

# AppID Migration Toolbox

<b>Intro</b>	<b>3</b>
<b>Migration Process</b>	<b>3</b>
Overview	3
Phase 1: Rule Marking	5
Phase 2: Log Analysis	6
Phase 3: Rule Cloning	7
Phase 4: AppID Rules Review	8
Phase 5: Activation of AppID Rules	10
Phase 6: Cleaning of Legacy Rules & more	12
Step 1 : update rules usage statistics	12
Step 2 : run the rules cleaner script	12
FAQ, Tips and Tricks	14

## Intro

AppID Migration Toolbox is a set of scripts to provide an easier, phased, almost freeze-less and foolproof approach to AppID Migration engagements.

The different phases, procedures and safeguards should allow Engineers to adapt to customer's risk aversion level, SLAs and Change Processes and staff availability.

## Migration Process

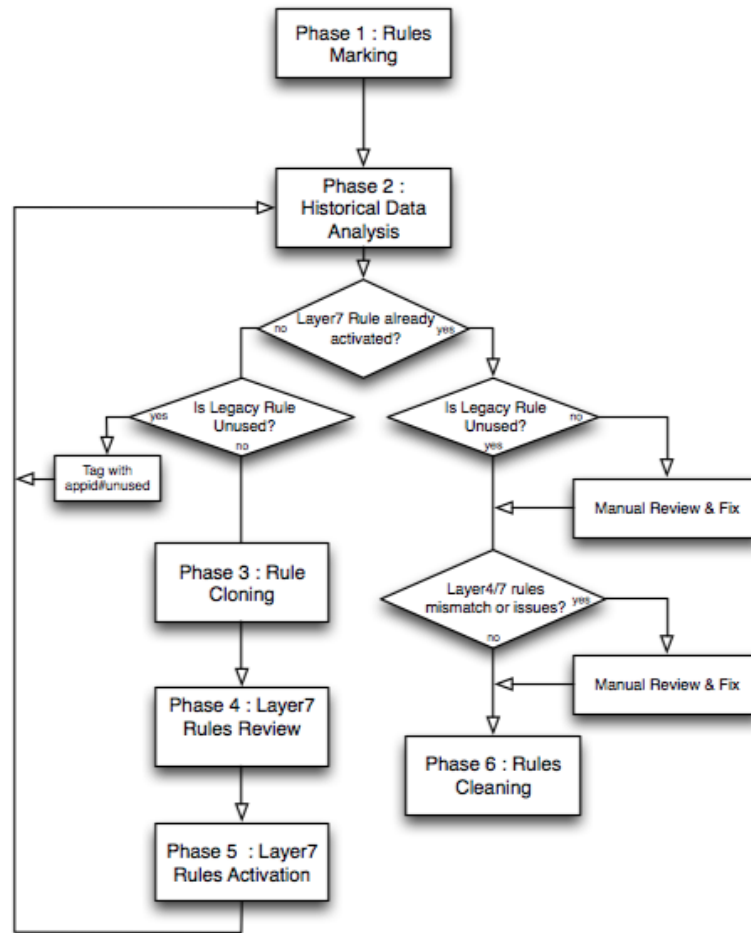
### Overview

Each firewall Vsys or Panorama Device-Group shall be considered as a sub-project for which, the same process will be followed:

- Phase 1: Convertible Rule Marking
- Phase 2: Log Analysis
- Phase 3: Rule Cloning
- Phase 4: AppID rules review
- Phase 5: Activation of AppID rules
- Phase 6: Cleaning of Legacy rules

It's important to know that Phase 2 to 6 can be run several times in loop to ensure the safest and most accurate migration.

High level workflow diagram as follows:



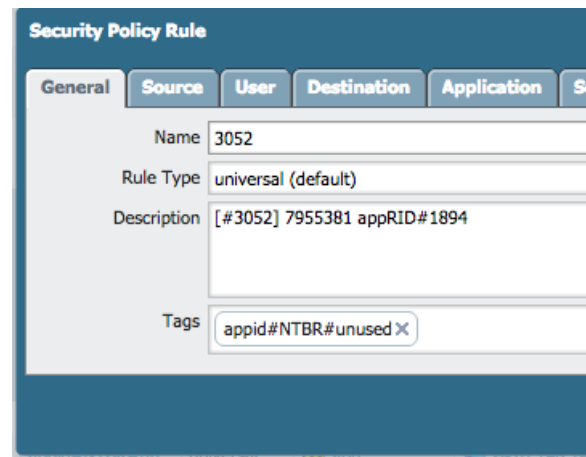
## Phase 1: Rule Marking

Duration : 15 minutes.

Freeze/Config lock: yes.

Remember that our proposed approach doesn't require the customer to do a Freeze on his configuration changes. In order to track rules over several weeks/months and ensure the process will not create troubles with customer's changes like new rules creations, cloning or deletions then our tools will Mark each concerned rule with a unique identifier.

The Identifier is a word that is added into the Rule description field. It is in the form of 'appRID#XXX' where XXX is unique (see screenshot):



The screenshot shows a web interface for configuring a 'Security Policy Rule'. The 'General' tab is selected. The fields are as follows:

Field	Value
Name	3052
Rule Type	universal (default)
Description	[#3052] 7955381 appRID#1894
Tags	appid#NTBR#unused X

For this phase, we will be using script 'rule-marker.php'. A rule will be marked if it fulfills the following conditions:

- It is not marked already
- It is not a DENY rule
- Its Application field is ANY
- It does not have a tag called 'appid#ignore'

Script usage is the following:

```
.>php rule-marker.php in=api://1.1.1.1 location=deviceGroup1
```

## Phase 2: Log Analysis

Duration: variable.

Freeze/Config lock: not needed.

Now that all rules have been marked, we can start looking for applications seen in traffic logs. This can be a lengthy process that depends on numerous factors: number of rules, number of hits, log retention period, busyness of FW/Panorama management and database performances. A period of 1-3 days is common for 1000 rules.

Script 'report-generator.php' will look at all marked rules will start crunching data :

```
> php report-generator.php in=api://xxxx location=deviceGroup2 [OPTIONAL ARGS]
```

Listing optional arguments:

- logHistory=XX : script will generate rules usage reports based on the XX last days time period(default=60)
- updateOnlyUnusedTagRules : only the rules which have tag appid#NTBR#unused will have a usage report generated
- updateOnlyActivatedRules : useful to check if legacy rules are still unused
- resetPreviousData : if previous data was found, erase them and insert newly generated statistics instead (incompatible with update flag)
- updatePreviousData : if previous data was found, erase them and insert newly generated statistics instead (incompatible with reset flag)

This script will store statistics in a XML file located in the current directory. Filename will be VSYS/DG name.xml (ie: deviceGroup1.xml).

If the script happens to terminate (sometimes Panorama can stop answering for a minute or 2 or be rebooted or Log job can fail), just start it again, it will resume where it stopped as the statistics file is updated after each rule report is read from the device.

As you probably already understood, there are no changes made to rules at this point, only log statistics collection.

## Phase 3: Rule Cloning

Duration: 30 minutes

Freeze/Config lock: not required but recommended.

Now that application usage for each eligible rules has been calculated in previous phase, each Legacy rule will be cloned with the following method:

- Clone Legacy rule with name 'legacyRuleName-app'
- Applications found from log collection will be added to the cloned AppID rule.
- AppID rule is created in disabled mode
- AppID rule 'service' field will be set to Any if Legacy rule has 'application-default'.
- If an issue is found, then a Tag is added to Legacy or AppID rule in the form of 'appid#NTBR#XXX'. See Phase 4 to learn more about that.

Note: creating the rule in Disabled mode avoids creating a Change Request on the customer side. It's also very helpful for next phases as manual reviews and changes will still not impact production.

```
php rule-cloner.php help
```

```
***** START OF SCRIPT rule-cloner.php *****
```

```
USAGE: php rule-cloner.php in=api://xxx location=deviceGroup2 [bundleApiCalls]  
[ignoreApps=app1,app2...]
```

## Phase 4: AppID Rules Review

Duration : 1 day – infinite ;)

Freeze/Config lock : not needed.

Here comes the lengthy part of the project: now that legacy rules have cloned with Applications and remain in disable state, it's time to review them one by one with customers. You can also decide to do a first pass on them and request customer assistance and opinion only for a selection of them.

During this phase, you will not make any change on active rules : only on disabled AppID rules in order to add/remove/fix application list. The purpose is to prepare the ground for Phase 5 where a script will activate/enable AppID rules for you. Phase 5 script will activate AppID rules for which there is not tag like 'appid#NTBR#XXX' where NTBR stands for Need To Be Reviewed and XXX as the reason.

So the process is to go after each NTBR tag and fix the rule accordingly. Once done, remove the Tag so Phase 5 can activate the rule.

Note that Phase5 can be applied several times: you can decide after a week of review to activate rules which have been fixed so far while you keep working on the remaining ones.



Here is a listing of Tags, their meaning and hints to fix them:

- **‘appid#NTBR#hasUnknownApps’** : when logs showed unknown applications being used. Please investigate to see what application it is and if a customer signature or App-Override can be worked out.
- **‘appid#NTBR#tooManyApps’** : when more than 10 apps are showing in a single rule. In most cases it means that it’s an ‘internet access’ based rule and shall deserve a different attention (no fixed list of apps) or at least read carefully this list of apps to see if it makes sense.
- **‘appid#NTBR#unused’** : from the logs, this rule is apparently unused. It may be due to the fact that for the moment only 28 days of logs are on Panorama or because they are really used once per quarter or never. You may want to investigate some of these to confirm or deny they are really unused. Whatever, you can just skip this rule, leave the tag as it is and try to refresh log statistics later to confirm it’s really not used.
- **‘appid#NTBR#onlyInvalidApps’** : logs are showing that the rule has only special applications like non-syn-tcp, incomplete and insufficient-data. This is a sign that the rule may not be used at all (rule is open but services on servers are offline/decommissioned, or not in production yet).
- **‘appid#NTBR#hasInsufficientData’** : insufficient-data means that an application was not identified by the firewall and was actually too short to generate an ‘unknown-app’. There are ways to work around this like AppOverride rules, depending on the context and customers wish.
- **‘appid#NTBR#appNotAny’** : there can be quite some time between the day reports are generated and the day I clone rules, in the meantime someone may have editing rules and added applications in it. Our Toolbox has lots of safeguard and tracks this situation. You will probably want to remove the Tag and rule unique Identifier as well to exclude this rule from future AppID process.

## Phase 5: Activation of AppID Rules

Duration: 10-60 minutes

Freeze/config log: required

Activating an AppID rule will happen if the following conditions are met:

- Script was able to find the couple of Legacy + AppID rules holding the same Unique Identifier.
- Legacy and AppID rules have no 'appid#NTBR#XXXX' like tags.
- Legacy rule is not disabled.
- Legacy and AppID rules are identical (outside of the Application field of course !)
- Legacy rule is placed after AppID rule.

When activation happens, the following operations are performed:

1. Legacy rule is renamed with the following model: appRID-XX-YYYY where XX is the Unique Identifier number and YYYY is a random number. Renaming the rule to a new that has never been seen before will help for Phase 6 to know if that rule is still in use.
2. Legacy rule 'log at start' is enabled to ensure the rule generates a log in case it's used in the middle of a session lifetime.
3. AppID rule is renamed with the original Legacy rule name (which ensures better traceability for the customer as the AppID Migration process doesn't change rule names).
4. AppID rule is enabled.
5. Both AppID and Legacy rule get a tag with their activation date in the form of :  
appid#activated#YYYYMMDD where YYYY is year, MM month of the year, DD day of the month. This will be used in Phase 6 to determine which legacy rules are still in use.

Script named 'rule-activation.php' is by default running in 'preview changes only' and is called this way :

```
> php rule-activation.php in=api://1.1.1.1 location=deviceGroup1
...
...
* tag appRID#2876 with 2 rules
- rule 'Rule 181-app'
- rule 'Rule 181'

- legacy rule will be renamed to 'appRID-2876-33819'
- no action taken because 'confirm' argument was not used

**** SUMMARY ****

Number of tags: 2875
Activated: 873
Already Activated: 0

SKIPPED#1 Too many rules :          1
SKIPPED#2 Only 1 rule :           1825
SKIPPED#3 Original rule not found:    1
SKIPPED#4 Cloned rule not found:     0
SKIPPED#5 Cloned rule has NTBR tags:  159
SKIPPED#6 Original rule has NTBR tags:  0
SKIPPED#7 Original rule is disabled:   1
SKIPPED#8 rules mismatch:            15
SKIPPED#9 legacy rule placed before AppID:  0

**** WARNING : no changes were made because you didn't use 'confirm' argument in the command line
****
```

Use the 'confirm' option to actually make changes on the firewall.

```
> php rule-activation.php in=api://1.1.1.1 location=deviceGroup1 confirm
```

Tip : you may want to run the script a few hours/day before without 'confirm' option to do a first pass of possible issues like Legacy vs AppID rule mismatch etc etc.

Tip: you can run this script several times as it will only activate rules which have not been activated yet.

## Phase 6: Cleaning of Legacy Rules & more

After a reasonable amount of time past their cloning & activation, legacy rules which are not used anymore can be safely deleted.

### *Step 1 : update rules usage statistics*

In order to know which rules are still hit or not, you need to fresh the log statistics with report-generator.php with the 'updatePreviousData' flag to make sure it will refresh already existing data. Don't forget to change the logHistory value to match the wanted number of days to compute statistics on if the 60days default is too short for you.

```
> php report-generator.php in=api://1.1.1.1 location=deviceGroup1 updatePreviousData logHistory=90
```

Once this is done, the cleaner script can be used.

### *Step 2 : run the rules cleaner script*

Cleaning a rule means different things that depend on its status.

- Rules which are considered as unused (should be tagged with appid#unused)

These rules were not cloned nor activated because no single logs were found during the whole process. Customer should delete them in the end but it's not your role to do this unless the customer agreed he wanted you to delete them.

The cleaner script will do only 1 thing for these rules: remove rule Unique Identifier from the rule description. The appid#unused tag will stay there to help customers investigate it later.

- Pair of Legacy + AppID Rules

If the script determines that legacy rule is not used anymore then it will proceed to the following:

- o Remove Unique Identifier from AppID rule
- o Remove appid#activated#XXXXXX tag from AppID rule
- o Remove appid#clonedRule tag from AppID rule
- o Delete Legacy rule

There are plenty of reasons why a rule may not be cleaned by the script : it has many safeguards to ensure it doesn't create any trouble to the customer. When you run 'rule-cleaner.php' , at the very end it will provide you with a summary of how many rules it cleaned and reasons why it didn't clean others:

```
> php rule-cleaner.php in=api://1.1.1.1 location=deviceGroup1 [daysSinceLastReport=X] [confirm]
```

```
...
```

```
**** SUMMARY ****
```

```
Number of tags: 2057
```

```
Cleaned: 0 (( if 'confirm' option had been used ))
```

```
SKIPPED#1 Too many rules :           0
SKIPPED#2 Only 1 rule :               0
SKIPPED#3 Original rule not found:    3
SKIPPED#4 Cloned rule not found:      0
SKIPPED#5 Cloned rule has NTBR tags:  0
SKIPPED#6 Original rule has NTBR tags: 0
SKIPPED#7 Original rule is disabled:  0
SKIPPED#8 rules mismatch:             0
SKIPPED#9 legacy rule placed before AppID: 0
SKIPPED#10 appid#ignore flag:         26
SKIPPED#11 still in use:              0
SKIPPED#12 no stats available         0
SKIPPED#13 rule is unused             1567
SKIPPED#14 report was too old         461
```

```
**** WARNING : no changes were made because you didn't use 'confirm' argument in the command line
****
```

You are encouraged to run the script first in preview mode which is the default. No change will be made to the firewall before you use the 'confirm' argument in the CLI. As you are fixing issues and run the CLI again and again then they should vanish or move from a category to another.

Here is how to fix some of the issues listed below:

## FAQ, Tips and Tricks

### 1. Customer does not have enough log retention

Report-generator.php script does collect data from summary logs which should usually have longer term retention. Anyway, if it's not enough then you can run report-generator.php as often as you want and use the following flags to merge statistics with previous runs: "php report-generator.php updatePreviousData ....."