PowerShell Code Review Guidelines

**General**

1. Does the function or script accomplish the intended task?
2. Is there any comment-based help including examples?
3. Is the function or script “advanced” using [CmdletBinding()]?
4. Are risk mitigation parameters supported if appropriate?
5. Does the code use aliases for cmdlets?
6. Does the script or function follow the Verb-Noun Convention?
7. Is the verb in the approved list?
8. Is it the correct verb?
9. Are the noun(s) consistent?
10. Is the function or script in a module with a discoverable name?
11. If the code is a function, does it follow the single responsibility principle?

**Parameters**

1. Do the Parameters have specified types?
2. Are the parameters named appropriately?
3. Are parameters set as mandatory when appropriate?
4. Is any declarative parameter validation used?
5. Are arrays allowed when appropriate?
6. Is pipeline input allowed and implemented properly?
7. Are switch parameters used to flag options?
8. Do parameters have appropriate default values?
9. Are “use cases” divided into ParameterSets?
10. Are named parameters used instead of positional parameters?

**Output**

1. Is the output in the form of objects?
2. Is output produced in the PROCESS block if possible?
3. Are format cmdlets used?
4. Is write-verbose used to supply user-directed output messages?
5. Is write-warning or write-error used for problem messages?
6. Is write-debug used for developer-directed output messages?

**Flow**

1. Are filtering operations as far to the left in pipelines as possible?
2. Are pipelines used appropriately in place of sequential logic?
3. Are pipelines overly used (extremely long pipelines?)

**Comments**

1. Are comments used to explain logic (and not statements)?
2. Is commented-out code present?

**Error Handling**

1. Is there appropriate error handling in the code?
2. Are try/catch/finally used for terminating errors?
3. Are errors signaled with **write-error?**
4. Are terminating errors forced with –ErrorAction STOP?
5. Are console apps checked using $LastExitCode
6. Do write-error calls use the -TargetObject parameter?

**Testing**

1. Are there tests for the code (e.g. Pester)?
2. Do the tests exercise all of the logical paths?
3. Do the tests employ mocks to isolate the code from the envrironment?