Uses for token-based licensing

The information on this page refers to v5.1 and newer, which banned sharing both tokens and their dependencies at the same time. For documentation applicable to earlier versions, see documentation for previous versions.

There are many different ways you can use token-based licensing, as described in the following examples.

License pools

One primary use for token-based licensing is to let users purchase a number of pseudo-features that each require one or more real licenses. This gives users a "pool" of licenses they can draw upon for license checkout requests, providing the flexibility to use various combinations of features as their needs require.

For example, say that the product suite MySolutions consists of three individually sold features: MyDraw, MyWrite, and MySpreadsheet. ABC Corp purchases 20 token-based licenses of MySolutions, including all three applications. Whenever there is a checkout request for one of these three applications, a particular number of licenses is taken from the 20-license pool for the MySolutions suite: MyDraw requires 5 licenses for each checkout; MyWrite requires 2 licenses, and MySpreadsheet requires 1 license.

As you can see from the two different scenarios below, a pool of token-based licenses gives customers significantly more flexibility than purchasing a specific number of licenses for each feature. The users can check out any combination of the features, up to the maximum 20-license limit.

Example Scenario 1

Feature	# of Licenses Consumed per Checkout	# of Checkout Requests	Total Licenses Consumed
MyDraw	5	2	10
MyWrite	2	3	6
MySpreadsheet	1	3	3
Totals		**8**	**19**
Number of licenses remaining			**1**

Example Scenario 2

Feature	# of Licenses Consumed per Checkout	# of Checkout Requests	Licenses Consumed
MyDraw	5	1	5
MyWrite	2	4	8
MySpreadsheet	1	7	7
Totals		**11**	**20**
Number of licenses remaining			**0**

Example

The following example shows a token-based license for the license pool scenario described above, where MySolutions is a license pool that is drawn upon to fulfill license requests for MyDraw, MyWrite and MySpreadsheet.

```
FEATURE MySolutions
{
  VENDOR=ABC_Software COUNT=20 KEYTYPE=EXCLUSIVE VERSION=1.0
  KEY=bhq3ed873qcrKHG6783rhJgvkhvTUtxcuBiouVtyCuyVy78Gftq...
}

FEATURE MyDraw
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=MySolutions VERSION=1.0 COUNT=5"
  KEY=b978bv5ybui7Noyg6c3Vd57fngN987NGSC54sDiugU6v5eio8g...
}

FEATURE MyWrite
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=MySolutions VERSION=1.0 COUNT=2"
  KEY=yig*bv7tu6r879yu09yut75evbGJvHGHdrCHJVJGCt79g78gvv...
}

FEATURE MySpreadsheet
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=MySolutions VERSION=1.0 COUNT=1"
  KEY=hygvCTYGg6r67fg890hbyvGTCVKJBjhc5r7y9joiVGckjlnut8...
}
```

Product suite licenses

Product suite licenses specify that one token-based license depends on multiple real licenses. Product suite licenses enforce a logical AND rule, requiring *a ll* licenses to be valid in order to perform a checkout. This is essentially the opposite of license pools, which specify that multiple features depend on a single real license.

For example, MyDraw may be composed of two modules: Sketcher and Printer. A token-based license can specify that in order to use MyDraw, you must have 1 license of each of the two modules.

Example

The following example shows a token-based license for the product suite license scenario described above.

```
FEATURE Sketcher
{
  VENDOR=ABC_Software COUNT=5 KEYTYPE=EXCLUSIVE VERSION=2.0
  KEY=mBpIAWB9Uuzl2b2B3v]vcsFBx7qEQG1SwXCz8A9d612hU3vSKT...
}

FEATURE Printer
{
  VENDOR=ABC_Software COUNT=5 KEYTYPE=EXCLUSIVE VERSION=1.5
  KEY=B9Uuzl2b2B3v]vcsFBx7qEvcsFBx7qEQG1SwXCz8A9d6U3vSKT...
}

FEATURE MyDraw
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=Sketcher VERSION=2.0 COUNT=1"
  TOKEN_DEPENDENCY="FEATURE=Printer VERSION=1.5 COUNT=1"
  KEY=mBpIAWB9Uuzl2b2B3v]vcsFBxBx7qEQG1SwXCz8A9dj1g866USKT...
}
```

Alternate licenses

You can use token-based licenses to allow license requests to be fulfilled by one or more alternate product licenses. This enforces a logical OR rule, since it requires one license *or* another to succeed with a checkout.

For example, instead of providing a real license of MyDraw, you could define a license where MyDraw has a token dependency fulfilled by either Lower_Priced_License or Higher_Priced_License. Although the alternate token licenses may be more expensive than the real license, the mapping allows users to more reliably access the software they need.

The order of the token-based licenses in the license file determines the order that alternate checkouts are attempted. For example, if the MyDraw features's Lower_Priced_License token-based license precedes the Higher_Priced_License token-based license, then Lower_Priced_License will be preferred for filling checkout requests. Higher_Priced_License will be used only if Lower_Priced_License is unavailable. End users can change the license order if they desire.

Example

The following example shows a license for the alternate license scenario described above. Note that the first MyDraw token-based license refers to Lower_Priced_License, so this will be the preferred license for MyDraw checkout requests.

```
FEATURE Lower_Priced_License
{
  VENDOR=ABC_Software COUNT=5 KEYTYPE=EXCLUSIVE VERSION=1.0
  KEY=mBpIAWB9Uuzl2b2B3v]8GJqW300arlnWmnT01nZXSOIYdF...
}

FEATURE Higher_Priced_License
{
  VENDOR=ABC_Software COUNT=10 KEYTYPE=EXCLUSIVE VERSION=1.0
  KEY=mBpIAWB9Uuzl2b2B3v]vcsFBx7qEQG1SwXCz8A9d6U3vSKT...
}

FEATURE MyDraw
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=Lower_Priced_License VERSION=1.0 COUNT=5"
  KEY=mYsfn30C6ShBYszCq2WVicpTZXQwkfKJTohkzg1wNkle163...
}

FEATURE MyDraw
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=Higher_Priced_License VERSION=1.0 COUNT=10"
  KEY=2w66Ng3wVSVp6ttmWCc8GyJqW300arlnWmnT01nZXSOIYdF...
}
```

Cascading (recursive) licenses

Cascading licenses let you specify a recursive list of token-based licenses, all of which must be available in order to fulfill a checkout request.

For example, say that the application MyWrite depends on one license of MyDraw, which in turn depends on two licenses of Sketcher. With this recursive dependency, MyWrite checkouts will consume two licenses.

The maximum number of recursive dependencies you may define for a token-based license is 16.

The maximum number of dependencies the token-based license may have is 512. (However, there is no limit on the number of token-based licenses you may define.)

Example

The following example shows a token-based license for the cascading license scenario described above.

```
FEATURE Sketcher
{
  VENDOR=ABC_Software COUNT=10 KEYTYPE=EXCLUSIVE VERSION=1.0
  KEY=2w66Ng3wVSVp6ttmWCc8GyJqW300arlnWmnT01nZXSOIYdF...
}

FEATURE MyDraw
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=2.0
  TOKEN_DEPENDENCY="FEATURE=Sketcher VERSION=1.0 COUNT=2"
  KEY=4i]mYsfn30C6ShBYszCq2WVicpTZXQwkfKJTohkzg1wNkle...
}

FEATURE MyWrite
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=2.0
  TOKEN_DEPENDENCY="FEATURE=MyDraw VERSION=2.0 COUNT=1"
  KEY=mBpIAWB9Uuzl2b2B3v]vcsFBx7qEQG1SwXCz8A9d6U3vSKT...
}
```

Token sharing and token dependency sharing

Token-based licenses are always unlimited; therefore, sharing tokens is unnecessary. However, sharing token dependencies may be useful in some cases.

For example, you may have an application with multiple features that can be sold separately or together. Multiple instances of each feature can be shared on a single host. With token sharing, all instances of the particular feature share licenses, but only amongst themselves, not between the different features.

Token dependency sharing allows the features and their instances to share their available licenses and thereby use the licenses more efficiently in cases where you do not want to count features' licenses independently. Dependency sharing can be implemented using multiple token-based licenses that share a common dependency.

Tokens and their dependencies cannot both be shared at the same time.

Example Scenario

For example, say you have two token-based features, "ModuleA" and "ModuleB," with a common token dependency, "Product." ModuleA requires three Product licenses and ModuleB requires two Product licenses, and there are a total of 10 Product licenses.

Without sharing implemented, 2 instances of ModuleA (3 licenses required per instance * 2 instances = 6) and 2 instances of ModuleB (2 licenses required per instance * 2 instances = 4) will consume all 10 Product licenses.

Token sharing lets you share licenses between multiple instances of ModuleA and ModuleB running on the same host. Using the same example with token sharing implemented, two instances of ModuleA and two instances of ModuleB on a single host will use only five licenses, because the licenses are shared *between each feature's instances*. Two instances of ModuleA share two Product licenses amongst themselves and two instances of ModuleB share three Product licenses among themselves.

Token dependency sharing allows for sharing the dependency's licenses freely between ModuleA and ModuleB. Continuing to use the same example with token dependency sharing implemented, two instances of ModuleA and two instances of ModuleB are able to share just three Product licenses.

The example scenario described above is illustrated in the table below.

Sharing Type	Product licenses used for two instances of ModuleA	Product licenses used for two instances of ModuleB
None	6	4
Token sharing	3	2
Token dependency sharing	3	0 (shares 2 licenses with ModuleA)

Token sharing license example

The following example shows a token-based license for the token sharing scenario described above.

```
FEATURE Product
{
  VENDOR=ABC_Software COUNT=10 VERSION=1.0
  KEY=mBpIAWB9Uuzl2b2B3v]vcsFBx7qEQG1SwXCz8A9d6U3vSKT...
}

FEATURE ModuleA
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0 SHARE=HOST
  TOKEN_DEPENDENCY="FEATURE=Product VERSION=1.0 COUNT=3"
  KEY=4i]mYsfn30C6ShBYszCq2WVicpTZXQwkfKJTohkzg1wNkle...
}

FEATURE ModuleB
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0 SHARE=HOST
  TOKEN_DEPENDENCY="FEATURE=Product VERSION=1.0 COUNT=2"
  KEY=2w66Ng3wVSVp6ttmWCc8GyJqW300arlnWmnT01nZXSOIYdF...
}
```

Token dependency sharing license example

The following example shows a token-based license for the token dependency sharing scenario described above.

```
FEATURE Product
{
  VENDOR=ABC_Software COUNT=10 VERSION=1.0 SHARE=HOST
  KEY=mBpIAWB9Uuzl2b2B3v]vcsFBx7qEQG1SwXCz8A9d6U3vSKT...
}

FEATURE ModuleA
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=Product VERSION=1.0 COUNT=3"
  KEY=4i]mYsfn30C6ShBYszCq2WVicpTZXQwkfKJTohkzg1wNkle...
}

FEATURE ModuleB
{
  VENDOR=ABC_Software KEYTYPE=TOKEN VERSION=1.0
  TOKEN_DEPENDENCY="FEATURE=Product VERSION=1.0 COUNT=2"
  KEY=2w66Ng3wVSVp6ttmWCc8GyJqW300arlnWmnT01nZXSOIYdF...
}
```