terrascan Documentation

Release 0.1.0

Cesar Rodriguez

Aug 14, 2020

Contents

Contents:

CHAPTER 1

Terrascan

A collection of security and best practice tests for static code analysis of terraform code.

- GitHub Repo: https://github.com/accurics/terrascan
- Documentation: https://terrascan.readthedocs.io.
- Tutorial: https://www.cloudsecuritymusings.com/blog/using-terrascan-for-static-code-analysis-of-your-infrastructure-code-part-

1.1 About Accurics

Accurics enables organizations to protect their cloud native infrastructure in hybrid and multi-cloud environments. It seamlessly scans infrastructure as code for misconfigurations, monitors provisioned cloud infrastructure for configuration changes that introduce posture drift, and enables reverting to a secure posture.

Learn more at https://www.accurics.com

1.2 Features

Terrascan will perform tests on your terraform templates to ensure:

- **Encryption**
  - Server Side Encryption (SSE) enabled
  - Use of AWS Key Management Service (KMS) with Customer Managed Keys (CMK)
  - Use of SSL/TLS and proper configuration
- **Security Groups**
  - Provisioning SGs in EC2-classic
  - Ingress open to 0.0.0.0/0
- **Public Exposure**

  - Services with public exposure other than Gateways (NAT, VGW, IGW)

- **Logging & Monitoring**
  - Access logs enabled to resources that support it

1.3 Installing

Terrascan uses Python and depends on pyhcl and terraform-validate (a fork has been included as part of terrascan that supports terraform 0.12+). After installing python in your system you can follow these steps:

$ pip install terrascan

Terrascan is also available as a Docker image and can be used as follows

$ docker run accurics/terrascan

1.4 Running the tests

To run, execute terrascan.py as follows replacing with the location of your terraform templates:

$ terrascan –location tests/infrastructure/success –vars tests/infrastructure/vars.json

- **Returns 0 if no failures or errors; 4 otherwise**
  - helps with use in a delivery pipeline
- **Parameters**:

```
-h, --help            show this help message and exit
-l LOCATION, --location LOCATION
                      location of terraform templates to scan
-v [VARS [VARS ...]], --vars [VARS [VARS ...]]
                      variables json or .tf file name
-o OVERRIDES, --overrides OVERRIDES
                      override rules file name
-r RESULTS, --results RESULTS
                      output results file name
-d [DISPLAYRULES], --displayRules [DISPLAYRULES]
                      display the rules used
-c CONFIG, --config CONFIG
                      logging configuration: error, warning, info, debug, or
                      none; default is error
```

- **Override file example**

1. The first attribute is the name of the rule to be overridden.
2. The second attribute is the name of the resource to be overridden.

3. The third attribute is the RR or RAR number that waives the failure. This is required for high severity rules; can be an empty string for medium and low severity rules.

```
{
        "overrides": [
                [
                        "aws_s3_bucket_server_side_encryption_configuration",
                        "noEncryptionWaived",
```

(continues on next page)

(continued from previous page)

```
                        "RR-1234"
                ],
                [
                        "aws_rds_cluster_encryption",
                        "rds_cluster_bad",
                        "RAR-98765"
                ]
        ]
}
```

- **Example output**:

```
Logging level set to error.
.................
----------------------------------------------------------------------
Ran 16 tests in 0.015s

OK

Processed 19 files in C:\DEV\terraforms\backends\10-network-analytics


Results (took 1.08 seconds):

Failures: (2)
[high] [aws_dynamodb_table.encryption.server_side_encryption.enabled] should be
↪'True'. Is: 'False' in module 10-network-analytics, file␣
↪C:\DEV\terraforms\backends\10-network-analytics\main.tf
[high] [aws_s3_bucket.noEncryption] should have property: 'server_side_encryption_
↪configuration' in module 10-network-analytics, file␣
↪C:\DEV\terraforms\backends\10-network-analytics\main.tf

Errors: (0)
```

1.5 Using as pre-commit

Terrascan can be used on pre-commit hooks to prevent accidental introduction of security weaknesses into your repository. This requires having pre-commit installed. An example configuration is provided in the comments of the here file in this repository.

1.6 Feature Status

Legend:

- *:heavy_minus_sign:* = test needs to be implemented
- *:heavy_check_mark:* = test implemented
- **blank** - N/A

Terraform resources	Encryption	Security Groups	Public exposure	Logging & Mor
aws_alb			*:heavy_check_mark:*	*:heavy_check_m*
aws_alb_listener	*:heavy_check_mark:*			
aws_ami	*:heavy_check_mark:*			
aws_ami_copy	*:heavy_check_mark:*			
aws_api_gateway_domain_name	*:heavy_check_mark:*			
aws_cloudfront_distribution	*:heavy_check_mark:*			*:heavy_check_m*
aws_cloudtrail	*:heavy_check_mark:*			*:heavy_check_m*
aws_codebuild_project	*:heavy_check_mark:*			
aws_codepipeline	*:heavy_check_mark:*			
aws_db_instance	*:heavy_check_mark:*		*:heavy_check_mark:*	
aws_db_security_group		*:heavy_check_mark:*		
aws_dms_endpoint	*:heavy_check_mark:*			
aws_dms_replication_instance	*:heavy_check_mark:*		*:heavy_check_mark:*	
aws_dynamodb_table	*:heavy_check_mark:*			
aws_ebs_volume	*:heavy_check_mark:*			
aws_efs_file_system	*:heavy_check_mark:*			
aws_elasticache_security_group		*:heavy_check_mark:*		
aws_efs_file_system	*:heavy_check_mark:*			
aws_elasticache_security_group		*:heavy_check_mark:*		
aws_elastictranscoder_pipeline	*:heavy_check_mark:*			
aws_elb	*:heavy_check_mark:*		*:heavy_check_mark:*	*:heavy_check_m*
aws_emr_cluster				*:heavy_check_m*
aws_instance	*:heavy_check_mark:*		*:heavy_check_mark:*	
aws_kinesis_firehose_delivery_stream	*:heavy_check_mark:*			*:heavy_check_m*
aws_lambda_function	*:heavy_check_mark:*			
aws_launch_configuration				*:heavy_check_m*
aws_lb_ssl_negotiation_policy	*:heavy_minus_sign:*			
aws_load_balancer_backend_server_policy	*:heavy_minus_sign:*			
aws_load_balancer_listener_policy	*:heavy_minus_sign:*			
aws_load_balancer_policy	*:heavy_minus_sign:*			
aws_opsworks_application	*:heavy_check_mark:*		*:heavy_minus_sign:*	
aws_opsworks_custom_layer			*:heavy_minus_sign:*	
aws_opsworks_ganglia_layer			*:heavy_minus_sign:*	
aws_opsworks_haproxy_layer			*:heavy_minus_sign:*	
aws_opsworks_instance			*:heavy_minus_sign:*	
aws_opsworks_java_app_layer			*:heavy_minus_sign:*	
aws_opsworks_memcached_layer			*:heavy_minus_sign:*	
aws_opsworks_mysql_layer			*:heavy_minus_sign:*	
aws_opsworks_nodejs_app_layer			*:heavy_minus_sign:*	
aws_opsworks_php_app_layer			*:heavy_minus_sign:*	
aws_opsworks_rails_app_layer			*:heavy_minus_sign:*	
aws_opsworks_static_web_layer			*:heavy_minus_sign:*	
aws_rds_cluster	*:heavy_check_mark:*			
aws_rds_cluster_instance			*:heavy_check_mark:*	
aws_redshift_cluster	*:heavy_check_mark:*		*:heavy_check_mark:*	*:heavy_check_m*
aws_redshift_parameter_group	*:heavy_minus_sign:*			*:heavy_minus_s*
aws_redshift_security_group		*:heavy_check_mark:*		
aws_s3_bucket	*:heavy_check_mark:*		*:heavy_check_mark:*	*:heavy_check_m*
aws_s3_bucket_object	*:heavy_check_mark:*			
aws_security_group		*:heavy_check_mark:*	*:heavy_check_mark:*	

Continued on nex

Table 1 – continued from previous page

Terraform resources	Encryption	Security Groups	Public exposure	Logging & Mor
aws_security_group_rule		*:heavy_check_mark:*	*:heavy_check_mark:*	
aws_ses_receipt_rule	*:heavy_minus_sign:*			
aws_sqs_queue	*:heavy_check_mark:*			
aws_ssm_maintenance_window_task				*:heavy_check_m*
aws_ssm_parameter	*:heavy_check_mark:*			

CHAPTER 2

Installation

2.1 Stable release

To install terrascan, run this command in your terminal:

```
$ pip install terrascan
```

This is the preferred method to install terrascan, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for terrascan can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/cesar-rodriguez/terrascan
```

Or download the tarball:

```
$ curl  -OL https://github.com/cesar-rodriguez/terrascan/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

CHAPTER 3

Usage

To use terrascan in a project:

```
import terrascan
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/cesar-rodriguez/terrascan/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

terrascan could always use more documentation, whether as part of the official terrascan docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/cesar-rodriguez/terrascan/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *terrascan* for local development.

1. Fork the *terrascan* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/terrascan.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv terrascan
$ cd terrascan/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 terrascan tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/cesar-rodriguez/terrascan/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_terrascan
```

CHAPTER 5

Credits

5.1 Development Lead

- Cesar Rodriguez - cesar@cloudsecuritymusings.com

5.2 Contributors

- sbalbach
- Zach Zeid
- Logan Rakai
- Guy Kisel

CHAPTER 6

History

6.1 0.2.3

- Introduces the '-f' flag for passing a list of ".tf" files for linting and the '–version' flag.

6.2 0.2.2

- Adds Docker image and pipeline to push to DockerHub

6.3 0.2.1

- Bugfix: The pyhcl hard dependency in the requirements.txt file caused issues if a higher version was installed. This was fixed by using the ">=" operator.

6.4 0.2.0 (2020-01-11)

- Adds support for terraform 0.12+

6.5 0.1.2 (2020-01-05)

- Adds ability to setup terrascan as a pre-commit hook

6.6 0.1.1 (2020-01-01)

- Updates dependent packages to latest versions
- Migrates CI to GitHub Actions from travis

6.7 0.1.0 (2017-11-26)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search