیله شماره پیگیری

به وسیله‌ی این API می‌توان وضعیت صورتحساب ارسالی را با استفاده از کد پیگیری‌ای که وب سرویس جمع آوری به ازای هر صورتحساب در اختیار قرار می‌دهد استعلام نمائیم. نحوه‌ی کار این API به صورت زیر است:

استعلام به وسیله شماره پیگیری – Inquiry by Reference Id	
https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-reference-id	**آدرس**
GET	**Method**

فیلدهای زیر:
• referenceIds: لیست شماره‌ی پیگیری‌هایی که قرار است استعلام شوند.
• start: شروع بازه زمانی که در آن به دنبال صورتحساب هستیم. فرمت:
`2023-05-14T00:00:00.000000000+03:30`
• end: پایان بازه زمانی که در آن به دنبال صورتحساب هستیم. فرمت:
`2023-05-14T23:59:59.123456789+03:30`
ورودی‌های start و end اختیاری هستند و در صورتی که مقدار دهی نشوند، جستجو در بین صورتحساب‌های ارسالی در ۲۴ ساعت گذشته انجام می‌گیرد. همچنین بازه زمانی مورد استعلام نمی‌تواند از یک هفته بیشتر باشد.
محل قرارگیری: Request Params
	ورودی

خروجی	آرایه‌ی JSON شامل اشیائی با محتویات زیر (به ازای هر شماره پیگیری یک شیء JSON در این آرایه برگردانده می‌شود): • referenceNumber: شماره پیگیری درخواست استعلام شده • uid: شناسه درخواست (همان requestTraceId که صورتحساب با آن ارسال شده بود) • status: وضعیت صورتحساب دارای حالت‌های زیر: ○ SUCCESS: صورتحساب فاقد خطا بود و با موفقیت در کارپوشه ثبت شد. ○ FAILED: صورتحساب ارسالی دارای خطا بوده و رد شده است. ○ IN_PROGRESS: صورتحساب ارسالی هنوز در صف بررسی می‌باشد. ○ NOT_FOUND: شماره پیگیری داده شده یافت نشد. • data: لیست خطاها / اخطارهای موجود در صورتحساب • packetType: نوع بسته • fiscalId: شناسه حافظه ارسال‌کننده‌ی صورتحساب • sign: امضای پاسخ به فرمت بسته‌ی JWS که ساختار آن در جدول بعدی توضیح داده می‌شود (فقط در حالت SUCCESS)

همانطور که گفته شد خروجی درخواست استعلام صورتحساب دارای فیلدی به نام sign می‌باشد که این فیلد در حالت SUCCESS مقدار دهی شده و در سایر حالات برابر با "" می‌باشد. ساختار این فیلد JWS که توسط هسته تولید و مقداردهی می‌شود به شکل زیر است:

امضای پاسخ نتیجه‌ی اعتبارسنجی صورتحساب – JWS	
Header	یک شیء JSON دارای فیلدهای زیر: • alg: الگوریتم امضا: RS256 • sigT: زمان امضای توکن Format: yyyy-MM-dd'T'HH:mm:ss'Z' Example: 2023-05-13T10:44:47Z
Payload	شیء JSON دارای فیلدهای زیر: • referenceNumber: شماره پیگیری درخواست صورتحساب ارسالی. مثال:

	6e1c7696-064c-4d95-b9eb-711ab931a734 • clientId: شناسه‌ی فرستنده‌ی صورتحساب. مثال: A1112K • taxId: شماره مالیاتی صورتحساب موفق درج شده در کارپوشه. مثال: A1112K04D1271489087018 • receivedDate: تاریخ دریافت صورتحساب در سامانه‌ی مودیان. مثال: 2024-01-08T11:38:54.388Z
Signature	امضای بسته‌ی حاصل از الحاق Header و Payload با استفاده از الگوریتم RS256 و با استفاده از کلید سازمان. شما می‌توانید صحت امضای بسته را با استفاده از کلید عمومی سازمان (دریافتی از متد Get Server-Information) اعتبارسنجی نمایید.

نمونه داده‌ی امضا شده هنگام استعلام وضعیت صورتحساب:

eyJhbGciOiJSUzI1NiIsInNpZ1QiOiIyMDI0LTAyLTE0VDAzOjA4OjM5WiJ9.eyJyZWZl
cmVuY2VOdW1iZXIiOiIzMTk3NzhlMC03Zjg0LTQyNDktOThmNC1kZTc5OTBjYjdjNjkiLCJjbGl
lbnRJZCI6IkExMTFPSyIsInRheElkIjoiQTExMU9LMDREMzYxMjAwRThEQjIxNCIsInJlY2Vpdm
VkRGF0ZSI6IjIwMjQtMDItMTNUMTE6NTI6NDIuOTgyWiJ9.lanl052gttnAp6F1MX9NxPn9swN4
-
IjFiowTX2ytm_1YgAqgzBLytvTBRrq_NopeIH9LIRd275iFJaysMxbFmVnstTLp91hG_iMGFFQv
AXxwY5EL_LYnexu-
biPSswfTXK9lf6RjKrIIDm0hyBr2pCZpc_ZXSgwLHyaf1p_GgxvYSql4l3xCNzzFKfp3yI8r0Pa
hIHovmUlkbdriH_YXKckDw9ZlH6a3NcVslmj65Ed7pemBMmJLPf_Kj7tn7e8s4dUGh0fs6HEotY
fCB7LMDejPDu6jgbIPkwANHWAxx3dZOeO3bUKSo6SIyInvo3GQvDP5dpyGngkQ9NAme7HKg-
HjDHJJzEcjKACG-
Gj3LEsyQveHrL4s5q9YGYWwavZVinbTRMwl75_LRgIeB1UuWNVLS5PtSk2o6drzRnNJ6iN542HH
tOvnF_y8J57oO1LHXDtpMS5ncQF2e_NJ2tGmFxQs868T1ODgqaHzBc3H7VyQ8KC1uD_W3zAf93e
1ErKbAdVt2eDHZVFHoREZVoYW3m7qraO5fiYcYUle9TUXLARQwd_ihEcKbRtLbtw42ZHqfXZe52
Q77vJdY79Xb8OkUjA5kfJ6g4o80Qqa-NaIoeWrFte0g5e-
SUSJklnd4pZTTApZfCMOngoKFSGJ0USfi0T3M-OX7lyxJbOYKeqUzt4

نمونه درخواست استعلام وضعیت صورتحساب با استفاده از شماره پیگیری:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-reference-
id?referenceIds=f9173085-2316-4ca6-918e-e41aaf7ef8dd&referenceIds=93367b02-
23dd-4568-90e1-2b47d799f361&start=2023-05-
14T10%3A00%3A00.000000000%2B03%3A30&end=2023-05-
14T21%3A00%3A00.000000000%2B03%3A30' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGciO...[ JWT TOKEN ]...q4RcXogA'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
[
  {
    "referenceNumber": "93367b02-23dd-4568-90e1-2b47d799f361",
```

```
    "uid": "2b982bfd-9a60-47cd-9da7-30fc0dabd37d",
    "status": "FAILED",
    "data": {
      "error": [
        {
          "code": "012802",
          "message": "مجاز مقادیر جز «تسویه روش» فیلد در شده وارد مقدار
نیست.",
          "errorType": "ERROR"
        }
      ],
      "warning": [],
      "success": false
    },
    "packetType": "receive_invoice_confirm",
    "fiscalId": "A111OK",
    "sign": ""
  },
  {
    "referenceNumber": "f9173085-2316-4ca6-918e-e41aaf7ef8dd",
    "uid": "c5352f85-9322-41bc-a5b2-9abb130fe622",
    "status": "SUCCESS",
    "data": {
      "error": [],
      "warning": [],
      "success": true
    },
    "packetType": "receive_invoice_confirm",
    "fiscalId": "A111OK",
    "sign":
"eyJhbGciOiJSUzI1NiIsInNpZ1QiOiIyMDI0LTAyLTE0VDAzOjA4OjM5WiJ9.eyJyZWZlcmVuY
2VOdW1iZXIiOiIzMTk3NzhlMC03Zjg0LTQyNDktOThmNC1kZTc5OTBjYjdjNjkiLCJjbGllbnRJ
ZCI6IkExMTFPSyIsInRheElkIjoiQTExMU9LMDREMzYxMjAwRThEQjIxNCIsInJlY2VpdmVkRGF
0ZSI6IjIwMjQtMDItMTNUMTE6NTI6NDIuOTgyWiJ9.lanl052gttnAp6F1MX9NxPn9swN4-
IjFiowTX2ytm_1YgAqgzBLytvTBRrq_NopeIH9LIRd275iFJaysMxbFmVnstTLp91hG_iMGFFQv
AXxwY5EL_LYnexu-
biPSswfTXK9lf6RjKrIIDm0hyBr2pCZpc_ZXSgwLHyaf1p_GgxvYSql4l3xCNzzFKfp3yI8r0Pa
hIHovmUlkbdriH_YXKckDw9ZlH6a3NcVslmj65Ed7pemBMmJLPf_Kj7tn7e8s4dUGh0fs6HEotY
fCB7LMDejPDu6jgbIPkwANHWAxx3dZOeO3bUKSo6SIyInvo3GQvDP5dpyGngkQ9NAme7HKg-
HjDHJJzEcjKACG-
Gj3LEsyQveHrL4s5q9YGYWwavZVinbTRMwl75_LRgIeB1UuWNVLS5PtSk2o6drzRnNJ6iN542HH
tOvnF_y8J57oO1LHXDtpMS5ncQF2e_NJ2tGmFxQs868T1ODgqaHzBc3H7VyQ8KC1uD_W3zAf93e
1ErKbAdVt2eDHZVFHoREZVoYW3m7qraO5fiYcYUle9TUXLARQwd_ihEcKbRtLbtw42ZHqfXZe52
Q77vJdY79Xb8OkUjA5kfJ6g4o80Qqa-NaIoeWrFte0g5e-
SUSJklnd4pZTTApZfCMOngoKFSGJ0USfi0T3M-OX7lyxJbOYKeqUzt4"
  }
]
```


همانطور که می‌بینید یکی از صورتحساب‌ها با موفقیت ثبت شده و هیچ خطا و اخطاری ندارد. ولی یکی از صورتحساب‌ها دارای خطا بوده و خطای موجود در آن در قسمت error قابل مشاهده می‌باشد.

۸- ۲- استعلام با uid

فراخوانی این API دقیقا همانند استعلام به وسیله Reference number است با این تفاوت که در ورودی به جای شماره پیگیری، uid و fiscalId می‌گیرد.

استعلام به وسیله شناسه درخواست – Inquiry by uid	
آدرس	https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-uid
Method	GET
ورودی	فیلدهای زیر: • uidList: لیست شناسه درخواست‌های صورتحساب‌های مورد نظر • fiscalId: شناسه حافظه صادر کننده صورتحساب • start: شروع بازه زمانی که در آن به دنبال صورتحساب هستیم. فرمت: 2023-05-14T00:00:00.000000000+03:30 • end: پایان بازه زمانی که در آن به دنبال صورتحساب هستیم. فرمت: 2023-05-14T23:59:59.123456789+03:30 ورودی‌های start و end اختیاری هستند و در صورتی که مقدار دهی نشوند، جستجو در بین صورتحساب‌های ارسالی در ۲۴ ساعت گذشته انجام می‌گیرد. محل قرارگیری: Request Params
خروجی	دقیقا مانند استعلام با شماره پیگیری

نمونه درخواست و پاسخ این API را می‌توانید ببینید:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-by-
uid?uidList=cb080c58-e36f-4bb0-a932-90f672109fb6&uidList=b3bd6327-1c57-
4cae-85ed-5c88de28aea3&fiscalId=A111YO&start=2023-06-
10T00%3A00%3A00.000000000%2B03%3A30&end=2023-06-
10T23%3A59%3A59.999999999%2B03%3A30' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGciOiJ...[JWT TOKEN]...1kvski8e-A'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
[
  {
    "referenceNumber": "780c7cb1-84cf-4df8-a87d-160448f38c55",
    "uid": "b3bd6327-1c57-4cae-85ed-5c88de28aea3",
    "status": "FAILED",
    "data": {
      "error": [
        {
```

```
          "code": "011107",
          "message": "فیلد در شده وارد مقدار طول «شناسه/ملی شناسه/شماره» مقدار طول) .است مجاز طول حداکثر از بزرگتر «خریدار فراگیر کد/مدنی مشارکت باشد 10 حداکثر باید شده وارد)",
          "errorType": "ERROR"
        }
      ],
      "warning": [
        {
          "code": "111208",
          "message": "فیلد در شده وارد مقدار طول «خریدار اقتصادی شماره» مقدار طول) .است مجاز طول حداقل از کوچکتر باشد 14 حداقل باید شده وارد)",
          "errorType": "WARNING"
        }
      ],
      "success": false
    },
    "packetType": "receive_invoice_confirm",
    "fiscalId": "A111OK",
    "sign": ""
  },
  {
    "referenceNumber": "60e834f9-14b6-43be-8b77-85471678d3da",
    "uid": "cb080c58-e36f-4bb0-a932-90f672109fb6",
    "status": "SUCCESS",
    "data": {
      "error": [],
      "warning": [],
      "success": true
    },
    "packetType": "receive_invoice_confirm",
    "fiscalId": "A111OK",
    "sign":
"eyJhbGciOiJSUzI1NiIsInNpZ1QiOiIyMDI0LTAyLTEzVDEyOjAxOjExWiJ9.eyJyZWZlcmVuY
2VOdW1iZXIiOiJmMWQyMDYwZC0zMzZkLTQwNDUtYjg2ZC03ZTAwMTUzOWQyZTUiLCJjbGllbnRJ
ZCI6IkExMTFPSyIsInRheElkIjoiQTExMU9LMDREMzY3MjJEMzk2QzQ2MyIsInJlY2VpdmVkRGF
0ZSI6IjIwMjQtMDItMTNUMTI6MDE6MDcuODk0WiJ9.PKlMMmvlU6aVsB4u6CvVD4UANik5QFm1o
tsnil9EwOiEBTLpn_2S0bkBN5CecIGDX7abRMr8X3H385_gE1oglCv_F7e-
_r5JVzQVbks82AFhUGkT09AGjhTqTQk45MvtOY1mb26YgsWovERPy4ApgINLUwJH9Pw4otnorHI
eNgnu-_-JKTVPul4mnfTFMeb7PTDUyv9uIbpwaR5BbVL8OPtL6J4RAl5lI-coTLnCUMi9LLj-
rUDVkHjdf3_NPL7NADA4GvZwgFZ3Fv2sZxete8z7DBuN4ZdNhvpbs9fYlqdwEkoTE2ZsgD9XapE
lUMDCQt8oAGG4G2W0JInb30F2ZrSqg2UaVSZ3OiCZFbJ8qXoGlX7IomcgCJ2Bs991c4umrn78Mx
2s2UFiF9ckkfJgsIUgpk3jeQ8oEVJuLCRoeTA5MmP4j3Z_z2AFDzkazhZAXmGfDtp6O80HHKz4K
pm77ccTgIPiIYZawb5VUZe1OdOdguFb2T2q3aeP7bA91ue4ntmYaL19vjaEhi7XI5XdbH189t7x
2OaHx_DR70h2KI9hbFZBSEZJM_JmXPDJ6slhFvHcblMPf-
p9fT2r3NOj5mNiBUREnTLi6NvUXwUz6hhspdvdE-xrW51aMkGPp4_Zy51YNKi9KZ1d_EWhMpwe-
HtQjXeVEyc8ldkE4uSP6IE"
  }
]
```


همانطور که مشاهده می‌کنید، یکی از صورتحساب‌ها با موفقیت در کارپوشه ثبت شده و دیگری دارای خطا می‌باشد و لیست خطاها و هشدارهای موجود در صورتحساب برگردانده می‌شوند.

۸– ۳– استعلام براساس بازه‌ی زمانی

این API نیز مانند دو مورد قبلی است و صورتحساب‌های فرستاده شده در یک بازه‌ی زمانی مشخص را به صورت صفحه‌بندی شده برمی‌گرداند. همچنین امکان فیلتر کردن صورتحساب براساس وضعیت نیز وجود دارد.

Inquiry – استعلام در بازه زمانی	
آدرس	https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry
Method	GET

ورودی	فیلدهای زیر:
	• pageNumber: شماره صفحه (مقدار پیشفرض: ۱)
	• pageSize: اندازه‌ی صفحه (مقدار پیشفرض: ۱۰ – مقادیر مجاز: ۱ تا ۱۰۰)
	• status: وضعیت صورتحساب‌هایی که به دنبال آن هستیم.
	○ مقادیر مجاز: SUCCESS, FAILED, TIMEOUT, IN_PROGRESS
	○ در صورتی که این فیلد خالی باشد همه‌ی صورتحساب‌ها برگردانده می‌شود.
	• start: شروع بازه زمانی که در آن به دنبال صورتحساب هستیم. فرمت:
	`2023-05-14T00:00:00.000000000+03:30`
	• end: پایان بازه زمانی که در آن به دنبال صورتحساب هستیم. فرمت:
	`2023-05-14T23:59:59.123456789+03:30`
	ورودی‌های start و end اختیاری هستند و در صورتی که مقدار دهی نشوند، جستجو در بین صورتحساب‌های ارسالی در ۲۴ ساعت گذشته انجام می‌گیرد.
	محل قرارگیری: Request Params

خروجی	دقیقا همانند استعلام با شماره پیگیری و uid

نمونه درخواست ارسالی:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry?start=2023-06-10T00%3A00%3A00.000000000%2B03%3A30&end=2023-06-10T23%3A59%3A59.999999999%2B03%3A30&pageNumber=1&pageSize=10' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGciOi...[JWT TOKEN]...iKKXzBjRZw'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
[
  {
    "referenceNumber": "780c7cb1-84cf-4df8-a87d-160448f38c55",
    "uid": "b3bd6327-1c57-4cae-85ed-5c88de28aea3",
    "status": "FAILED",
    "data": {
      "error": [
        {
          "code": "011107",
          "message": "فیلد در شده وارد مقدار طول» شماره/ملی شناسه/شماره
مقدار طول) .است مجاز طول حداکثر از بزرگتر «خریدار فراگیر کد/مدنی مشارکت
باشد 10 حداکثر باید شده وارد)",
          "errorType": "ERROR"
        }
      ],
      "warning": [
        {
          "code": "111208",
          "message": "فیلد در شده وارد مقدار طول» شماره اقتصادی خریدار«
مقدار طول) .است مجاز طول حداقل از کوچکتر باید شده وارد حداقل 14 باشد)",
          "errorType": "WARNING"
        }      ],
      "success": false
    },
    "packetType": "receive_invoice_confirm",
    "fiscalId": "A111OK",
    "sign": ""
  },
  {
    "referenceNumber": "60e834f9-14b6-43be-8b77-85471678d3da",
    "uid": "cb080c58-e36f-4bb0-a932-90f672109fb6",
    "status": "SUCCESS",
    "data": {
      "error": [],
      "warning": [],
      "success": true
    },
    "packetType": "receive_invoice_confirm",
    "fiscalId": "A111OK",
    "sign":
"eyJhbGciOiJSUzI1NiIsInNpZ1QiOiIyMDI0LTAyLTE3VDA5OjA3OjUzWiJ9.eyJyZWZlcmVuY
2VOdW1iZXIiOiI4MjczYzhkMi1jNWJmLTQyYmQtYTQwZi0wNDA1MjA5YzU2MjkiLCJjbGllbnRJ
ZCI6IkExMTFPSyIsInRheElkIjoiQTExMU9LMDREM0FFQjM1MjAxQTc1NyIsInJlY2VpdmVkRGF
0ZSI6IjIwMjQtMDItMTdUMDk6MDc6NTEuNDUyWiJ9.VnC48sJLngr2YqjpD50bHYmBP0oIZP-
KK7CxB9aZjTPAU5x4xKSyYdCNGRfWlLmnHRpVAHUKs3oeRsHF_uxfD2nGDC6qXHg6MFOcnLU3Hc
IETk-tqIgyVRv3eO6zMyf9ImjatQkD-niW0wjyk7YkK_wl3XFy7lve4YMnR2yE49V-
1ejjj1CwlnICyVlMKG-
IkpnGNy76HA3V0mPZ7Jtlb2cSFt037WVkD9eAGMSl8biQYXsPHgsu1f1USHIrqAWCGaqQfkp_qF
C2nHC4um8RXGAKfxqBQTbmvSEOEk7R3dxRsciNSsxseS9QZIDJqMPmxZcOZFjT0H5Gt8Kpgj2qj
3LcVxXyk3R_QGH1w8FhLU8SAECCnLtkPazZ8I_WHnSAOwXvV_N3gKB5QkCLK-
8No2LPvktsuDT1R5-GB8wI2cTsxsRuG9QI0MCKBvbGVntSR5oEA_UgP-
pT7OMXswFsvkoCAmkEfRaqor0-
IFzNU6z2NVy8bNZ4snwp8VaQNbqXgxuncUVikdoUsbC_uT648alg6QZw4xDeRCbt29Ut9VNY91S
UT9s5BCnyS1B-
```

```
6fk4IhQUO8KkpD37HMTCTNHur33dFQ_qDHG4zcttGI1Q4lj7f6FwP8c1wW59oVCAJfLyekJjGr5
a0aqJtQXF3G5YwBzfQmtRj6hP62cyWC5ixv8"
  }
]
```

۹ – استعلام اطلاعات حافظه و مودی

۹–۱– اطلاعات حافظه

این API اطلاعات مربوط به حافظه مالیاتی (شامل فعال‌بودن / عدم فعال‌بودن، حد مجاز فروش و شماره اقتصادی متصل به آن) را برمی‌گرداند. نحوه‌ی کار این API به صورت زیر است:

GET Fiscal Information – دریافت اطلاعات حافظه	
https://tp.tax.gov.ir/requestsmanager/api/v2/fiscal-information	**آدرس**
GET	**Method**
• memoryId: شناسه حافظه مورد نظر محل قرارگیری: Request Params	**ورودی**
شیء JSON شامل فیلدهای زیر: • nameTrade: شناسه حافظه • fiscalStatus: وضعیت حافظه • nationalId: شناسه‌ی ملی پرونده / کد ملی صاحب پرونده • economicCode: کد اقتصادی پرونده	**خروجی**

نمونه ورودی:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/fiscal-
information?memoryId=A11216' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGc...[ JWT Token ]...OFh9zw'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
{
  "nameTrade": "A11216",
  "fiscalStatus": "ACTIVE",
  "nationalId": "14003778990",
  "economicCode": "14003778990"
}
```

۹- ۲- اطلاعات مودی

این API اطلاعات پرونده مودی را برمی‌گرداند و نحوه‌ی کار آن به شکل زیر است:

GET Taxpayer – دریافت اطلاعات مودی	
آدرس	https://tp.tax.gov.ir/requestsmanager/api/v2/taxpayer
Method	GET
ورودی	• economicCode: شماره اقتصادی مودی محل قرارگیری: Request Params
خروجی	شیء JSON شامل فیلدهای زیر: • nameTrade: نام مودی / پرونده اقتصادی • taxpayerStatus: وضعیت پرونده شامل فیلدهای: «NOT_ALLOCATED» به معنی تخصیص نیافته، «ACTIVE» به معنی فعال مجاز، «DEACTIVATED» به معنی غیرفعال (عبور از حد مجاز ماده ۶)، «TEMPORARY_UNAUTHORIZE» به معنی غیرمجاز موقت، «PERMANENT_UNAUTHORIZE» به معنی غیرمجاز دائم و «ARTICLE_2_SUBJECT» به معنی مشمول تبصره ماده ۲ می‌باشد. • nationalId: شناسه‌ی ملی پرونده / کد ملی صاحب پرونده

نمونه درخواست ارسالی:

```
curl -X 'GET' \

'https://tp.tax.gov.ir/requestsmanager/api/v2/taxpayer?economicCode=1400377
8990' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGci...[ JWT Token ]...klQXOM-_uA'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
{
  "nameTrade": "انزلی آزاد منطقه ایرانیان الکترونیک پیشخوان",
  "taxpayerStatus": "ACTIVE",
  "nationalId": "14003778990"
}
```

۱۰– استعلام وضعیت صورتحساب در کارپوشه

این سرویس وضعیت صورتحساب ها را برمیگرداند و و نحوه کار آن به شکل زیر است.

GET INVOICE STATUS – استعلام وضعیت صورتحساب	
https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-invoice-status	**آدرس**
GET	**Method**
• taxIds: لیست شماره مالیاتی صورتحساب محل قرارگیری: Request Params	**ورودی**
شیء JSON شامل فیلدهای زیر: • taxId : شماره مالیاتی ارسالی • invoiceStatus: وضعیت صورتحساب دارای مقادیر زیر: REJECTED(رد شده) APPROVED(تایید شده) SYSTEMIC_APPROVED(تایید سیستمی) IMPOSSIBLE_REACTION(عدم امکان واکنش) AWAITING_REACTION(در انتظار واکنش) NO_NEED_REACTION(عدم نیاز به واکنش) CANCELED(باطل شده) • article6Status: وضعیت عبور از حد مجاز ماده ۶ دارای مقادیر زیر: EXCEEDED (عدول) NOT_EXCEEDED(عدم عدول) • error: خطا، دارای مقادیر زیر NOT_FOUND(صورتحساب یافت نشد) ACCESS_DENIED(دسترسی ندارید)	**خروجی**

نمونه درخواست ارسالی:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/inquiry-invoice-
status?taxIds=A111DW04E8300004349008&taxIds=A111DW04E8300003CC4EA5' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJhbGci...[ JWT Token ]...klQXOM-_uA'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
[{
  "taxId": " A111DW04E8300004349008",
  "invoiceStatus": "APPROVED",
  "article6Status": "NOT_EXCEEDED",
  "error": null
},
{
  "taxId": "A111DW04E8300003CC4EA5",
  "invoiceStatus": "APPROVED",
  "article6Status": "NOT_EXCEEDED",
  "error": null
}
]
```

۱۱– ارسال پرداخت صورتحساب

این وب سرویس بدین صورت است که مودی می‌تواند با فراخوانی سرویس پرداخت تجمیعی صورتحساب مرتبط با مورد کاربرد برای صورتحساب های الکترونیکی که روش تسویه آنها به صورت نسیه یا نقدی/ نسیه هستند ثبت پرداخت انجام دهد و وضعیت پرداخت انجام شده را به صورت همگام مشاهده کند.

پرداخت و استعلام پرداخت – POST INVOICE PAYMENT	
آدرس	https://tp.tax.gov.ir/requestsmanager/api/v2/invoice-payment
Method	POST
ورودی	• paymentReqeust: شی ارسال پرداخت محل قرارگیری: Request Body Json شامل فیلد های زیر است • taxid(شماره مالیاتی صورتحساب) • paidAmount(مبلغ پرداخت) • paymentDate(تاریخ و زمان پرداخت) • paymentMethod(روش پرداخت) ▪ CHEQUE : چک ▪ BARTER : تهاتر

	▪ CASH : وجه نقد ▪ POS : دستگاه پوز ▪ INTERNET : درگاه پرداختی اینترنتی ▪ CARD: کارت به کارت ▪ TRANSFER : انتقال به حساب ▪ OTHER : دیگر • terminalNumber(شماره پایانه) • referenceNumber(شماره مرجع پرداخت) PaymentMethod شامل موارد زیر میباشد
خروجی	شیء JSON شامل فیلدهای زیر: • requestStatus: وضعیت درخواست شامل (SUCCESS, FAILED) • error: لیست خطاها شامل : ○ code (کد خطا) ○ Message(پیغام خطا)

نمونه درخواست ارسالی:

```
curl -X 'GET' \
  'https://tp.tax.gov.ir/requestsmanager/api/v2/ /invoice-payment' \
  -H 'accept: */*' \
  -H 'Authorization: Bearer eyJjcml0Ij...-w' \
  -H 'Content-Type: application/json' \
  -d '{
  "taxId": "A111H104EA6001D0B32AC6",
  "paidAmount": 0,
  "paymentDate": 0,
  "paymentMethod": "CHEQUE",
  "terminalNumber": "string",
  "referenceNumber": "string"
}'
```

در صورت صحت توکن و درست بودن ساختار درخواست خروجی برابر است با:

```
{
[
 "requestStatus": "SUCCESS",

  "error": [
    {
      "code": "string",
      "message": "string"
    }
  ]

]
}
```

۱۲- پیوست‌ها

۱۰- ۱- گواهی و کلید خصوصی امضا

گواهی امضایی که در Header بسته‌های Jws فرستاده می‌شود برابر است با:

-----BEGIN CERTIFICATE-----
MIIDejCCAmKgAwIBAgIUV27QXqJjK2EgFy9zeYkpsX+ISPswDQYJKoZIhvcNAQEL
BQAwcTELMAkGA1UEBhMCSVIxDDAKBgNVBAgMA1RlaDEMMAoGA1UEBwwDVGVoMREw
DwYDVQQKDAhNb2hheW1lbjEMMAoGA1UECwwDVGF4MSUwIwYJKoZIhvcNAQkBFhZt
Lm1hbHZlcmRpQG1vaGF5bWVuLmlyMB4XDTIzMDMyNDEzMjgyM1oXDTI0MDMyMzEz
MjgyM1owTTEPMA0GA1UEAwwGQW56YWxpMRcwFQYDVQQKDA5RdW9WYWRpcyBHcm91
cDELMAkGA1UEBhMCSVIxFDASBgNVBAUTCzE0MDAzNzc4OTkwMIIBIjANBgkqhkiG
9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzLgyk5KO6+j9d1ud0ilJArrZ3Whw/w9wEzHB
9yXwENRa5fm5AbRukMF5b6VGeKzD6LZuL9+tfdFfWyPcjI++gGNywzRmEHKpTnzP
1t6NyuXKfm4nBVAblsugSw8Y5DEXRfTqILgBWN/pZ4zGlifEALMcGAs7ADcjnv7b
/tM2wxr9rHxsCvW4HvlzQasK8Qr1CrKgT0EI66rSXCHep/uIONDWp0W2OelMlZtM
6AAjWXRLGcshPIHuK+ZLfAFxWtoGonf6qN9ypos2B18D/EFa8WHON62eYKT0kW3j
BVa3yPEkRwkdDjDu/3CPzymhf3WFYwxpb4t35oWb/qUXGVIdvwIDAQABoy4wLDAf
BgNVHSMEGDAWgBSx+Oq+RO3x/FmyCp+jcmfOH+Fn9TAJBgNVHRMEAjAAMA0GCSqG
SIb3DQEBCwUAA4IBAQAgKATXlnS+pPtAiRIYGtydVU5Vi7Aq+D6QW07uFqcB7vBh
ddN3yX2lVVcwpTNJzhv8UCM+mDMvlmsRVKVtMoo5fHfII92/Wo8rUz1RP+yhyCk0
Vz8I11v+bjLwVur/agC/s5Rf0m66pNNjFZ9J3S2N3lChXYwz2vvA8pdAYvWTu9g5
u4FMFqlsaLwMGC+WaA0g3KYzRkdWRy1vd23hLTUcVsWM8wpgZ11wEGE1khca/Sd0
mCU2HG5vIbqFfTjA6to0fY07CE5fD8aR3UcXjNduosVO52ZqCX5SabrhFS3AGHFR
jpFnI5LZespiCXSA8Sv3kOSCSRQKqFbiwSFM8Zjg
-----END CERTIFICATE-----

همچنین کلید خصوصی متناظر با این گواهی که در فرآیند امضای بسته‌ها استفاده شده کلید زیر است:

-----BEGIN PRIVATE KEY-----
MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQDMuDKTko7r6P13
W53SKUkCutndaHD/D3ATMcH3JfAQ1Frl+bkBtG6QwXlvpUZ4rMPotm4v36190V9b
I9yMj76AY3LDNGYQcqlOfM/W3o3K5cp+bicFUBuWy6BLDxjkMRdF9OoguAFY3+ln
jMaWJ8QAsxwYCzsANyOe/tv+0zbDGv2sfGwK9bge+XNBqwrxCvUKsqBPQQjrqtJc
Id6n+4g40NanRbY56UyVm0zoACNZdEsZyyE8ge4r5kt8AXFa2gaid/qo33KmizYH
XwP8QVrxYc43rZ5gpPSRbeMFVrfI8SRHCR0OMO7/cI/PKaF/dYVjDGlvi3fmhZv+
pRcZUh2/AgMBAAECggEACPpC7YnNzram9ucDosXAt+ftyfHckrLgnVbfRLFbN8G4
QsGSxWpeNublJmo/Due0p63oYx0SBKR75AlMkLV1CzhRPhI8L5h3qEN88dVMrosp
OCYoe+kpbJF9dA0zcD5e4Oh+o/StynH3UF0yED+qLsWsA7nqWnYQj9ZpW2Fz01Z1
i5NRX4YgyIopHfqcLWJWpOR8n4HwDY18BL7tMi31f0sZXz56EUgBPxq5RMi+1iKW
pyZdwy4TrJL/Sj4tWkKJ7ELVd47VunizAqeLDy8bLlPBX0PewRvR37P9axHLjsj/
d1Iw/xvqgVEWZTyUXzg4qgFU5Na5u6xIW8JyqejdEQKBgQD2QlDdoPOtG1URyuuo
4HJlQ85ZfEWWA0vvp8esvJLJ7t73Rz28RkKAMHW/njr23ExxV3eek9J1lGENJUZF
KsNNtjRWMgCxUicO0MGTJuraWHrEma00YNNjpVd3pIi68p51CZNzMRLn2J8F+7CP
eGiynKEvKU43KcJr6KOkWP14FQKBgQDU0T3SNr7af4y3rO3ds2DMaoKEG531tuLu
nrZNnL4a/i2r3AlI5K470UkmAXDD1kgGf235KxrRm7x7VVp5SZLgOPqgJ2OKKk9b
HLvuAdxCeUlnDfJWrAmYcmyLLCSDHBudN/evF6WiNaQi074MhdbaxYdLiiXdS3VV
flQ/8m6/gwKBgD4ijXTeX52V/+j1YnDB8RtL+IzrpkMrocVeeCtFiWQaOXf7KcCP
mcfuckdfDVGsVD1k7HG+qqOwRKykcw6Qs6awCpSVGUekiuZaFf2jHC7rlE522BUX
OT8zQNaXVUiWXxT4zZOLFlIZfkZsMyiAISqwCptzuKCCkOPZVzDoo0vhAoGBAMo8
2XnVyoKLKWc2r4i6OOeo48S1FeP12yuVqXqR1FqEZ1RlMnGR1z1DAjdasRV5oVKD
cDeTzdWZIIE3uFWAJFJt80WUiNQ4ptbXtINWQ0DsT2PebggNTsUPH7UVytDJOjiq
gfZjC2TdgtAR1g3Cdk3J3mtbqeXlGmiXN2rZcIMPAoGADfTonaehrsnscUcH4Dgs
qZdyZm9JRmoNyisLBmbGkTNxoYO9Vm/03u3NMsohkjopt2ly38ZMYXl4FyXKEKcI
977r33JD9PxRcovqFhcPR3WuQrPf6ND3IX6eB5p8d7m6fmFYSe/0NhWoeH99a6/c
csAr2hPMOb/R3GdRewkSGgQ=
-----END PRIVATE KEY-----