MIS|TI™ PRESENTS

InfoSecWorld

Conference & Expo 2018

BEYOND AUTOMATED AWS SECURITY ASSESSMENT: ADVANCED USAGE OF SCOUT2

Loic Simon

Director of Product Security @ Tanium

@l01cd3v

INTRODUCTION: SCOUT2 – WHAT?

- Security Tool for AWS
  - Static analysis of the configuration of AWS services and resources
- Open Source
  - Available on GitHub at https://github.com/l01cd3v/AWSScout2
- Python
  - Available via PIP: pip install awsscout2
- Command line tool
- Standalone HTML report
- Does not require AWS pentesting form to be filled out
- **Supports custom rules and rulesets**

INTRODUCTION: SCOUT2 – WHY?

BIZ & IT —

AWS console breach leads to demise of service with “proven” backup plan

Code Spaces closes shop after attackers destroy Amazon-hosted customer data.

Security

Dev put AWS keys on Github. Then BAD THINGS happened

Fertile fields for Bitcoin yields - with a nasty financial sting

Hackers Use Amazon’s AWS Computing Resources to Mine Bitcoin

INTRODUCTION: SCOUT2 – HOW?

1. Preparation
- Installation & setup of required tools
- Configuration of AWS credentials in test environment

2. Getting familiar with the tools
- Run Scout2 in test environment
- Review of the first report
- Discovery and usage of custom summaries

3. Adding value with custom input
- Creation and management of exceptions
- Creation and management of known CIDRs

4. Tweaking the analysis with a custom ruleset
- Update of parameterized rules
- Enable a non-default rule

5. Adding functionality with new rules
- Creation a new parameterized rule with new input
- Creation of a new rule
- Creation of a new parameterized rule from scratch

PREPARATION

INSTALLATION & SETUP

Option1: Install AWSScout2

```
(sudo) pip install awsscout2
(sudo) pip install awscli
git clone https://github.com/nccgroup/aws-recipes
```

Option2: Use the VMWare or VirtualBox image available on the USB stick

```
username: aws
password: scout2
```

If using a virtual machines, ensure that at least 2048MB of RAM has been allocated to the VM; 4096MB is preferable.

CONFIGURATION OF AWS CREDENTIALS

1. Browse to https://training.awsscout2.net
2. Enter your email address and submit the sign up form
3. Check your email for your credentials
4. Browse to https://console.aws.amazon.com
5. In the account alias or account ID field, enter “l01cd3v-scout2”
6. Enter the username and passwords that were generated and provided via email
7. Create a new password when prompted
8. Confirm you were able to sign into the AWS web console

CONFIGURATION OF AWS CREDENTIALS (2)

1. In the AWS web console, click on “Services” then on “IAM”

   https://console.aws.amazon.com/iam/home [Ignore the error messages]

1. Click on “Users”, then click on your username
2. Under the “Security credentials” tab, click on “create an AWS access key”
3. In your terminal, enter the following command

   aws configure --profile scout2lab

5. When prompted for the AWS access key ID, copy and paste the ID displayed in the web console
6. When prompted for the secret key, copy and paste the secret value displayed in the web console
7. Leave the default region name and output format to “None”
8. Confirm the AWS credentials have been successfully configured with the following command

   aws --profile scout2lab iam list-users

FIRST RUN

RUN SCOUT2 AND EXPLORE RESULTS

In the terminal, run the following command

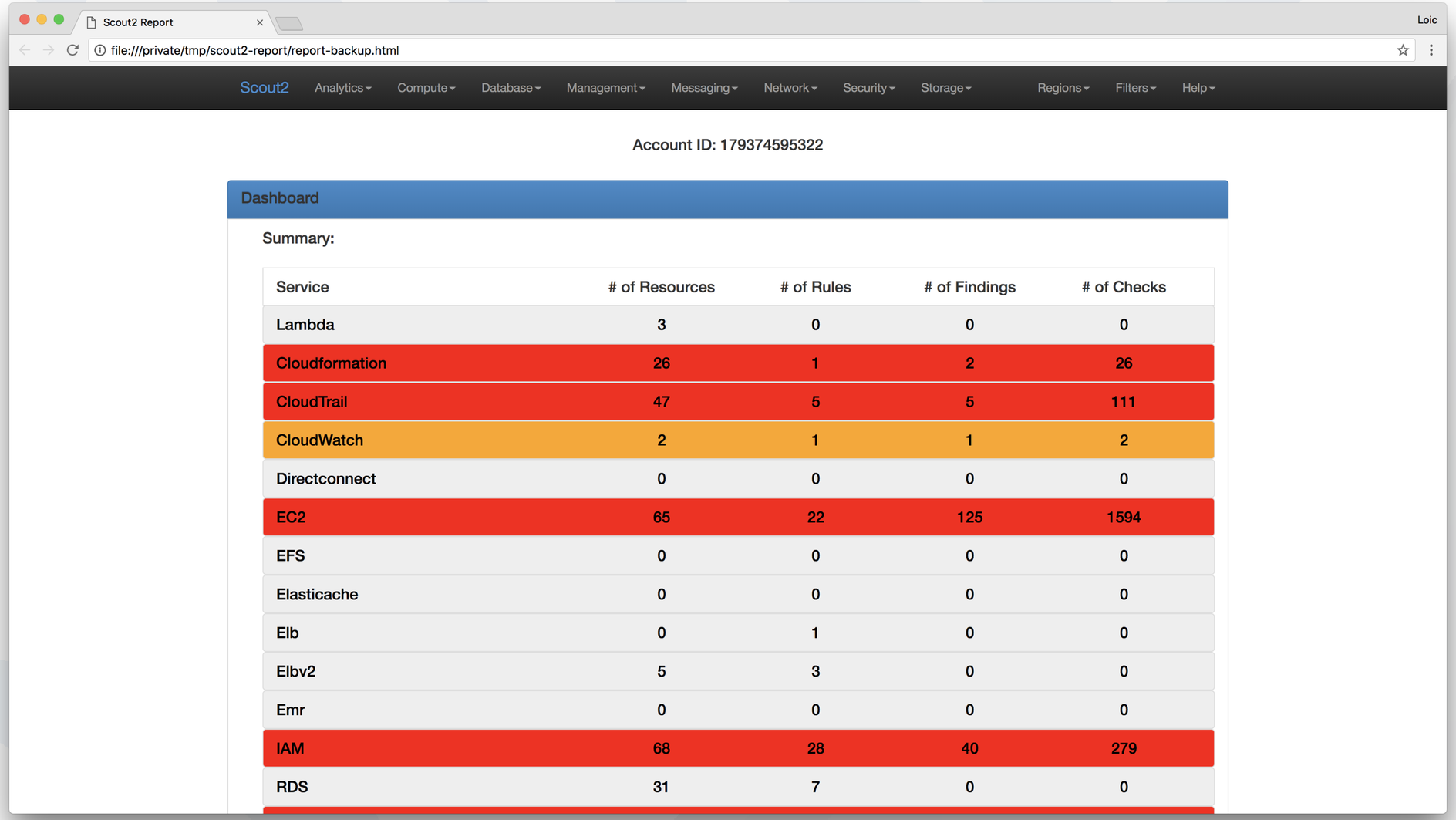
Scout2 --profile scout2lab

Confirm no error messages are displayed, and the HTML report opens in a browser

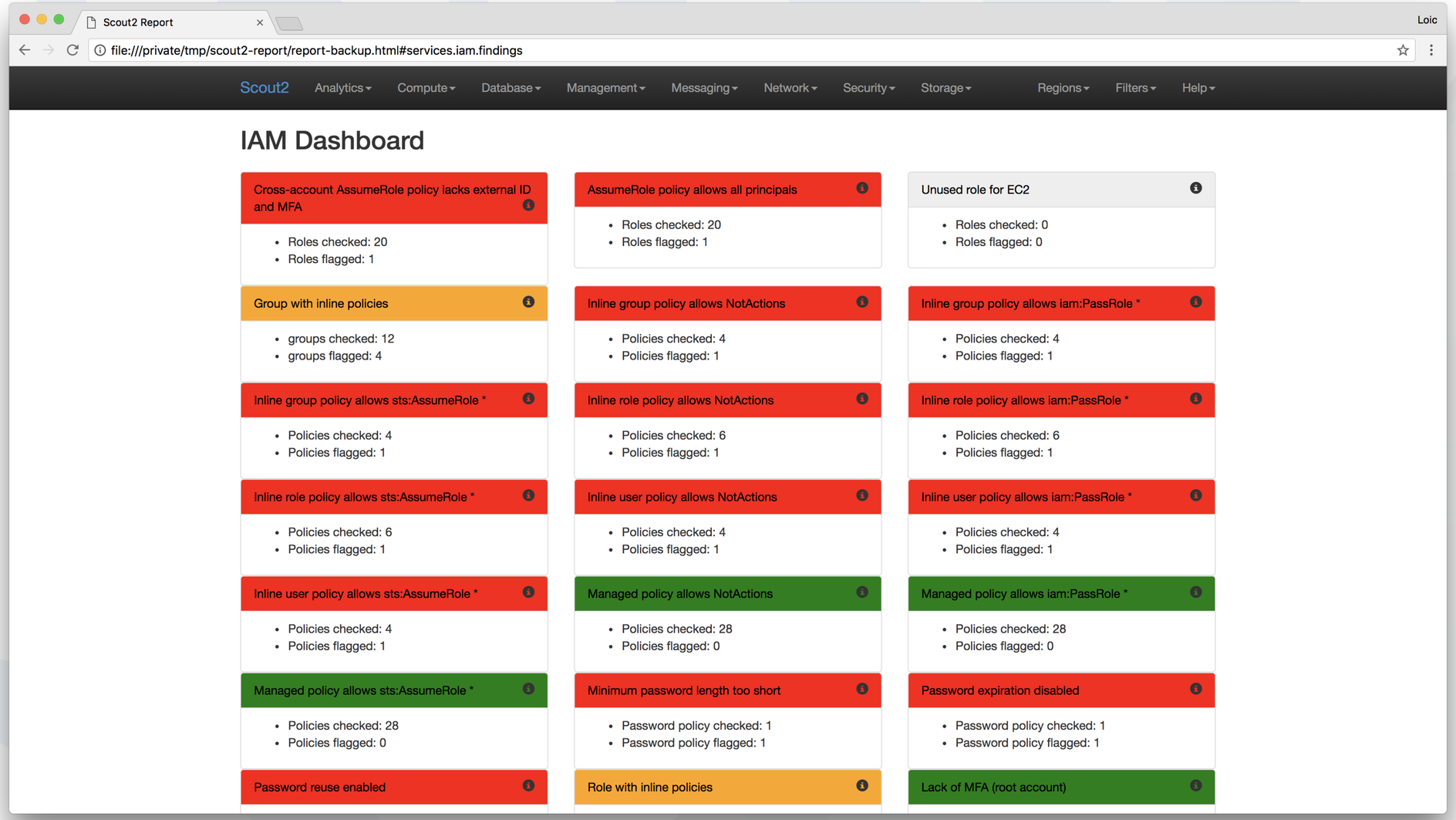
Explore the results by clicking on links in the various dashboard

Explore the custom summaries available in Scout2, such as the IAM permissions summary

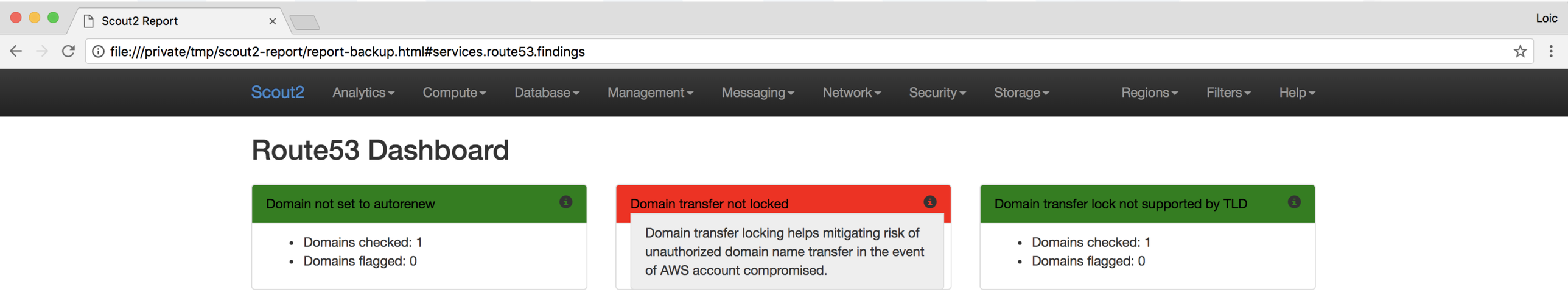
RUN SCOUT2 AND EXPLORE RESULTS: DASHBOARDS



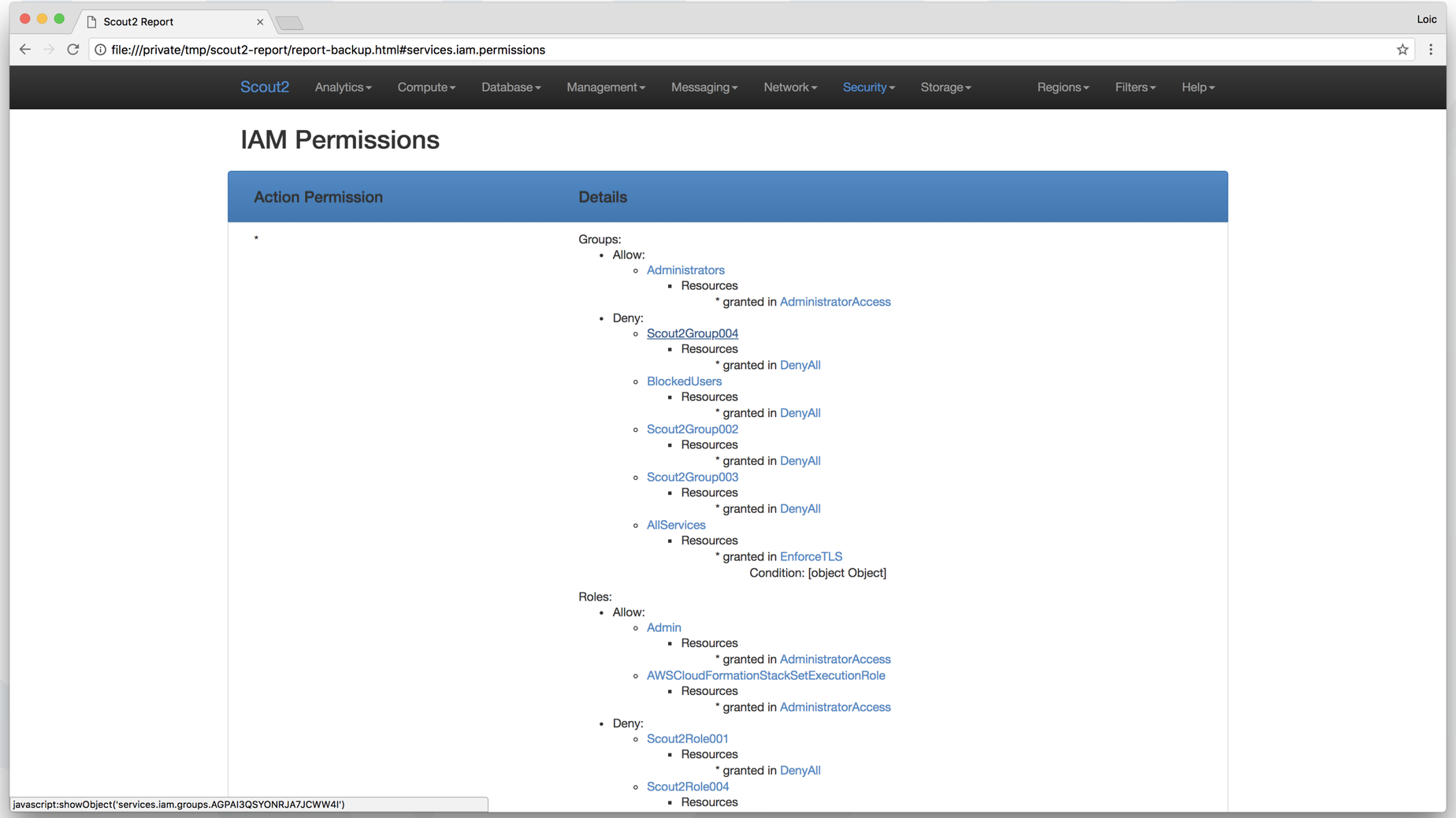
RUN SCOUT2 AND EXPLORE RESULTS: DASHBOARDS



RUN SCOUT2 AND EXPLORE RESULTS: DASHBOARDS



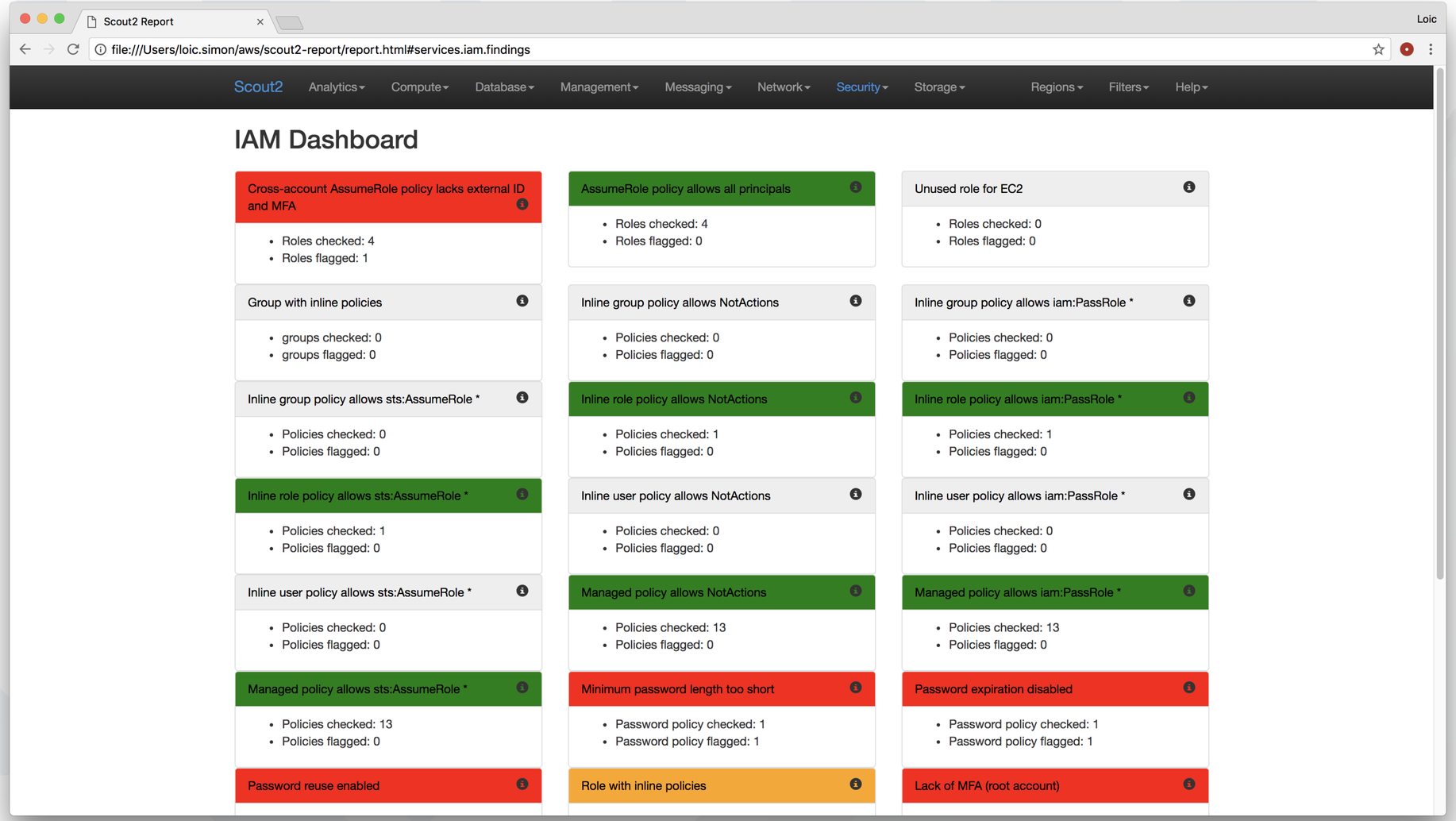
RUN SCOUT2 AND EXPLORE RESULTS: SUMMARIES



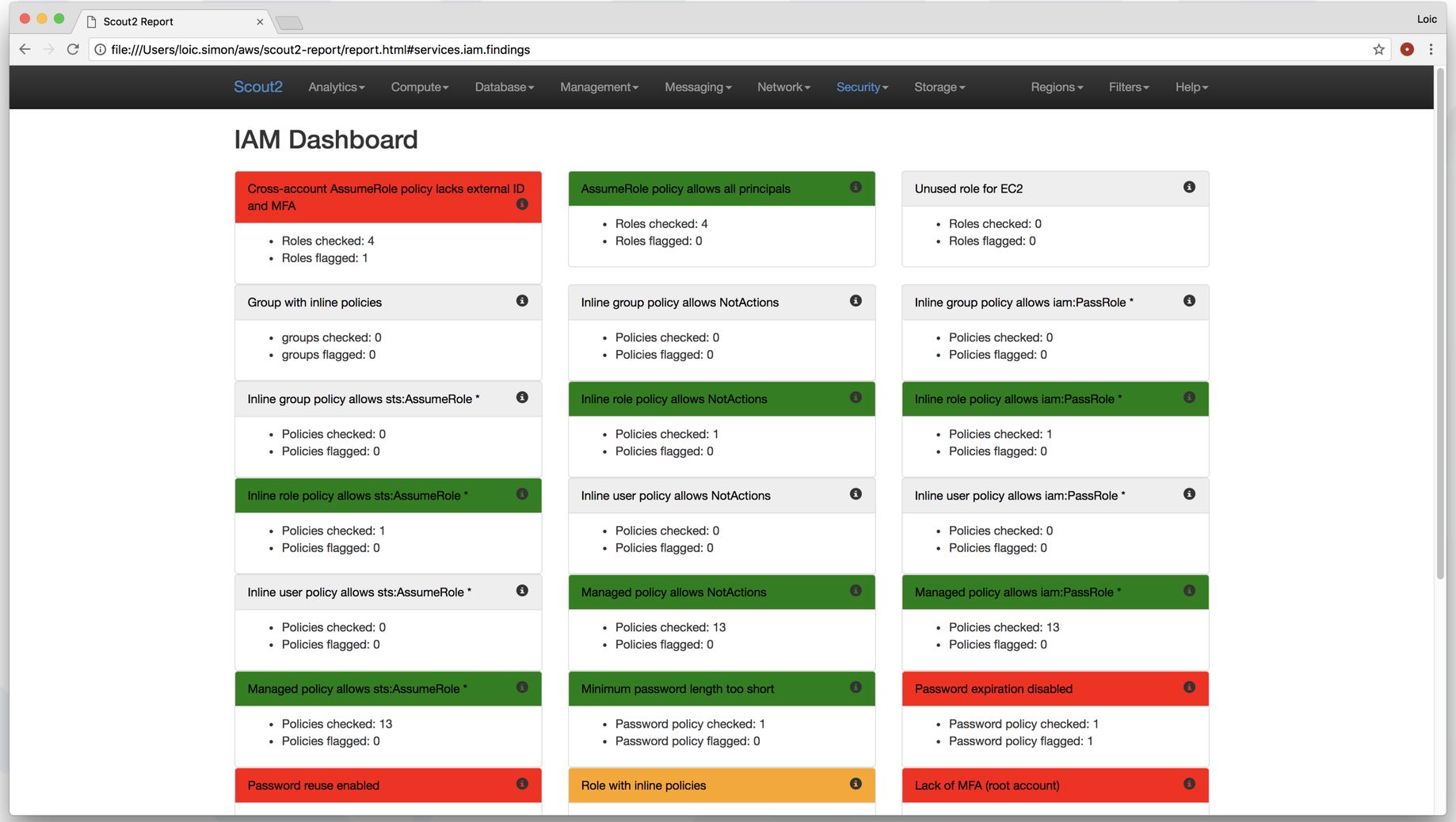
CUSTOM INPUT

MANAGE EXCEPTIONS

MANAGE EXCEPTIONS: EXPECTATION



MANAGE EXCEPTIONS: EXPECTATION



MANAGE EXCEPTIONS: GUIDE

1. In the Scout2 HTML report, browse to a finding
2. Click on the highlighted value
3. Click on “OK” at the “Mark this item as an exception” prompt
4. Click on “Help” then “Export exceptions”
5. Re-Run Scout2 and provide the list of exceptions as an argument

   Scout2 --profile <> --local --exceptions ~/Downloads/exceptions.json

6. Browse to the service dashboard and confirm that the exception count has decreased

MANAGE EXCEPTIONS: UNDER THE HOOD

```
$ cat ~/Downloads/exceptions-report.json

exceptions =
{
    "iam": {
        "iam-password-policy-minimum-length": [
            "iam.password_policy.MinimumPasswordLength"
        ]
    }
}
```

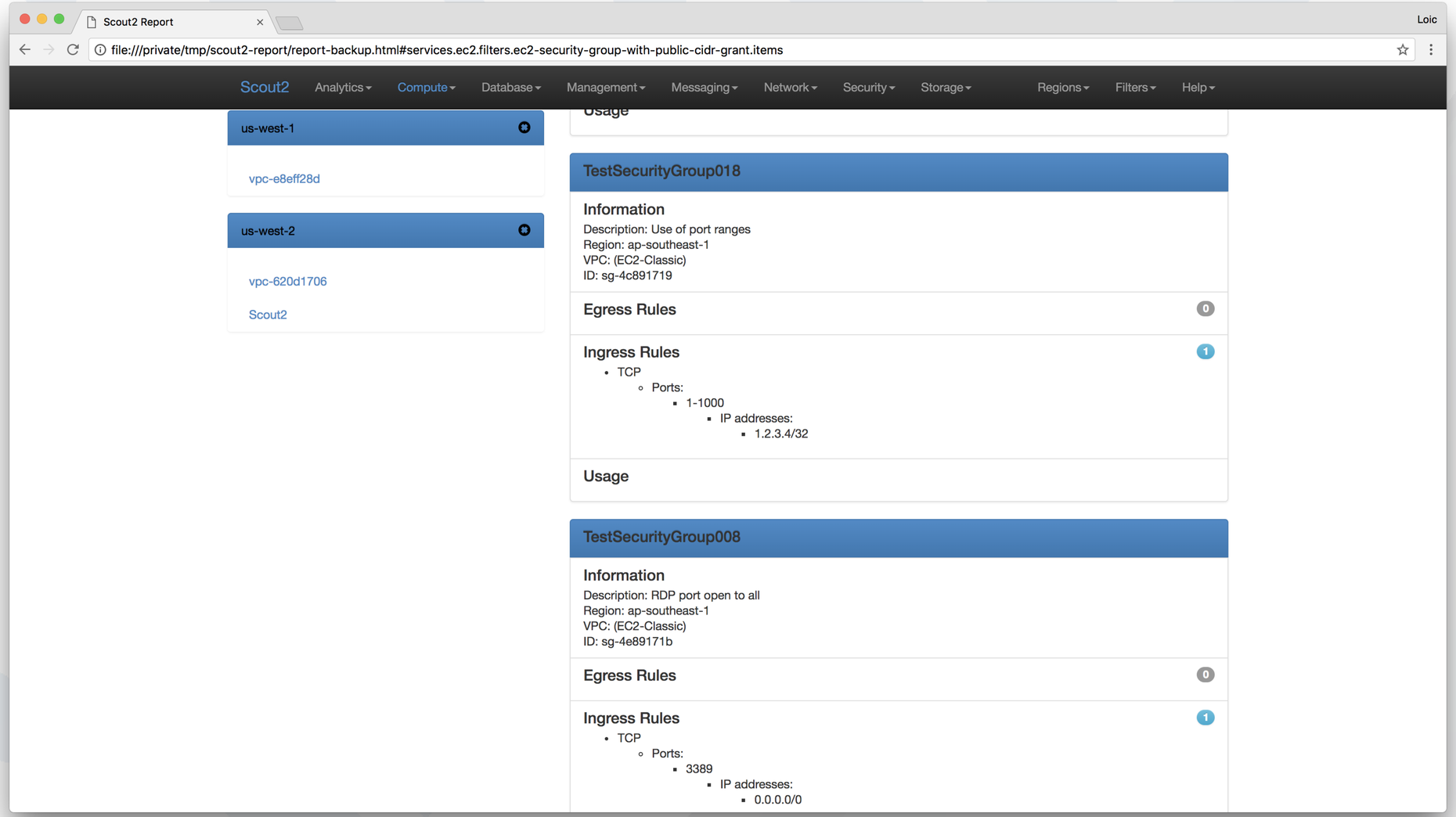
MANAGE EXCEPTIONS: LAB

Instructions provided during lab

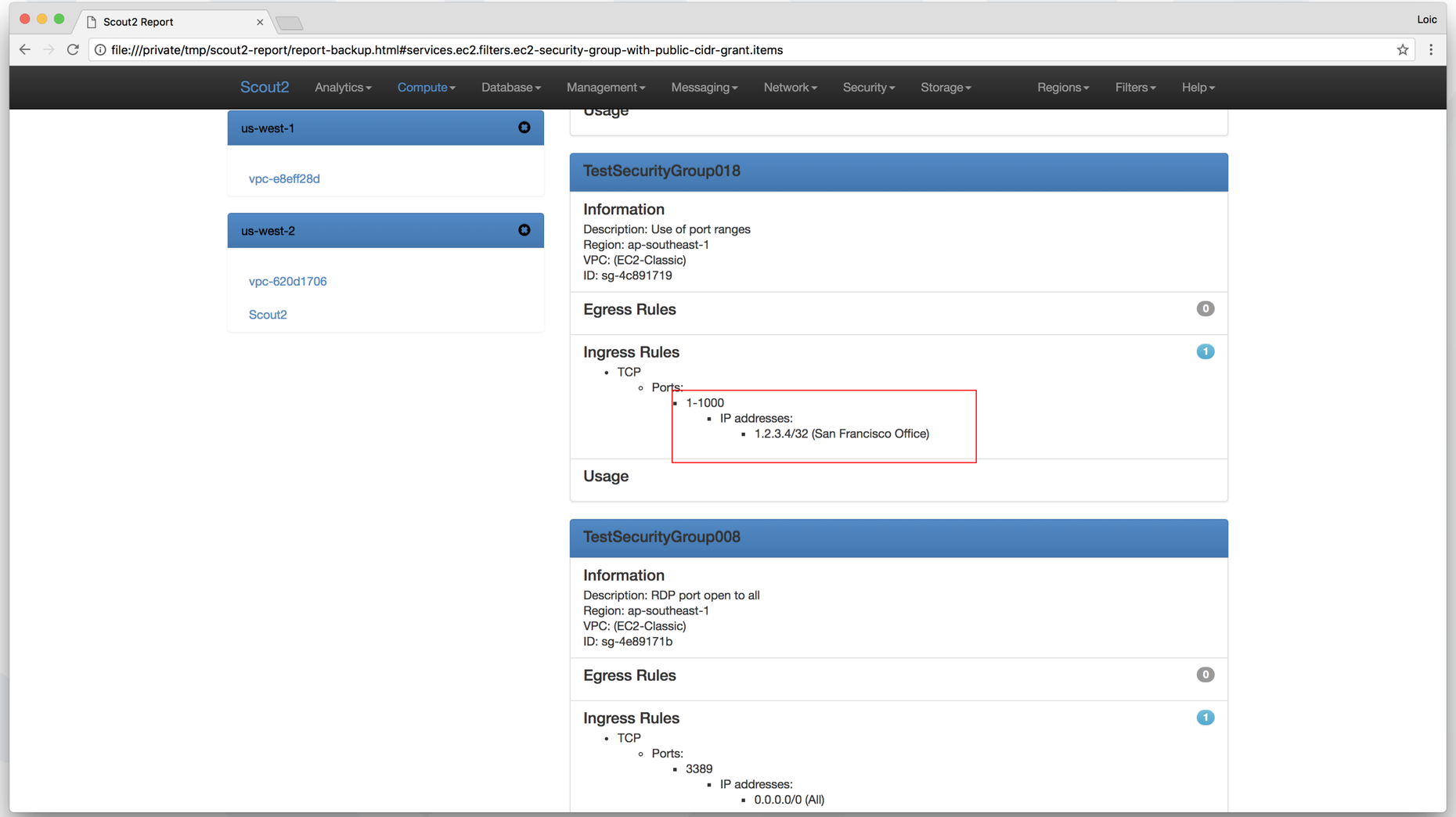
CUSTOM INPUT

USE A LIST OF KNOWN IP ADDRESSES

USE A LIST OF KNOWN IP ADDRESSES: EXPECTATION



USE A LIST OF KNOWN IP ADDRESSES: EXPECTATION



USE A LIST OF KNOWN IP ADDRESSES: GUIDE

1. Create a list of known IP addresses with the *aws_recipes_create_ip_ranges.py* tool
    1. Interactive mode

       awsrecipes_create_ip_ranges.py --interactive --debug
    2. From a CSV file

       TODO
    3. From any AWS account

       awsrecipes_create_ip_ranges.py --profile <>
2. Run Scout2 with the list of known IP addresses

   Scout2 --profile <> --local –ip-ranges ip-ranges-file.json
3. Observe subnet and host names displayed in the security group detailed grants view

USE A LIST OF KNOWN IP ADDRESSES: UNDER THE HOOD


```
{
    "createDate": "2018-03-10-10-36-46",
    "prefixes": [
        {
            "ip_prefix": "5.6.7.8",
            "name": "New York Office"
        },
        {
            "ip_prefix": "1.2.3.4/32",
            "name": "San Francisco Office"
        }
    ]
}
```

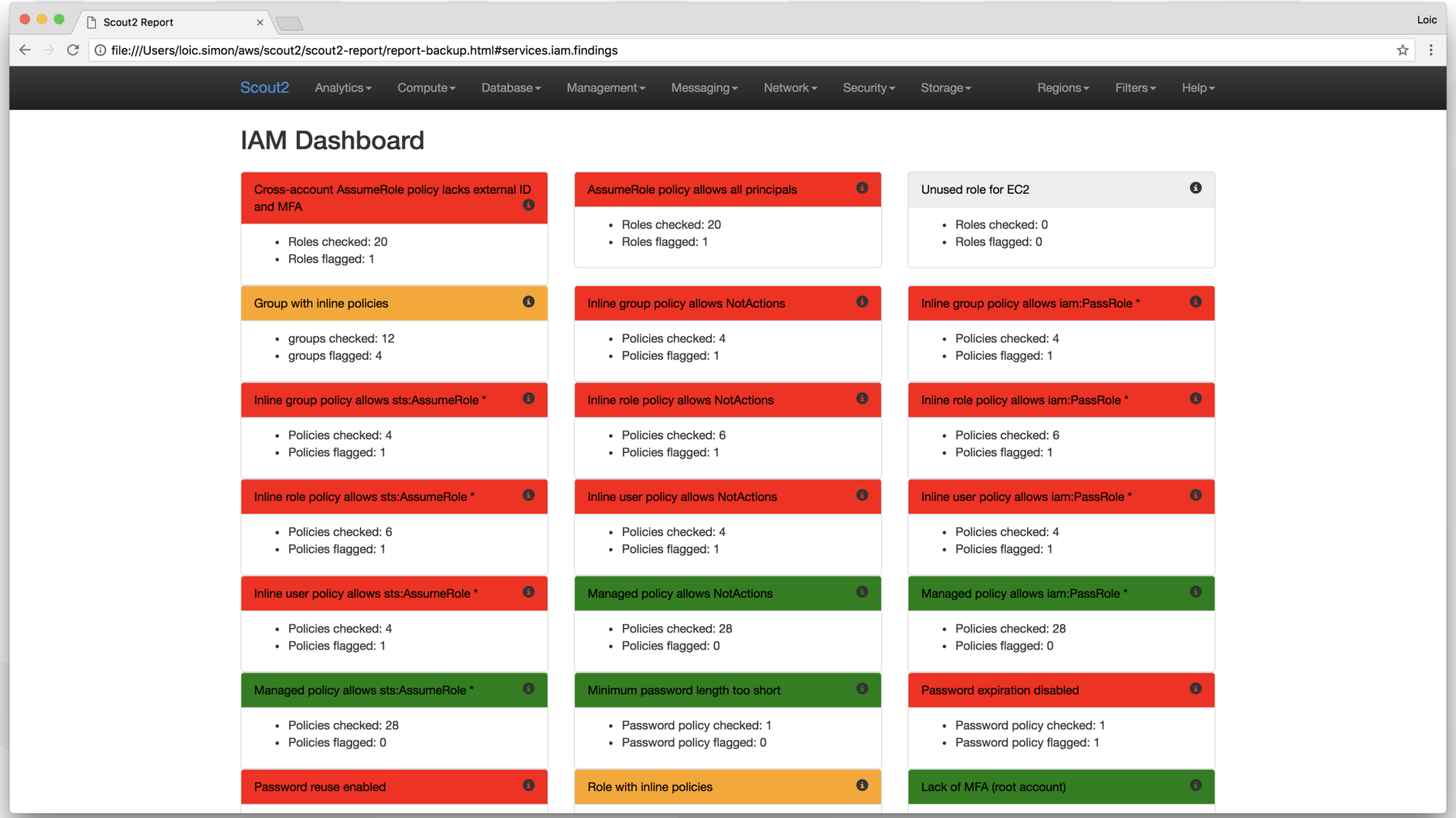
USE A LIST OF KNOWN IP ADDRESSES: LAB

Instructions provided during lab

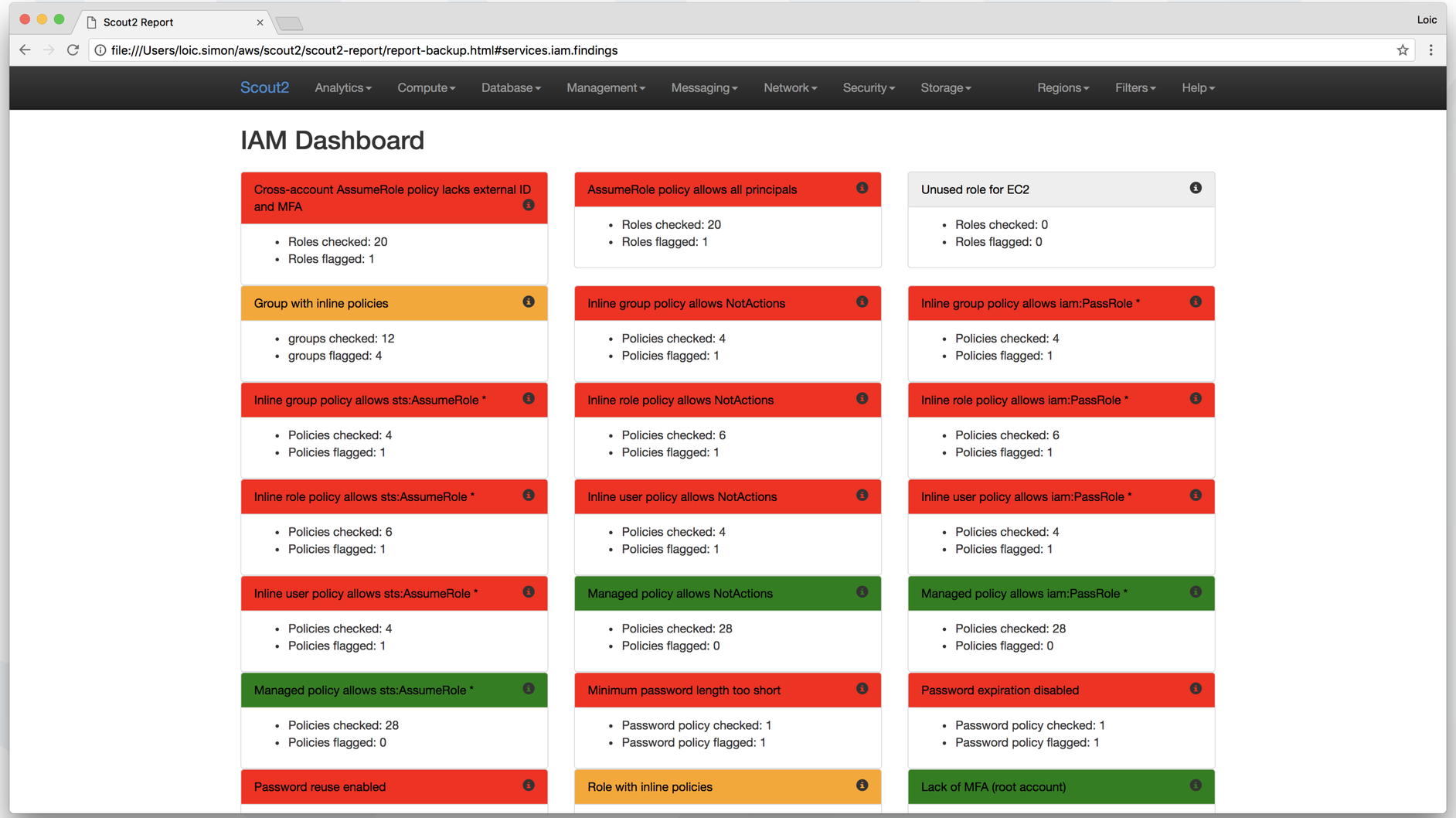
CUSTOM RULESETS

UPDATE OF A PARAMETERIZED RULE

UPDATE OF A PARAMETERIZED RULE: EXPECTATIONS



UPDATE OF A PARAMETERIZED RULE: EXPECTATIONS



UPDATE OF A PARAMETERIZED RULE: GUIDE

1. Start the ruleset generator

   Scout2RulesGenerator --ruleset-name demo

2. Find a parameterized rule

   1. For example, Security > IAM > “Minimum password length too short”

3. Modify the parameter value

4. Click on the “Generate Ruleset” link in the top right corner

5. Run Scout2 with the new ruleset

   ./Scout2.py --profile <> --local --ruleset ~/Downloads/demo-ruleset.json

6. Confirm results have been updated accordingly

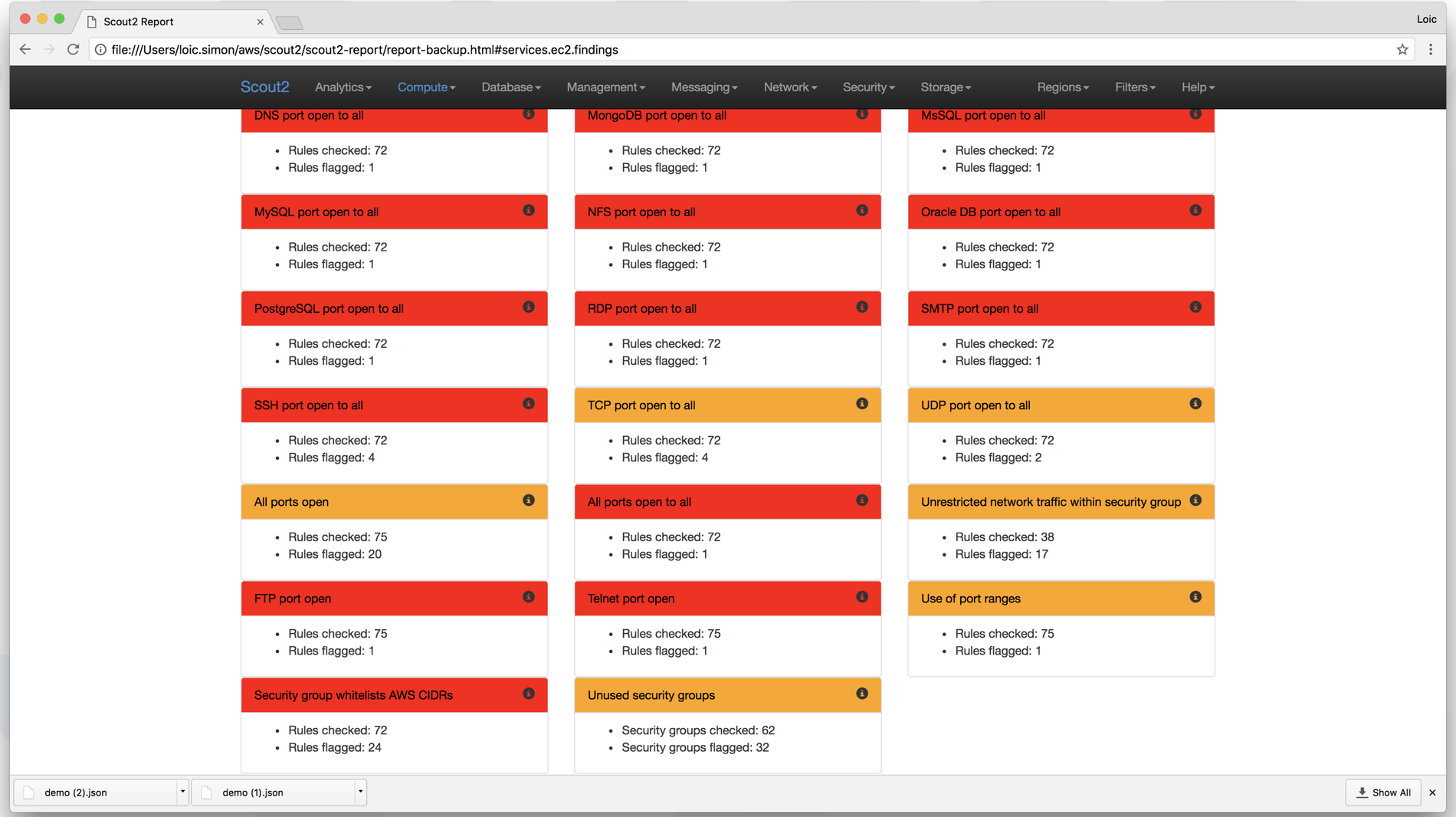
UPDATE OF A PARAMETERIZED RULE: UNDER THE HOOD

```
{
    "about": "Short brief about the ruleset.",
    "rules": {
        ...
        "iam-password-policy-minimum-length.json": [
            {
                "args": [
                    "8"
                ],
                "enabled": true,
                "level": "danger"
            }
        ],
        ....
    }
}
```

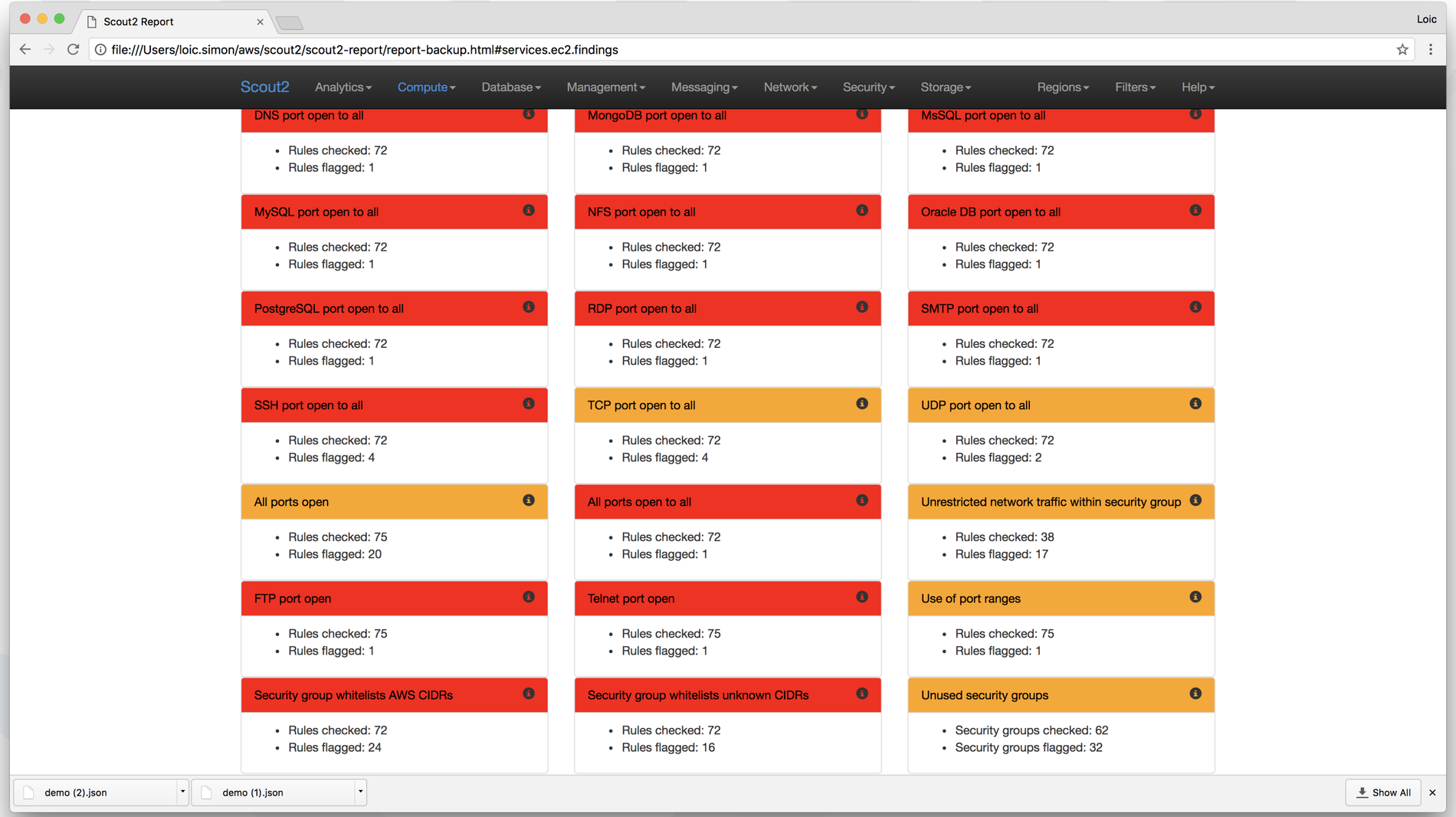
CUSTOM RULESETS

ENABLE AN OPTIONAL RULE

ENABLE AN OPTIONAL RULE: EXPECTATIONS



ENABLE AN OPTIONAL RULE: EXPECTATIONS



ENABLE AN OPTIONAL RULE: GUIDE

1. Start the ruleset generator

   ./Scout2RulesGenerator.py --ruleset-name demo

2. Find an optional rule (*i.e.* enabled check box not selected)

   1. For example, Compute > EC2 > “Security group whitelists unknown CIDRs”

3. Enable the rule (*i.e.* tick the check box)

4. Click on the “Generate Ruleset” link in the top right corner and save the file

5. Run Scout2 with the new ruleset

   ./Scout2.py --profile <> --local --ruleset demo.json

6. Confirm a new rule exists in the corresponding service’s dashboard

ENABLE AN OPTIONAL RULE: LAB

Instructions provided during lab

CUSTOM RULES

REUSE A PARAMETERIZED RULE

REUSE A PARAMETERIZED RULE: GUIDE

1. Open the ruleset file in a text editor

   <scout2_dir>/AWSScout2/rules/data/rulesets/default.json

2. Identify the parameterized rule reference

3. Create a new dictionary entry with a payload

```
{
        “args”: [
        “Arg0-Value”,
        …
        “ArgN-Value”
        ],
        “enabled”: true,
        “level”: “danger”
}
```

4. Run Scout2 with the new ruleset

   ./Scout2.py --profile scout2lab --local --ruleset scout2-labXXX.json

REUSE A PARAMETERIZED RULE: LAB

Instructions provided during lab

CUSTOM RULES

CREATE A NEW RULE

CREATE A NEW RULE: GUIDE

1. Create a new rule file and open it in a text editor

   <scout2_dir>/AWSScout2/rules/data/findings/demo.json

2. Create the new rule definition respecting the following schema

```
{
        "description": ”Name of finding in dashboard",
        "rationale": "Why this rule matters”,
        "path": ”service.path_to_resources.id",
        "dashboard_name": "Name of resource in dashboard",
        "conditions": [ ”operator-and-or",
                [ "path_to_test_value", "test_case", ”expected_value" ],
                [ "path_to_test_value", "test_case", ”expected_value" ]
        ],
        "id_suffix": ”path_to_highlighted_value”
}
```

3. Start the Ruleset Generator, identify the new rule, enable it, and generate a new ruleset

4. Run Scout2 with the new ruleset

CREATE A NEW RULE: LAB

Instructions provided during lab

CUSTOM RULES

CREATE A NEW PARAMETERIZED RULE

CREATE A NEW PARAMETERIZED RULE: GUIDE

Similar to the creation of a new rule, but requires additional information

1. **Name of the arguments: to be displayed in the ruleset generator**
2. **Key: unique identifier for the finding (defaults to the name of the file defining the rule)**
3. **Argument values must be referenced using the _ARG_N_ notation in any rule attribute**

CREATE A NEW RULE: GUIDE

1. Create a new rule file and open it in a text editor

   <scout2_dir>/AWSScout2/rules/data/findings/parameterized-demo.json

2. Create the new rule definition respecting the following schema

```
{
        "description": "Name of finding in dashboard",
        "arg_names": [ "arg_name_0", …, "arg_name_N" ],
        "key": "paramaterized-demo-_ARG_0_",
        "rationale": "Why this rule matters",
        "path": "service.path_to_resources.id",
        "dashboard_name": "Name of resource in dashboard",
        "conditions": [ ”operator-and-or",
                [ "path_to_test_value", "test_case", "_ARG_0_" ],
                [ "path_to_test_value", "test_case", "expected_value" ]
        ],
        "id_suffix": "path_to_highlighted_value"
}
```

CREATE A NEW RULE: GUIDE

1. Create a new rule file and open it in a text editor

   <scout2_dir>/AWSScout2/rules/data/findings/parameterized-demo.json

2. Create the new rule definition respecting the following schema

```
{
        "description": "Name of finding in dashboard",
        "arg_names": [ "arg_name_0", …, "arg_name_N" ],
        "key": "paramaterized-demo-_ARG_0_",
        "rationale": "Why this rule matters",
        "path": "service.path_to_resources.id",
        "dashboard_name": "Name of resource in dashboard",
        "conditions": [ ”operator-and-or",
                [ "path_to_test_value", "test_case", "_ARG_0_" ],
                [ "path_to_test_value", "test_case", "expected_value" ]
        ],
        "id_suffix": "path_to_highlighted_value"
}
```

CREATE A NEW RULE: GUIDE

1. Create a new rule file and open it in a text editor

   <scout2_dir>/AWSScout2/rules/data/findings/parameterized-demo.json

2. Create the new rule definition respecting the following schema

```
{
        "description": "Name of finding in dashboard",
        "arg_names": [ "arg_name_0", …, "arg_name_N" ],
        "key": "paramaterized-demo-_ARG_0_",
        "rationale": "Why this rule matters",
        "path": "service.path_to_resources.id",
        "dashboard_name": "Name of resource in dashboard",
        "conditions": [ ”operator-and-or",
                [ "path_to_test_value", "test_case", "_ARG_0_" ],
                [ "path_to_test_value", "test_case", "expected_value" ]
        ],
        "id_suffix": "path_to_highlighted_value"
}
```

CREATE A NEW RULE: GUIDE

1. Create a new rule file and open it in a text editor

   <scout2_dir>/AWSScout2/rules/data/findings/parameterized-demo.json

2. Create the new rule definition respecting the previous schema

3. Start the Ruleset Generator, identify the new rule, enable it, and generate a new ruleset

4. Run Scout2 with the new ruleset

CREATE A NEW PARAMETERIZED RULE: LAB

Instructions provided during lab

MIS|TI™ PRESENTS

InfoSec World

Conference & Expo 2018

THANK YOU

PLEASE FILL OUT YOUR EVALUATIONS!

Loic Simon

Director of Product Security @ Tanium

@l01cd3v