# Ways to Use SkyView Policy Minder

**SkyView Policy Minder** is an i5/OS & IBM i compliance product that automates security policy compliance monitoring and delivers comprehensive security administration functionality. With Policy Minder you can:

- Check compliance for user profiles, libraries, objects, directories, authorization lists, system values and much more.
- Document your security implementation with unique templates that reflect your security policy requirements.
- Report the details of the compliance status of each section of your security policy.
- Deploy "Fix-It" to return your systems' configuration to your security policy specifications.

Policy Minder allows you to determine the method for reviewing your compliance status that best meets your organization's requirements. You can send the information to a csv file (spreadsheet) or outfile, email a PDF to designated reviewers, review the results via the command line interface displays or review multiple systems' output on one simple display using the browser-based GUI interface.

*(**Note:** Most of our existing clients did many of the following processes "manually" before implementing SkyView Policy Minder to replace them. By automating such procedures, they reduced the time and resources it took them to ensure that their systems remain in compliance, resulting in measureable cost savings.)*

The following are some of the **Ways to Use SkyView Policy Minder** (from either the "green screen" interface or browser-based GUI interface):

- **Create templates to discover "new" items**. Many administrators are creating templates to discover "new" items on their systems. For example, to discover when a new library has been created on the system, they create a library template, include all libraries and set the "Allow new libraries" attribute to be *NO. Any new library created after taking an initial baseline check will be identified. Now you can discover the libraries created by installing vendor software, programmers creating duplicate libraries to test with, and more.
    - Administrators are using the "Allow new xxx" template attribute to manage many aspects of their system. Here are a few more examples:
        - Create a template for the objects in QGPL to discover what programmers are placing in the library.
        - Create a template for all user profiles having *ALLOBJ special authority to discover any new powerful profiles that get created or changed to have *ALLOBJ.
        - Create a template for members of QSECOFR or other profiles to discover new members of powerful group profiles.
        - Create a template for the root ('/') directory to discover newly created directories or other objects created into root.
        - Identify new items and cross-referencing these with their HA system to

    - ensure their HA replication process is working as they expect.
    - When migrating to a new system, create user profile and object templates to identify new objects being created on the existing system so that they can also be created on the new system until the cut-over occurs.
    - Identify new libraries created or restored to ensure they included in the save strategy and documented with any special disaster recovery considerations.
- **Define a policy that reflects how application objects should be owned and the authority settings required**, run a compliance check to determine what objects are out of compliance, run the FixIt function to get the application objects owned and authorized correctly.
  - In addition to making sure there will be no authority errors or exposures due to incorrect ownership or authority settings, this is also a good way to identify programmers who may be circumventing change management processes (Going through change management would have set the appropriate ownership and authorizations. If the objects are owned by the programmer's profile rather than the appropriate application profile it's possible that your change management process was not followed.)
  - You can also ensure that the adopted authority parameters are set correctly for an application's programs, service programs and SQL packages.
- **Define a policy to ensure all user profiles** in a specific user class or that belong to a specific group profile **have the appropriate user profile parameters set**.
  - Employees change positions all the time. If the people creating user profiles are not trained properly they may start creating profiles with attributes that are not correct for the new employee's role. Policy Minder will reveal the profiles that have been created incorrectly so that the attributes can be set properly.
- **Define a policy to automatically manage inactive user profiles**. One area on which auditors often focus is that of inactive profiles (that is, profiles that are no longer in use.) Organizations' policies often require that profiles' status be set to *DISABLED after a certain period of inactivity or that their password be set to *NONE. Using the new "inactive" selection criteria for user profiles, you can use FixIt to automatically set profile attributes (such as the Status, Special authority or Password) to meet your policy's requirements for managing inactive profiles. In addition, you can use FixIt in conjunction with the new *DELETE user profile template to automatically delete profiles that have been inactive for a specified length of time. Using these templates, you can exclude the profiles that are always inactive but shouldn't have action taken on them, for example, group profiles, IBM-supplied profiles, profiles that own objects, etc. Therefore, you can be assured that action is only taken on profiles that are truly inactive.
- **Run the user profile upload** feature to eliminate deleting or disabling user profiles "by hand." You may receive spreadsheets from your HR department of employees that have left the company. Rather than individually keying in each profile name on a command line to delete the profiles, you can upload the profile names from the spreadsheet into the *DELETE template and run FixIt. The profiles will be deleted in batch. In addition, you'll have a complete record of when the profile was deleted and by whom.
- **Document exceptions to your policies**. For example, you can define a policy that no user profile should have a default password. However, you may have third-party vendor

software that ships a profile with a default password and cannot be changed. Using Policy Minder, you can define the no default password policy and then omit the third-party profiles that are an exception to this rule. Your policy (including the exceptions) is now documented for the auditors. Running a compliance check against this policy will not flag the exceptions and will allow you to show an auditor that all other profiles are in compliance with this policy. You can also add a description to your policy stating why the exceptions must be in place along with what processes are in place to mitigate the risks.

- **Monitor compliance for all system values.** With the exception of system values that don't have a fixed value – such as the QDATE and QTIME system values or the values that cannot be changed (such as the QMODEL) you can use Policy Minder to ensure all system values remain set to your required policy settings.
- **Discover newly created or changed programs that adopt authority.** You can choose to specifically check for programs owned by QSECOFR or you can choose to check for programs owned by any user with *ALLOBJ. By initializing your policies – specifically the Adopted authority category – you take a baseline of the programs, service programs and SQL packages that are owned by and adopt an *ALLOBJ user. If new objects are discovered during the next compliance check, they are flagged. This function is not limited to just *ALLOBJ profiles, you can monitor for objects that adopt any profile, such as QPGMR.
- **Determine whether third-party software changed** system values, added another program that adopts an *ALLOBJ user or changed a command to be allowed to run as a limited capability user. You can initialize these policies to your current system settings. Then you can run a compliance check for these policies after installing a third-party package to see if any of these were modified.
- **Monitor command authorities.** Administrators often choose to restrict access to certain i5/OS commands. Whether it is one of the commands pre-defined within Policy Minder or one you add yourself, you can be assured that the commands you have intended to secure stay secured. And you can use the Import Policies function to ensure that the same commands are checked consistently on all systems.
- **Find objects whose object auditing values have been turned on.** Some organizations have, at one time or another, turned on object auditing to discover who or what processes are accessing a particular object. However, it is often the case that the objects' auditing attributes are never set back to *NONE – that is, auditing is not turned off for those objects. This leads to the following scenarios - extraneous audit journal entries are generated or administrators that are leery of turning object auditing on for other objects because of the flood of entries generated by previously set object auditing values. By defining an object template for all objects in a particular library or, for all objects on the system and specifying that the object auditing value is to be set to *NONE, you can discover – by running a Policy Minder compliance check – the objects (including objects in the IFS) for which the auditing value is not *NONE. You can either use Policy Minder FixIt to set the objects' auditing values or change them yourself.
- **Ensure objects are created into the proper library.** Many organizations structure their applications so that program objects get created into one library and data objects (such as files, data areas, etc) into another library. By defining two object templates for each library – one for the object types allowed to be in that particular library (such as all *FILE, *DTAARA, *JRN and *JRNRCV objects) and another that acts as a "catchall" you can very easily determine when any non-approved object type is created into the library.

That's because new objects added to the "catch all" template will be out of compliance. This allows you to ensure application development and code promotion processes are being followed and allows for easy discovery and clean-up if any objects are created, moved or restored into the wrong library.

- **Find source files** embedded in production libraries. To properly secure source files, you have to discover where they're located. Rather than look through long lists of files for each library or write a program to try to discover them, you can define a Policy Minder template to define what their security attributes should be if discovered. Once you run a compliance check, any source files discovered will be listed in the policy. If they are not configured correctly, you can use FixIt to set the proper security settings.
- **Monitor files that are supposed to be journaled** to ensure journaling remains active on critical files.
- **Discover programs and service programs containing dynamic SQL.** Rather than using the PRTSQLINF command on every library and then manually inspecting every spooled file generated, you can use the DSPSQLINF command which produces a report listing the programs containing embedded SQL along with the current Dynamic user profile setting.

- **Integrate compliance checks and FixIt into automated processes.** Policy Minder provides a full set of commands that allow you to integrate the compliance checking and FixIt functions of Policy Minder into your automated processes and procedures. For example:
  - you can call the FixIt commands as an exit out of your change management software to set the authorities of new objects to match the policy settings as defined in your Policy Minder templates.
  - you can call the FIXUSRPRF command as part of a user profile creation process to ensure the profile has the appropriate attributes.
  - if you monitor the system audit journal with monitoring software and detect that authorities have changed on a critical file or perhaps a system value, you can send an alert and your response can be to run one of the FixIt commands (such as the FIXSYSVAL or FIXLIBAUT command) to set the values back to your policy settings.

- **Import policies to other systems**
  - such as from Production to QA to ensure the security settings of the QA system match Production
  - such as from a source system to an HA system then running a compliance check on a periodic basis and especially before a role swap to ensure all system values, user profile and object authorities have been replicated properly
  - to allow you to manage user profiles consistently on all systems
  - to ensure consistent system value settings on all systems