



## QUESTION

The PATROL product allows the use of recovery actions to execute a specific set of instructions whenever a particular alarm condition is triggered. These instructions are stored in the KM file, along with the rest of the knowledge and instructions. One drawback of this method is that a change to the recovery action script requires updating the KM and re-deploying it throughout the environment to maintain version consistency. Another drawback is the lack of flexibility imposed by the fact that recovery actions can only be created at the parameter level.

## SOLUTION

To overcome these drawbacks, the EventSpring product allows recovery action scripts to be created at the Application Class, Instance or Parameter level, which are stored in the Patrol Agent's pconfig database. This means that making a change to a recovery action script is as simple as changing one variable definition and deploying the change with the pconfig utility. The version of the KM remains the same, and the recovery action need only be deployed to the hosts that will make use of it.

## GENERAL CONSIDERATIONS

You can use either OS commands or PSL scripts. In general, PSL recovery actions (RAs) stored with the arsCommand config variable are the easiest to deploy and manage.

However, other factors can help decide which type of RA command to use, such as:

- 1) Requirements to interact with the agent. For example, if you need to collect agent information such as parameter annotations then using PSL RAs is likely the best choice.
- 2) Pre-existing OS scripts that need to be integrated.

## OS COMMANDS

1) Recovery scripts should make no assumptions about a pre-existing environment (e.g., environment variables such as PATH, ORACLE\_HOME, etc). The environment inherited by the PATROL agent is that which existed prior to starting the agent. If there are special environment variables needed by recovery or notification scripts then they must be set BEFORE starting the agent so that they are part of the agent's environment (to check the environment of a Unix agent: `print(system("env"))`); or for NT: `print(system("set"))`;

- 2) Scripts that take a long time to finish executing should be run in the background.

## PSL SCRIPTS

- 1) If the PSL RA script calls an OS command then follow the above guidance.
- 2) Watch PSL script size if running on a preV3.5 agent and saving the PSL script code within the value of the arsCommand rule.
- 3) These 3 rules can be attached to any object level (host, app class, instance or parameter) and can be set differently for ALARMS, WARNings and INFORMATION/OK alerts:

arsAction - Determines if EventSpring should check for a configured recovery action.

arsCmdType - Specifies the type of RA command, e.g., PSL, OS, custom. This rule is optional in many cases. EventSpring will assume the value to be 'OS' by default. If the arsCommand has a '.PSL' extension the 'PSL' will be assumed.

arsCommand - The actual OS command, PSL code or PSL script file to be executed.

## EXAMPLES

### *PSL RA SAMPLE*

This ruleset will send the annotation of the CPUprcrProcessorTimePercent parameter to the Notification Server. The annotation data will be available in the USERDEFINED variable. Note the arsAction is set to 6, which instructs the agent to perform notification (4) and run recovery actions (2).

```
PATROL_CONFIG
"/AS/EVENTSPRING/NT_CPU/CPU__Total/CPUprcrProcessorTimePercent/arsAction" = {
REPLACE = "6" },
"/AS/EVENTSPRING/NT_CPU/CPU__Total/CPUprcrProcessorTimePercent/arsCmdType" = {
REPLACE = "PSL" },
"/AS/EVENTSPRING/NT_CPU/CPU__Total/CPUprcrProcessorTimePercent/arsCommand" = {
REPLACE =
"__undefvar__=annotate_get("\^".__appl_class__.\^".__instance__.\^".__param__);" }
```

### *OS RA SAMPLE*

This ruleset will run the script '/usr/scripts/clean\_tmp.sh' when the /tmp filesystem is near capacity. It will only run when the tmp.FSCapacity parameter is in ALARM. Any output generated by the script will be automatically displayed to connected consoles and save to the notification file identified by the AS\_NOTIFICATION\_FILE environment variable that is set prior to script execution.

```
PATROL_CONFIG
"/AS/EVENTSPRING/FILESYSTEM/tmp/FSCapacity/arsActionALARM" = { REPLACE = "6" },
"/AS/EVENTSPRING/FILESYSTEM/tmp/FSCapacity/arsCmdTypeALARM" = { REPLACE = "OS"
},
"/AS/EVENTSPRING/FILESYSTEM/tmp/FSCapacity/arsCommandALARM" = { REPLACE =
"/usr/scripts/clean_tmp.sh" }
```

## LIMITATIONS

- 1) Only one recovery action can be defined per object. PATROL KMs allow specifying multiple

RAs that would be executed in order at the end of each polling cycle/parameter value update, as long as the alert is outstanding.

The only work-around for this currently is to put more logic within the initial RA.

2) EventSpring does not support the option to 'ALARM only after ALL RAs failed'. If the parameter does not go into ALARM/WARN, the EventSpring RA will not run. The RA could be coded to prevent notification unless the RA failed. In this scenario, the parameter would visually go into ALARM but a NOTIFY\_EVENT would not get generated unless the RA decided that it should.

3) EventSpring does not currently support RAs for OK/INFO alerts.

4) PSL RAs can be limited in size (characters) when the RA code is stored as the value of the arsCommand rule/pconfig variable. This limitation is based on the pconfig value limit (e.g., pre 3.5 this is 1024 characters). Large PSL RAs can be run if stored in an external file. Version 3.5.x of PATROL does not have this size limitation.

---

*Has this article been helpful? Please let us know how we can improve the newsletter in the future by sending suggestions, comments or questions to [articles@advantisms.com](mailto:articles@advantisms.com)*

For more information on the services offered by Advantis Management Solutions, Inc., please feel free to contact us by e-mail at [info@advantisms.com](mailto:info@advantisms.com) or by phone at 617-233-4986.