Playbooks for Auto-remediation in AWS

Playbooks are called by the Lambda function and contain the Python scripts that remediate misconfigurations in your environment. They are presented here for reference only.

Disclaimer

By using this software and associated documentation files (the “Software”) you hereby agree and understand that:

1. The use of the Software is free of charge and may only be used by Wiz customers for its internal purposes.
2. The Software should not be distributed to third parties.
3. The Software is not part of Wiz’s Services and is not subject to your company’s services agreement with Wiz.
4. THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL WIZ BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE USE OF THIS SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

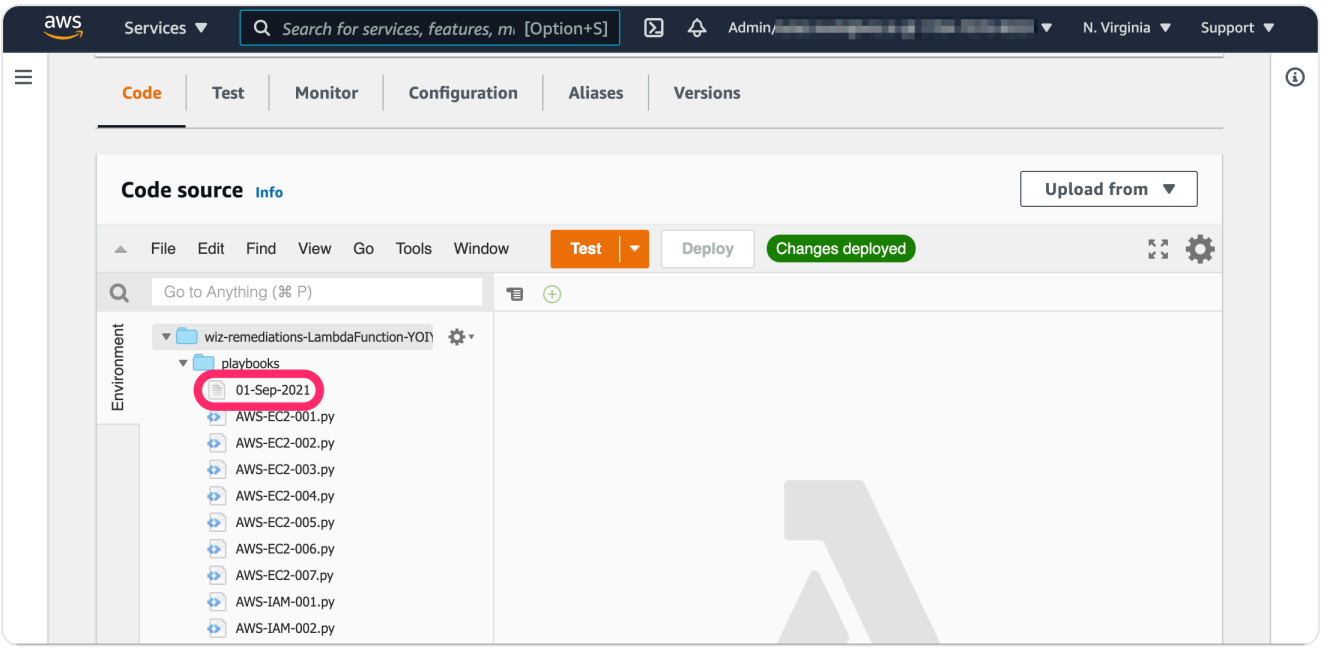
Latest playbook pack

The playbook pack was updated on November 16th, 2022.

There are fewer playbooks than supported Cloud Configuration Rules because similar CCRs (e.g. everything related to password policies) are handled by a single playbook. See the mapping in index_parser.py.

To view your current playbook pack release:

1. Navigate to the AWS Lambda service in the region where you deployed the automated remediation infrastructure, then open the remediation Lambda function.
2. In the Code source section, under the Code tab, expand the root and playbooks folders.



3. The first file in the list will have the date of the current playbook pack.

Update playbooks

Wiz is continuously updating this playbook repository, and in the future will package them into releases hosted in a Wiz GitHub repository. If a new release is available, the latest package will always be equivalent to the release with the latest date.

> ⚠ Updating the built-in playbooks deletes any custom playbooks. Be sure to save any custom playbooks somewhere safe before updating.

To update your Lambda package with the latest Wiz playbooks see the guide on updating in AWS.

EC2 Playbooks

EC2-001 EBS Volume Does Not Have Recent Snapshots

Python

```
"""
Remediate Wiz Issue:
```

```
AWS:EC2-001 EBS Volume Does Not Have Recent Snapshot

Description:

Elastic block store (EBS) volumes have no snapshots in recent, 15-day
history. These are needed to retain
critical system data, security logs, and system state.

EBS volume snapshots are an important tool to record system state, security
log data, and critical system
data at various points in your EC2 instance lifecycle. Snapshots provide
point-in-time recovery and review
capability that is necessary for many operational and security practices.

Required Permissions:

- ec2:CreateSnapshot
- ec2:DescribeSnapshots

Sample IAM Policy:

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:DescribeSnapshots"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import boto3
from botocore.exceptions import ClientError
from datetime import datetime
from datetime import date
```

EC2-003 Public AMI Detected

Python

```
"""
Remediate Wiz Issue:
```

```
AWS:EC2-003 Public AMI Detected

Description:

Publicly shared AMIs pose a risk in exposing sensitive data.

Required Permissions:

- ec2:DescribeImageAttribute
- ec2:ModifyImageAttribute

Sample IAM Policy:

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImageAttribute",
                "ec2:ModifyImageAttribute"
            ],
            "Resource": "*"
        }
    ]
}
"""

import boto3
from botocore.exceptions import ClientError

def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
  """

  image_id = alert['resource_id']
  region   = alert['region']

  ec2  = session.client('ec2', region_name=region)

  try:
    attribute = ec2.describe_image_attribute(ImageId=image_id,
```

EC2-004 AWS Security Groups allow internet traffic to SSH port (22)

Python

```
"""
Remediate Wiz Issue:
```

```
AWS:EC2-004 AWS Security Groups allow internet traffic to SSH port (22)

Description:

Global permission to access the TCP port 22 services (SSH) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. SSH is tcp-22)
admin_port_list  = [ 'tcp-22' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

EC2-005 AWS Security Groups allow internet traffic to Telnet port (23)

Python

English

```
"""
Remediate Wiz Issue:

AWS:EC2-005 AWS Security Groups allow internet traffic to Telnet port (23)

Description:

Global permission to access the TCP port 23 services (Telnet) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. Telnet is tcp-23)
admin_port_list  = [ 'tcp-23' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
```

EC2-006 AWS Security Groups allow internet traffic to RDP port (3389)

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-006 AWS Security Groups allow internet traffic to RDP port (3389)

Description:

Global permission to access the TCP port 3389 services (RDP) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. RDP is tcp-3389)
admin_port_list  = [ 'tcp-3389' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
```

English

EC2-007 AWS Security Groups allow internet traffic to DNS port (53)

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-007 AWS Security Groups allow internet traffic to DNS port (53)

Description:

Global permission to access the TCP port 53 services (DNS) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-53' , 'udp-53' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-008 Unrestricted VNC Listener Access (5500)

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-008 Unrestricted VNC Listener Access (5500)

Description:

Global permission to access the TCP port 5500 services (VNC Listener) should
not be allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-5500' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-009 Unrestricted VNC Server Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-009 Unrestricted VNC Server Access

Description:

Global permission to access the TCP port 5800/5900 services (VNC Server)
should not be allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-5800' , 'tcp-5900' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-010 Unrestricted RPC Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-010 Unrestricted RPC Access

Description:

Global permission to access the TCP port 135 services (RPC) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-135' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-011 Unrestricted SMB Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-011 Unrestricted SMB Access

Description:

Global permission to access the TCP port 445 services (SMB) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-445' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-012 Unrestricted CIFS Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-012 Unrestricted CIFS Access

Description:

Global permission to access the TCP port 445 services (CIFS) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-445' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-013 Unrestricted FTP Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-013 Unrestricted FTP Access

Description:

Global permission to access the TCP port 21 services (FTP) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-21' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-014 Unrestricted FTP-Data Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-014 Unrestricted FTP-Data Access

Description:

Global permission to access the TCP port 20 services (FTP-Data) should not
be allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-20' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-015 Unrestricted MSQL Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-015 Unrestricted MSQL Access

Description:

Global permission to access the TCP port 4333 services (MSQL) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-4333' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-016 Unrestricted MySQL Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-016 Unrestricted MySQL Access

Description:

Global permission to access the TCP port 3306 services (MySQL) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-3306' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-017 Unrestricted NetBIOS Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-017 Unrestricted NetBIOS Access

Description:

Global permission to access the UDP port 137/138 services (NetBIOS) should
not be allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'udp-137' , 'udp-138' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-018 Unrestricted PostgreSQL Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-018 Unrestricted PostgreSQL Access

Description:

Global permission to access the TCP port 5432 services (PostgreSQL) should
not be allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-5432' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-019 Unrestricted SQLServer TCP Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-019 Unrestricted SQLServer TCP Access

Description:

Global permission to access the TCP port 1433 services (SQLServer) should
not be allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-1433' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-020 Unrestricted SQLServer UDP Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-020 Unrestricted SQLServer UDP Access

Description:

Global permission to access the UDP port 1434 services (SQLServer) should
not be allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'udp-1434' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-021 Unrestricted SMTP Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-021 Unrestricted SMTP Access

Description:

Global permission to access the TCP port 25 services (SMTP) should not be
allowed in a security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'tcp-25' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-022 Unrestricted ICMP Access

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-022 Unrestricted ICMP Access

Description:

Global permission to access the ICMP services should not be allowed in a
security group.

**Note: THE Remediation will be executed if the service is found within a
port range.

Required Permissions:
- ec2:DescribeSecurityGroups
- ec2:RevokeSecurityGroupIngress

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:RevokeSecurityGroupIngress"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import json
import re
import boto3
from botocore.exceptions import ClientError

## admin_port_list has a join of protocol and port (e.g. DNS is tcp-53 or
udp-53)
admin_port_list  = [ 'icmp.-1', 'icmpv6.-1' ]
global_cidr_list = [ '0.0.0.0/0', '::/0' ]

def remediate(session, alert, lambda_context):
  """
```

English

EC2-029 EC2 instances should use instance Metadata Service Version 2 (IMDSv2)

Python

```
"""
Remediate Wiz Issue:

AWS:EC2-029 EC2 instances should use instance Metadata Service Version 2
(IMDSv2)

Description:

The rule checks whether the EC2 instance metadata version is configured with
Metadata Service Version 2 (IMDSv2),
which is considered more secured than IMDSv1. This rule fails if
MetadataOptions.HttpTokens is set to optional.
This playbook changes MetadataOptions.HttpTokens to required.

Required Permissions:
- ec2:DescribeInstances
- ec2:ModifyInstanceMetadataOptions

Sample IAM Policy:
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2Permissions",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:ModifyInstanceMetadataOptions"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
"""

import boto3
from botocore.exceptions import ClientError
from datetime import datetime
from datetime import date

def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
  """

  instance_id = alert['resource_id'].rsplit('/', 1)[-1]
```

English

IAM playbooks

IAM-001 Enforce AWS account best practices password policy

Python

```
"""
Remediate Wiz Issue:

AWS-IAM-001 Enforce AWS account best practices password policy

>>>> Set dry_run to False if you want to enable this remediation playbook,
instead of just reviewing its "what if" changes.

The remediation will set the following password policy attributes, even if
an Issue was created for one of these:
- MinimumPasswordLength: 14 or more **
- RequireLowercaseCharacters: True **
- RequireUppercaseCharacters: True **
- RequireNumbers: True **
- RequireSymbols: True **
- MaxPasswordAge: 90 or less **
- AllowUsersToChangePassword: True <<< Not checking right now.
- PasswordReusePrevention: 0 **
If there's an existing password policy, the remediation script will:
- take the higher number for MinimumPasswordLength and
PasswordReusePrevention
- take the lower number for MaxPasswordAge

Required Permissions:
- iam:UpdateAccountPasswordPolicy
Sample IAM Policy:
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1507759700000",
      "Effect": "Allow",
      "Action": [
        "iam:UpdateAccountPasswordPolicy"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
"""

# password policy will only run ig dry run is set to False
dry_run = True
```

English

```python
def remediate(session, issue, lambda_context):
    ....
```

IAM-003 AWS IAM policy allows full administrative privileges

Python

```
"""
Remediate Wiz Issue:

AWS-IAM-003 AWS IAM policy allows full administrative privileges

Description:

The best practice is not to create a custom IAM policy with full
administrator access.
If administrator access granting is necessary, then use the default
"AdministratorAccess" policy

Required permissions:

- iam:CreatePolicyVersion

Sample IAM Policy:

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1507759700000",
      "Effect": "Allow",
      "Action": [
        "iam:CreatePolicyVersion"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}

Remediation result:
- Modify the custom policy with a placeholder policy that has minimum needed
access.
For more information on why "sts:getCallerIdentity" is set for the minimum
permission (you have to put something...), visit:
https://docs.aws.amazon.com/STS/latest/APIReference/API_GetCallerIdentity.html

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RemediatedAdminPolicy",
```

"Effect": "Allow"

IR playbooks

AWS-IR-001 SSH brute-force attempts on a publicly exposed VM with lateral movement to admin that enables password authentication using weak password

Python

```
"""
Remediate Wiz Issue:

AWS:IR-001 SSH brute-force attempts on a publicly exposed VM with lateral
movement to admin that enables password authentication using weak password

Description:
This playbook isolates an EC2 instance primarily by:

1. Removing any associated IAM roles to restrict the permissions
2. Removing all security groups and associating a specified restricted group
which can be done in 2 ways:
2a You can specify an existing security group by name and the playbook will
apply this to the instance (note we do not create the group)
2b If you are using AWS Firewall Manager you can specify a tag name and
value to be associated with

Optional isolation actions
Additionally this playbook can run the following actions after the primary
method described above:
- Stop instance
- Take snapshop

Supplying optional actions to this playbook via Wiz
When creating a new action you can add the following `remediation_actions`
key to the request body:
{
   "remediation_actions":  [
      { "stop_instance":"false" },
      { "take_snapshot":"false" }
   ],
  "trigger": {
    "source": "{{triggerSource}}",
    ....

When you run the action you can choose to override these settings with
true|false as required.

Parameters
To configure this playbook the following SSM parameters need to be set
/playbook_lookup/AWS-IR-001/params/isolation_type   - Instance isolation
```

English

```
type, valid values "security_group" |"firewall_manager"
/playbook_lookup/AWS-IR-001/params/fwm_tag_name     - Tag name used by AWS
Firewall Maanger, only required when isolation_type=firewall_manager
/playbook_lookup/AWS-IR-001/params/fwm_tag_value    - Tag value used by AWS
Firewall Maanger, only required when isolation_type=firewall_manager
/playbook_lookup/AWS-IR-001/params/sg_group_name    - Security group name
```

RDS playbooks

RDS-002 RDS Database Publicly Accessible

Python

```
"""
Remediate Wiz Issue:

AWS-RDS-002 RDS Database Publicly Accessible

Description:

RDS database instances should not be accessible from the public internet.
RDS databases should normally be privately accessible and only from within
your VPC.

Required Permissions:

- rds:DescribeDBInstances
- rds:ModifyDBInstance

Sample IAM Policy:

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RDSPermissions",
      "Action": [
        "rds:DescribeDBInstances",
        "rds:ModifyDBInstance"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
"""

import boto3
from botocore.exceptions import ClientError
```

```python
def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
  """

  resource_id = alert['resource_id']
  region      = alert['region']

  rds = session.client('rds', region_name=region)
```

S3 playbooks

S3-001 AWS S3 buckets are accessible to public

Python

```python
"""
Remediate Wiz Issue:

AWS-S3-001 AWS S3 buckets are accessible to public

Description:

The remediation will put "Block all public access" on the S3 bucket
permissions settings.

Required Permissions:
- s3:PutBucketPublicAccessBlock

Sample IAM Policy:

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "S3Permissions",
      "Action": [
        "s3:PutBucketPublicAccessBlock"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
"""

import boto3
from botocore.exceptions import ClientError


def remediate(session, alert, lambda_context):
```

```
  """
  Main Function invoked by index_parser.py
  """
  bucket_id      = alert['resource_name']
  region         = alert['region']
  cloud_account  = alert['subscription']['id']

  s3 = session.client('s3', region_name=region)

  try:
    bucket_acl = s3.get_bucket_acl(Bucket=bucket_id)
  except ClientError as e:
```

S3-002 AWS S3 bucket has global view ACL permissions enabled

Python

```
"""
Remediate Wiz Issue:

AWS-S3-002 AWS S3 bucket has global view ACL permissions enabled

Description:

This remediation will remove the existence of Global ACL permissions on S3
Bucket for the All Users and Authenticated Users groups.

Required Permissions:

- s3:GetBucketAcl
- s3:PutBucketAcl

Sample IAM Policy:

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "S3Permissions",
      "Action": [
        "s3:GetBucketAcl",
        "s3:PutBucketAcl"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
"""

import boto3
```

```python
from botocore.exceptions import ClientError


def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
  """

  bucket_name  = alert['resource_name']
  region       = alert['region']
  cloud_account  = alert['subscription']['id']

  s3 = session.client('s3', region_name=region)
```

S3-003 S3 Object Versioning Not Enabled

Python

```
"""
Remediate Wiz Issue:

AWS-S3-003 S3 Object Versioning Not Enabled

Description:

S3 Object Versioning is key in protecting your data within a bucket.
Once you enable Object Versioning, you cannot remove it; you can suspend
Object Versioning at any time on a bucket if you do not wish for it to
persist.

Required Permissions:

- s3:PutBucketVersioning

Sample IAM Policy:

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "S3Permissions",
      "Action": [
        "s3:PutBucketVersioning"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
"""

import boto3
```

```python
from botocore.exceptions import ClientError


def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
  """

  bucket = alert['resource_id']
  region = alert['region']

  s3 = session.client('s3', region_name=region)
```

S3-009 S3 Bucket Logging Disabled

Python

```
"""
Remediate Wiz Issue:

AWS:S3-009 S3 Bucket Logging Disabled

Description:

S3 access logging provides a way to get details about S3 bucket activity.
By default, AWS does not enable logging for S3 buckets.

Required Permissions:

- sts:GetCallerIdentity
- s3:CreateBucket
- s3:GetBucketLogging
- s3:PutBucketLogging

Sample IAM Policy:

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "STSPermissions",
      "Action": [
        "sts:GetCallerIdentity"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "S3Permissions",
      "Action": [
        "s3:CreateBucket",
        "s3:GetBucketLogging",
```

```
        "s3:PutBucketLogging"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
"""

import boto3
from botocore.exceptions import ClientError
```

Redshift playbooks

REDSHIFT-001 Redshift Cluster Publicly Accessible

Python

```
"""
Remediate Wiz Issue:

AWS-REDSHIFT-001 Redshift Cluster Publicly Accessible

Description:

It is a recommended best practice that Redshift cluster nodes are not
accessible to public internet, and outside of your VPC.
Redshift databases should only be privately accessible and from within your
VPC.

Required Permissions:

- redshift:DescribeClusters
- redshift:ModifyCluster

Sample IAM Policy:

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "RedshiftPermissions",
      "Action": [
        "redshift:DescribeClusters",
        "redshift:ModifyCluster"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
```

```
}
"""

import boto3
from botocore.exceptions import ClientError


def remediate(session, alert, lambda_context):
  """
  Main Function invoked by index_parser.py
  """

  cluster_id = alert['resource_id']
  region = alert['region']
```

Miscellaneous playbooks

Test event

Python

```
import boto3

"""

Use the following Test Event to test Lambda:

{
  "Records": [
    {
      "messageId": "12341234-abcd-abcd-abcd-123412341234",
      "receiptHandle": "MessageReceiptHandle",
      "body": "{\"trigger\": {\"source\": \"Action\", \"type\": \"Action\",
\"ruleId\": \"1\", \"ruleName\": \"1\"}, \"issue\": {\"id\": \"ee878f26-
bcd8-4a4d-bba1-f9684a0488cf\", \"status\": \"ACTIVE\", \"severity\":
\"HIGH\", \"created\": \"2021-07-01M01:01:01Z\", \"projects\": null},
\"resource\": {\"id\": \"test-resource\", \"name\": \"test-resource\",
\"type\": \"security_group\", \"cloudPlatform\": \"aws\",
\"subscriptionId\": \"123456789012\", \"subscriptionName\": \"TestAccount\",
\"region\": \"us-east-2\", \"status\": \"ACTIVE\", \"cloudProviderURL\":
\"someurl.com\"}, \"control\": {\"id\": \"TEST-001\", \"name\": \"Test
Control\", \"description\": \"ssh\", \"severity\": \"HIGH\",
\"sourceCloudConfigurationRuleId\": \"TEST-001\",
\"sourceCloudConfigurationRuleName\": \"Test Control\"}}",
      "attributes": {
        "ApproximateReceiveCount": "1",
        "SentTimestamp": "1523233000000",
        "SenderId": "123456789012",
        "ApproximateFirstReceiveTimestamp": "1523233000001"
      },
```

```
      "messageAttributes": {},
      "md5OfBody": "7b270e59b47ff90a553787f90a553787",
      "eventSource": "aws:sqs",
      "eventSourceARN": "arn:aws:sqs:us-west-2:123456789012:WizIssuesQueue",
      "awsRegion": "us-east-2"
    }
  ]
}

To test the Lambda function, replace "123456789012" in the subscriptionId
field, with your AWS account ID.

Note: The "body" field must be a single line.

"""

def remediate(session, alert, lambda_context):
    print('I am a TEST playbook and I am now executing!! Yeppi.')
```

Updated about 2 months ago

← Troubleshoot Auto-remediation in AWS

Add Playbooks for Auto-remediation in AWS →