AWS Secure Environment Accelerator

Multi-file Accelerator Config file and YAML Support Details

Customers would like the ability to specify their configuration in YAML. This facilitates:

- commenting out entire sections, which is unavailable in standard JSON
- annotating aspects of configuration (e.g. cidr: "10.100.0.0/16" # We chose this for $reason.)
- aligning the Accelerator with CloudFormation, which supports JSON/YAML as input format

Customers would like the configuration file split into multiple files:

- one file for Global options + Mandatory accounts
- one file per OU
- one file for every approx. 2000 lines of workload accounts (Code Commit diff stops working at 3000 lines, allow for adding to each file)

Benefits:

1. Easier cut/paste/comparison of OU configurations
2. Allow CodeCommit diff functionality to function (File currently too large)
3. Allow easier updates to workload accounts (simple append)
4. Smaller scoped updates (de-risk accidentally changing the wrong section)
5. Both a customer request and something the team thought was a good idea

Steps FOR YAML:

- The loadAcceleratorConfig functionality should no longer assume config.json as the config filename in the config repo and/or S3, instead it should look for config.yaml and config.json
- Check for the existence of config.yaml and config.json (initially in S3, but also in CodeCommit on future executions)
- If both files exist, fail with an error message
- **Infer the file type from the extension, and parse accordingly**
- Any other failure should also be an error, fail with an error message
- The accelerator will continue to use JSON formatting internally, if a yaml file is supplied, we are simply converting it to JSON for use by the Accelerator
- All examples throughout this document use config.json as the example, but also apply to YAML
- Both JSON and YAML input files will be equally supported
- Only one file format is supported across all config files, either JSON or YAML, customers can NOT mix YAML and JSON file formats

Steps For File Split:

- When the `__LOAD` keyword is encountered, search relatively (from the same location as root config file) for the file, and insert into the config tree, recursively following `__LOAD` if necessary (to max depth of 2). Any file referenced in `__LOAD` must parse successfully in one of the two formats, otherwise FAIL.

```
{
  "core": {
     "__LOAD": "ous/core.json"
  }
}
```

Note that while we will provide sensible examples, there is no prescriptive requirement for file organization within a customer's configuration, customers can use the feature to break-out sections as is most effective for their deployment. Breaking out large repeatable sections like security groups is a good example and could be included off the main file, an account file, or off an ou file:

`"security-groups": [ "__LOAD": "global/security-groups.json" ]`

Examples:

1. All in one (single file like today):

```
.
  config.json
```

2. Split along major sections:

```
.
  config.json
  ous
    core.json
    dev.json
    test.json ---> could be one per ou, could only be for some ou's as determined by customer
  accounts
    workload-accounts-group1.json
    workload-accounts-group2.json
    my-workload-accounts.json
    more-accounts.json ---->we will encourage each file being as close to 2000 lines as possible (not one
  global
  global-options.json
  security-groups.json

etc
```

- Max depth of 2 means config.json can load ou/dev.json, which can load global/security-groups.json.
- security-groups.json CANNOT load another sub-file (unless security-groups.json was only directly loaded from config.json).

Dealing with Accelerator Automatic Config File Updates:

When customers create AWS accounts directly through AWS Organizations, the Accelerator automatically updates the config file, adding these new accounts. If a customer renames an OU we automatically update the config file. With multi-part files, how do we know what source file to update? We require two mechanisms:

1. Add the following new parameters to the global-options section of the config file

```
"workloadaccounts-param-filename": "accounts/more-accounts2.json",
"workloadaccounts-prefix" : "accounts/more-accounts",
"workloadaccounts-suffix" : 3,
```

- filename is set to `config.json`, and prefix to `config` in a single file configuration scenario (suffix is not used)
- While OU contents can be moved into `__LOADED` sub-files, it was decided the OU object itself must remain in the main config file
- The above parameters:
  - are required to be in the main config file and cannot be `__LOAD`'ed
  - Must be present or SM fails
  - Are used to decide where to add new accounts to the config file

2. Add the following new parameter to each mandatory and workload account config

```
"src-filename": "accounts/my-workload-accounts.json",
```

Accelerator Internal Operations:

- when updating an account in the config file, we use the `"src-filename"` parameters to find and update an accounts `ou`, `ou-path`, `account-name`, and `email` parameters
- When creating new accounts (inserting into config file):
  - if the update is not going to make the file larger than 2000 lines, insert the new account into the config file `"workloadaccounts-param-filename"`
  - if the insert will push the file over 2000 lines:
    * create the next unused filename for the given prefix in Code Commit (`{"workloadaccounts-prefix"}{"workload file format}`), i.e. `"accounts/more-accounts3.json"`
    * insert the new account into the new file in it's entirety
    * update `"workloadaccounts-param-filename"` to: `{"workloadaccounts-prefix"}{"workloadaccounts-suffi file format}`
    * add a new load stmt to the workload-accounts section of the config file with the name `{"workloadaccounts-prefix"}{"workloadaccounts-suffix"}.{customer file format}`
    * update `"workloadaccounts-suffix"` to: `{"workloadaccounts-suffix"} + 1`
    * be careful with comma's between files (JSON sections) when appending/connecting

Example

The entire main config file could be reduced to this:

```
{"global-options": {
   "workloadaccounts-param-filename": "accounts/more-accounts2.json",
   "workloadaccounts-prefix" : "accounts/more-accounts",
   "workloadaccounts-suffix" : 3,
   "__LOAD": "global/global-options.json"
},
"mandatory-account-configs": {
   "__LOAD": "accounts/mandatory-accounts.json"
},
"workload-account-configs": {
   "__LOAD": ["accounts/workload-accounts1.json",
     "accounts/my-other-accounts.json",
     "accounts/workload-accounts2.json"]
},
"organizational-units": {
  "core": {
    "__LOAD": "ous/core.json"
  },
  "Central": {
    "__LOAD": "ous/central.json"
  },
  "Dev": {
    "__LOAD": "ous/dev.json"
  },
  "Test": {
    "__LOAD": "ous/test.json"
  },
  "Prod": {
    "__LOAD": "ous/prod.json"
  },
  "UnClass": {
    "__LOAD": "ous/unclass.json"
```

```
  },
  "Sandbox": {
    "__LOAD": "ous/sandbox.json"
  }
}}
```

Acceptance Criteria:

- A new customer may start an Accelerator deployment with a config.json or config.yaml, and have it deploy as expected so long as the file is semantically correct according to structure and expected keys (and of course syntactically correct in either YAML or JSON)
- Accelerator should continue to function as it does today o i.e. on startup creates repo and copies all referenced config files, not just config.json to repo (json or YAML) o leverages config files in CodeCommit repo from this point forward (json or YAML as provided by customer) o SM runs against the commit id of each file at the start of the SM (i.e. don't allow changes to any file during execution)
- Accelerator leverages multiple config files to receive the same input parameters it previously did from one file
- All accelerator functionality both ALZ and Standalone versions continue to function as previously defined
- Customer can successfully provides multiple config files with the same result as the current one file

---

...Return to Customization Table of Contents