میان‌افزار tempurl

محتویات

میان‌افزار tempurl
مفهوم HMAC (Hash-based Message Authentication Code)
فرمت یو آر ال موقت
کلیدهای مخفی (Secret Keys)
امضای HMAC برای URL موقت
کارکرد میان‌افزار
گرفتن فایل‌ها با prefix
بررسی کد
دمو

میان‌افزار tempurl

این میان‌افزار، این امکان را فراهم می‌کند که URLها برای دسترسی موقت به اشیاء ایجاد شوند.

به عنوان مثال، یک وب‌سایت ممکن است بخواهد لینکی برای دانلود یک شیء بزرگ در سوئیفت ارائه دهد، اما حساب سوئیفت دسترسی عمومی ندارد. وب‌سایت می‌تواند یک URL ایجاد کند که برای یک مدت محدود به منابع دسترسی GET فراهم کند. وقتی کاربر مرورگر وب بر روی لینک کلیک می‌کند، مرورگر شیء را مستقیماً از سوئیفت دانلود می‌کند، که نیاز به عملکرد وب‌سایت به عنوان نماینده برای درخواست را از بین می‌برد. اگر کاربر لینک را با تمام دوستانش به اشتراک بگذارد یا به طور تصادفی در یک انجمن منتشر کند و غیره، دسترسی مستقیم با توجه به زمان انقضاء تنظیم شده در هنگام ایجاد لینک محدود می‌شود. علاوه بر این، این میان‌افزار قابلیت ایجاد URLها با امضاهایی را فراهم می‌کند که برای همه اشیاءی که پیشوند مشترکی دارند معتبر هستند. این URLهای مبتنی بر پیشوند برای به اشتراک گذاری مجموعه‌ای از اشیاء مفید هستند. همچنین می‌توان محدودیت‌ها را بر روی آی‌پی که منبع مجاز به دسترسی به آن است، قرار داد. این می‌تواند برای محدود کردن جایی که از URLها می‌توان استفاده کرد، مفید باشد.

برای بهتر دانستن نحوه‌ی کارکرد میان‌افزار، نیاز به دانستن مفهوم HMAC است که در ادامه توضیح آن آمده است.

مفهوم HMAC (Hash-based Message Authentication Code)

فرآیندها و تصمیمات مرتبط با کسب و کار به شدت به اصالت و درستی داده‌ها وابسته هستند. اگر حمله‌کنندگان این داده‌ها را دستکاری کنند، ممکن است بر روی فرآیندها و تصمیمات کسب و کار تأثیرگذار باشد. بنابراین، در حین کار آنلاین از طریق اینترنت، باید مراقبت شود که صداقت حفظ شود یا حداقل بدانیم که داده‌ها تغییر کرده‌اند. این‌جا، زمانی است که HMAC به کار می‌رود.[۱]

HMAC (Hash-based Message Authentication Code) نوعی از Message Authentican Code (MAC) است که با اجرای تابع هش رمزنگاری بر روی داده‌ها (که قرار است تأیید شوند) و یک کلید اشتراکی مخفی به دست می‌آید. همچنین همانند هر نوع MAC، برای اطمینان از صحت داده‌ها و تأیید هویت استفاده می‌شود. بررسی صحت داده‌ها برای طرفین ارتباط لازم است. FTPS، SFTP، HTTPS و سایر پروتکل‌های انتقال از HMAC استفاده می‌کنند. تابع هش رمزنگاری ممکن است SHA-1، MD-5 یا SHA-256 باشد. امضاهای دیجیتال تقریباً شبیه HMAC هستند، به عبارت دیگر هر دو از یک تابع هش و یک کلید مشترک استفاده می‌کنند. تفاوت در کلیدها است، یعنی HMAC از کلیدهای تقارنی (یک کپی مشابه) استفاده می‌کند در حالی که امضاها از کلیدهای ناهمگن (دو کلید متفاوت) استفاده می‌کنند.

HMAC به کاربر و سرور یک کلید خصوصی مشترک ارائه می‌دهد که فقط برای آن‌ها معلوم است. کاربر برای هر درخواست، یک هش یکتا (HMAC) ایجاد می‌کند. هنگامی که کاربر درخواست را به سرور می‌فرستد، داده‌های درخواست‌شده را با یک کلید خصوصی هش می‌کند و آن را به عنوان بخشی از درخواست ارسال می‌کند. هنگامی که سرور درخواست را دریافت می‌کند، HMAC خود را ایجاد می‌کند. سپس هر دو HMAC با یکدیگر مقایسه می‌شوند و اگر هر دو برابر باشند، درخواست کاربر، به عنوان درخواستی معتبر در نظر گرفته می‌شود.

فرمول HMAC به صورت زیر است:

```
HMAC = hashFunc(secret key + message)
```

فرمت یو آر ال موقت

یک URL موقت از URL یک شیء به همراه تعدادی query param که به آن اضافه شده است، تشکیل شده. مثال زیر، نمونه‌ای از یو آر ال موقت است:

```
https://swift-cluster.example.com/v1/my_account/container/object
?temp_url_sig=732fcac368abb10c78a4cbe95c3fab7f311584532bf779abd5074e13cbe8b88b
&temp_url_expires=1323479485
&filename=My+Test+File.pdf
```

این مثال اجزای زیر را نشان می‌دهد:

URL شیء: اجباری. مسیر کامل URL، به شئ درخواستی.

temp_url_sig: اجباری. امضای رمزنگاری شده HMAC که مقادیر HTTP method مجاز، تاریخ انقضاء، مسیر کامل به شیء و secret برای URL در آن رمز شده‌اند. الگوریتم هش (برای مثال، SHA-256 یا SHA-512) باید توسط کلاستر پشتیبانی شود؛ الگوریتم های هش مورد پشتیبانی در کلید tempurl.allowed_digests در قابلیت‌های میان‌افزار، پیکربندی خواهند شد.

temp_url_expires: اجباری. تاریخ انقضاء با فرمت Unix Epoch timestamp یا ISO 8601 UTC timestamp. برای مثال، می‌توان از 1390852007 یا 2014-01-27T19:46:47Z برای نمایش Mon, 27 Jan 2014 19:46:47 GMT استفاده کرد.

filename: اختیاری. این نام می‌تواند جایگزین نام پیش‌فرض فایل را برای URLهای موقت GET شود. Object Storage نام فایل پیش‌فرض را برای URLهای موقت GET براساس نام شیء تولید می‌کند. Object Storage این مقدار را در پاسخ، در هدر Content-Disposition برمی‌گرداند. مرورگرها ممکن است این مقدار نام فایل را به عنوان یک ضمیمه فایل برای ذخیره کردن در نظر بگیرند.

کلیدهای مخفی (Secret Keys)

امضای رمزنگاری استفاده شده در TempURL Middleware و همچنین در Form Post Middleware از یک کلید مخفی استفاده می‌کند. ذخیره‌سازی اشیاء به شما اجازه می‌دهد دو مقدار کلید مخفی برای هر حساب و دو مقدار برای هر کانتینر را ذخیره کنید. هنگام اعتبارسنجی یک درخواست، ذخیره‌سازی اشیاء امضاها را با همه کلیدها مقایسه می‌کند. استفاده از دو کلید در هر سطح امکان تعویض کلید را بدون ابطال URLهای موقت موجود فراهم می‌کند.

برای تنظیم کلیدها در سطح حساب، یکی یا هر دوی هدرهای درخواست زیر را با مقادیر دلخواه در یک درخواست POST برای حساب ایجاد یا ویرایش کنید:

```
X-Account-Meta-Temp-URL-Key
X-Account-Meta-Temp-URL-Key-2
```

برای تنظیم کلیدها در سطح کانتینر، یکی یا هر دوی هدرهای درخواست زیر را با مقادیر دلخواه در یک درخواست POST یا PUT برای کانتینر ایجاد یا ویرایش کنید:

```
X-Container-Meta-Temp-URL-Key
X-Container-Meta-Temp-URL-Key-2
```

مثلا، با دستور زیر می‌توان کلید مخفی‌ای در سطح اکانت ایجاد کرد:

```
TEMPURL_KEY="secret123"
curl -X POST -H "X-Auth-Token: $TOKEN" \
-H "X-Account-Meta-Temp-URL-Key: $TEMPURL_KEY" "$STORAGE_ACCT"
```

در این مثال، مقدار TEMPURL_KEY کلید secret ای است که بعدتر در ساخت یو آر ال موقت هم از آن استفاده خواهد شد. مقدار TOKEN$ توکن احراز هویتی است که از اندپوینت auth گرفته می‌شود. مقدار STORAGE_ACCT$ هم برابر مسیر یو آر ال منتهی به اکانتی است که کلید Temp-URL بر روی آن ایجاد می‌شود. مثلا این مقدار می‌تواند برابر آدرس زیر باشد.

```
http://localhost:8080/v1/AUTH_admin
```

امضای HMAC برای URL موقت

میان افزار URL موقت از امضای رمزنگاری HMAC استفاده می‌کند. این امضا شامل اجزای زیر است:

- متد مجاز: به طور معمول، GET یا PUT.
- زمان انقضاء: در مثالی که در زیر برای امضای HMAC-SHA256 در URLهای موقت نشان داده شده است، زمان انقضاء به 86400 ثانیه (یا 1 روز) در آینده تنظیم می‌شود. لطفاً توجه داشته باشید که باید از UNIX timestamp برای تولید امضا استفاده کنید (در درخواست API نیز استفاده از برچسب زمانی UTC ISO 8601 مجاز است).
- مسیر: شامل /v1/ به بعد و شامل نام یک کانتینر (container) و یک شیء (object) است. مسیر برای امضاهای مبتنی بر پیشوند باید با /prefix:/v1 آغاز شود. در این مرحله، مسیر نباید URL-encode شود.
- کلید مخفی (secret key): از یکی از مقادیر کلیدهای توصیف شده در کلیدهای مخفی استفاده شود.

کارکرد میان‌افزار

برای استفاده از میان‌افزار می‌توان به صورت زیر عمل کرد.

برای ایجاد مسیر یو آر ال موقت نیز به شیوه‌ی زیر می‌توان عمل کرد. کد زیر که با پایتون ۲ قابلیت اجرا دارد، متغیرهای متد درخواست، زمان انقضا، مسیر شی را می‌گیرد. سپس به کمک یه کلید secret آن را امضا می‌کند و خروجی را در متغیر sig می‌ریزد.

```
import hmac
from hashlib import sha1
from time import time

method = 'GET'
duration_in_seconds = 60*60*24
expires = int(time() + duration_in_seconds)
path = '/v1/AUTH_admin/c1/object'
key = 'secret123'
hmac_body = '%s\n%s\n%s' % (method, expires, path)
sig = hmac.new(key, hmac_body, sha1).hexdigest()
print(path+"?"+"temp_url_sig="+sig+"&temp_url_expires="+str(expires))
```

مثلا یک نمونه‌ی خروجی این کد می‌تواند به صورت زیر باشد.

```
/v1/AUTH_admin/c1/object?temp_url_sig=85127715e9a0e73d612c89da8f72ef8f5b8ec3c0&temp_url_expires=1690880815
```

حال کاربر می‌تواند با استفاده از این مسیر، بدون نیاز به احراز هویت فایل را دریافت کند. البته در صورتی که زمان دریافت فایل از زمانی که به عنوان انقضای فایل آمده است نگذشته باشد.

دقت کنید که متغیر key در این مثال باید برابر مقداری باشد که کاربر در هدر X-Account-Meta-Temp-URL-Key در درخواست قبلی به سوییفت اضافه کرده بود.

هم‌چنین می‌توان با استفاده از ابزار swift، یو آر ال موقت ایجاد کرد. دستور زیر نمونه‌ای از رفتار این ابزار است:

```
swift tempurl GET 3600 /v1/my_account/container/object MYKEY
```

که به عنوان مثال می‌تواند خروجی زیر را برگرداند:

```
/v1/my_account/container/object
?temp_url_sig=5c4cc8886f36a9d0919d708ade98bf0cc71c9e91
&temp_url_expires=1374497657
```

با چسباندن دامنه‌ی سوییفت به درخواست و صدا زدن آدرس کلی، به منبع گفته شده می‌توان دسترسی پیدا کرد:

```
https://swift-cluster.example.com/v1/my_account/container/object
?temp_url_sig=5c4cc8886f36a9d0919d708ade98bf0cc71c9e91
&temp_url_expires=1374497657
```

برای بررسی و استفاده این میان‌افزار با استفاده از Node.js، به این سند مراجعه شود.

گرفتن فایل‌ها با prefix

در میان‌افزار tempurl، قابلیتی وجود دارد که می‌توان به جای تعریف یو آر ال موقت برای یک شئ، می‌توان آن را برای یک prefix در یک کانتینر تعریف کرد. بدین وسیله، توانایی تعریف یو آر ال موقت برای تعداد زیادی از اشیا به طور همزمان فعال می‌شود. می‌توان از این قابلیت در کنار میان‌افزار post form استفاده کرد، به این صورت که با استفاده از میان‌افزار post form، می‌توان در یک prefix اشیای متعددی را از کاربر گرفت و بارگذاری کرد و از طریق میان‌افزار tempurl نیز می‌توان آن‌ها را دریافت کرد.

برای تعریف یک prefix به عنوان یو آر ال موقت، نیاز است هرجایی که قرار است مسیر موقت ساخته شود، آدرس زیر، به عنوان مسیری که قرار است مسیر موقتش ساخته شود داده شود:

```
prefix:/v1/AUTH_account/container/pre
```

سپس هنگام استفاده از مسیر موقت، مسیر زیر می‌تواند یک مسیر معتبر باشد:

```
https://swift-cluster.example.com/v1/AUTH_account/container/pre/object?
    temp_url_sig=da39a3ee5e6b4b0d3255bfef95601890afd80709&
    temp_url_expires=1323479485&
    temp_url_prefix=pre
```

دقت کنید که query parameter ای با عنوان temp_url_prefix به مسیر اضافه شده که در کارکرد با مسیر موقت حیاتی است. میان‌افزار از طریق این کوئری پارامتر متوجه استفاده از prefix می‌شود و امضا را با توجه به آن بررسی می‌کند. مسیر نیز به عنوان مسیری معتبر شناخته خواهد شد:

```
https://swift-cluster.example.com/v1/AUTH_account/container/pre/
    subfolder/another_object?
    temp_url_sig=da39a3ee5e6b4b0d3255bfef95601890afd80709&
```

```
temp_url_expires=1323479485&
temp_url_prefix=pre
```

بررسی کد

کد این میان‌افزار شامل چند بخش اصلی می‌باشد که در زیر به تفکیک توضیح داده خواهند شد.

تابع _get_temp_url_info

این تابع، اطلاعات مربوط به Temporary URLها را از محیط WSGI برای درخواست استخراج می‌کند. این اطلاعات ممکن است عبارت باشند از: امضا (sig)، انقضا (expires)، پیشوند (prefix)، نام فایل (filename)، نمایه درون‌خطی (inline) و محدوده IP Temporary URL (temp_url_ip_range).

در واقع، این تابع ابتدا پارامترهای مربوط به Temporary URLها را از Query String در محیط WSGI استخراج می‌کند. سپس از این پارامترها مقادیر متغیرهای مختلف مانند temp_url_expires، temp_url_sig و غیره را تنظیم می‌کند. برای مثال، اگر مقدار temp_url_expires به صورت یک عدد صحیح قابل تبدیل باشد، آن را به عنوان زمان انقضا می‌پذیرد. در صورتی که مقدار temp_url_expires به شکل تاریخ و زمان (مانند فرمت ISO8601) داده شده باشد، تابع تلاش می‌کند آن را به عدد صحیح تبدیل کند.

در نهایت، تابع مقادیر استخراج‌شده را در یک tuple قرار داده و به عنوان خروجی بازمی‌گرداند. این tuple حاوی مقادیر sig، inline، filename، prefix، expires و temp_url_ip_range است که به ترتیب اطلاعات مرتبط با Temporary URL ارائه می‌کنند.

تابع _get_keys

این تابع، مقادیر کلیدهای Temporary URL مرتبط با حساب کاربری یا کانتینر را از سرآیندهای متادیتای Account یا Container استخراج می‌کند و به عنوان لیستی از tupleها بازمی‌گرداند.

تابع ابتدا با استفاده از تابع `get_account_info` اطلاعات مرتبط با حساب کاربری را از محیط WSGI استخراج می‌کند. سپس با استفاده از تابع `get_tempurl_keys_from_metadata` مقادیر کلیدهای Temporary URL از متادیتای حساب کاربری (meta) را بدست می‌آورد.

سپس همین روند برای اطلاعات مرتبط با کانتینر نیز صورت می‌گیرد. تابع ابتدا اطلاعات مربوط به کانتینر را از محیط WSGI با استفاده از تابع `get_container_info` دریافت می‌کند. سپس مقادیر کلیدهای Temporary URL را از متادیتای کانتینر بدست می‌آورد.

در نهایت، تابع یک لیست از tupleها به عنوان خروجی بازمی‌گرداند. هر tuple شامل دو عنصر است: یکی مقدار کلید Temporary URL و دیگری دامنه‌ی اعتبار کلید که می‌تواند مقدار `ACCOUNT_SCOPE` یا `CONTAINER_SCOPE` باشد. این لیست تا ۴ tuple مختلف حاوی مقادیر کلیدهای Temporary URL مربوط به حساب کاربری و کانتینر را شامل می‌شود.

مقادیری که این تابع باز می‌گرداند، همان مقادیری است که در بخش کلیدهای مخفی در محیط سوییفت قرار می‌گیرند.

تابع _get_hmacs

این تابع برای ایجاد HMAC (Hash-based Message Authentication Code) برای Temporary URL ها به کار می‌رود. این HMAC ها از کلیدهای اختصاص یافته به هر حساب کاربری یا کانتینر و ترکیب با دیگر مشخصات مانند زمان انقضا و مسیر درخواست (path) ساخته می‌شوند.

ورودی‌های این تابع عبارت‌اند از:درخواست

- expires: زمان انقضا به صورت UNIX timestamps
- path: مسیری که برای تولید HMAC استفاده می‌شود
- scoped_keys: لیست tupleهای (key, scope) که همانند خروجی تابع `_get_keys()` می‌باشند
- hash_algorithm: الگوریتم هش برای استفاده در HMAC
- request_method: متد درخواست (مثلا GET یا POST)
- ip_range: محدوده آی‌پی‌های مجاز برای دسترسی به منبع

تابع با استفاده از اطلاعات درخواست، زمان انقضا، مسیر، کلیدهای مشخص شده و الگوریتم هش، HMACها را برای هر یک از کلیدها تولید می‌کند و همراه با دامنه‌ی اعتبار کلید در یک لیست از tupleها به عنوان خروجی باز می‌گرداند. این tupleها شامل اطلاعات HMAC و دامنه‌ی اعتبار کلید (که می‌تواند `ACCOUNT_SCOPE` یا `CONTAINER_SCOPE` باشد) می‌باشند.

تابع _call

این تابع اصلی‌ترین قسمت این میان‌افزار است و هنگام صدا زده شدن میان‌افزار، کدهای این قسمت اجرا خواهند شد. وظیفه اصلی این تابع اجرای منطق لازم برای بررسی و اجازه دسترسی به منابع با استفاده از Temporary URLs است.

در ابتدا تابع بررسی می‌کند که آیا درخواست متد OPTIONS است یا خیر. اگر درخواست OPTIONS باشد، درخواست به تابع بعدی ارسال می‌شود. سپس تابع `_get_temp_url_info(env)` فراخوانی می‌شود تا اطلاعات مربوط به Temporary URL مورد درخواست را استخراج کند.

در این تابع، اطلاعاتی نظیر امضا (temp_url_sig)، زمان انقضا (temp_url_expires)، پیشوند (temp_url_prefix)، نام فایل (filename)، نوع توزیع (inline_disposition) و محدوده آی‌پی (temp_url_ip_range) از Temporary URL به دست می‌آیند. سپس این اطلاعات برای بررسی و اعتبارسنجی Temporary URL استفاده می‌شوند.

سپس تابع به تولید HMACها می‌پردازد و با توجه به نوع درخواست (HEAD یا دیگر متدها) و محدوده آی‌پی، HMACهای معتبر برای اعتبارسنجی انتخاب می‌شوند. سپس با استفاده از HMACهای معتبر، دسترسی به منبع موردنظر اعتبارسنجی می‌شود. اگر عملیات اعتبارسنجی موفقیت‌آمیز باشد، تغییرات موردنیاز در محیط و درخواست اعمال می‌شوند و دسترسی به منبع با توکن Temporary URL تائید می‌شود.

دمو

این درخواست برای اضافه کردن کلید مخفی است:

```
STORAGEURL="http://localhost:8080/v1/AUTH_admin/c1/object"
STORAGE_ACCT=$(echo $STORAGEURL | sed -e 's!/v1/\([^/]*\)/.*!/v1/\1!')
TEMPURL_KEY="secret123"
TOKEN="AUTH_tk18915684b9a04cd2a07eed32839ac8a0"
curl -X POST -H "X-Auth-Token: $TOKEN" -H "X-Account-Meta-Temp-URL-Key: $TEMPURL_KEY" "$STORAGE_ACCT"
```

دقت کنید که مقدار TOKEN از اندپوینت auth گرفته شده است. حال می‌توان با این دستور، مسیر موقتی با استفاده از ابزار swift گرفت:

```
swift tempurl GET 600 "/v1/AUTH_admin/c1/object" "secret123"
```

که خروجی آن مقدار زیر است:

```
/v1/AUTH_admin/c1/object?temp_url_sig=85680282a99a03540639004168c00b234d16106a&temp_url_expires=1691228923
```

```
empurl GET 600 "/v1/AUTH_admin/c1/object" "secret123"
t?temp_url_sig=85680282a99a03540639004168c00b234d16106a&temp_url_expires=1691228923
```

گرفتن مسیر موقت

اکنون با صدا زدن مسیر موقت، می‌توان به محتوای فایل دسترسی پیدا کرد:

```
STORAGE_SERVER="${STORAGEURL/\/v1*/}"
TEMPURL_PATH="/v1/AUTH_admin/c1/object?
temp_url_sig=85680282a99a03540639004168c00b234d16106a&temp_url_expires=1691228923"
curl "$STORAGE_SERVER$TEMPURL_PATH"
```

دقت کنید که TEMPURL_PATH، مقدارش برابر خروجی دستور قبلی که با استفاده از ابزار سوییفت گرفته شد است.

```
user1@military:~/monster/testcname$ curl "$STORAGE_SERVER$TEMPURL_PATH"
version: "3.8"

services:
  monster01:
     image: registry.zdrive.ir/monster:1.7
     restart: always
     container_name: monster01
     hostname: monster01
     init: true
     ports:
     - 8080:8080

     volumes:
      - $PWD/rings:/rings
      - /mnt/sdb1:/srv/node/sdb1
      - /mnt/sdb2:/srv/node/sdb2
      - /mnt/sdb3:/srv/node/sdb3
      - /etc/localtime:/etc/localtime:ro


     networks:
      - monsterkeynet

networks:
  monsterkeynet:
    name: monsterkeynet
```

دریافت فایل با استفاده از مسیر موقت

هم‌چنین برای ایجاد درخواست PUT می‌توان به شیوه‌ی زیر یو آر ال موقت را گرفت:

```
swift tempurl PUT 36000 "/v1/AUTH_admin/c1/object" "secret123"
```

که خروجی زیر را خواهد داشت:

```
/v1/AUTH_admin/c1/object?temp_url_sig=7b892a435e7cf1475507cfd88622bda1a1b9cae5&temp_url_expires=1693150571
```

سپس می‌توان درخواست زیر را انجام داد و فایلی در آن مسیر گفته شده بارگذاری کرد:

```
curl --location --request PUT 'http://localhost:8080/v1/AUTH_admin/c1/object?
temp_url_sig=7b892a435e7cf1475507cfd88622bda1a1b9cae5&temp_url_expires=1693150571' \
--header 'Content-Type: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet' \
--data '@/home/user1/Documents/order1.xlsx'
```

1. https://www.geeksforgeeks.org/what-is-hmachash-based-message-authentication-code/

برگرفته از «https://kateb.burna.ir/w/index.php?title=میان‌افزار_tempurl&oldid=48052»

مشارکت‌کنندگان: مهسا شادی, مبینا حاج ابراهیمی, محمد مهدی ایل بیگی

این صفحه آخرین‌بار در ۱۶ آبان ۱۴۰۲ ساعت ۱۲:۱۲ ویرایش شده‌است.