6 Best Practices for Amazon Security Groups

OPSWAT®

Introduction

Security groups are Amazon's own version of firewalls, offering a simplified approach and easier configuration. Every running server needs to have at least one security group attached to it.

The good news is that we can add and remove security groups on the fly, and we can change the configurations inside and all of the changes will take effect immediately. If an instance has multiple security groups, it has the sum of all the rules in the various groups.

With security groups, Amazon follows the best practice of emphasizing whitelisting. When it comes to firewalls and security in general, it is better to use a whitelist approach to security rules instead of a blacklist. Blacklists usually have to block endless possibilities, and hackers are on the lookout for corner cases where system admins forgot to deny access.

However, not paying close attention to security group configuration results in messy security groups, full of unidentifiable IP addresses and hitting the limit of maximum entries. Poorly configured security groups are very hard to audit and clean up.

These best practices can help organizations have stable, long-running security group configurations that are easy to maintain and extend:

1. Only Grant the Minimum Level of Access
2. Always Add Descriptions for Each Rule
3. Divide Security Groups Based on Scope
4. Be Afraid of Pinging
5. Leverage Security Groups and Recursivity
6. Validate IP Sources

1. Only Grant the Minimum Level of Access

Servers in cloud infrastructure are usually very specialized, following the principle of separation of concerns. Common server types include web servers, cache servers, database servers, and load balancers.

But each of them requires one specific port to be open; web servers will connect on 443 and perhaps 80, Redis cache uses 6379, and databases also have default well-known ports. If services are only used in the local VPC, then it is a good idea not to expose any ports publicly. For example, it is possible to expose publicly only the web server, while the cache and database servers remain closed from external connections.

Also, if external connections are necessary to such services, it is a good practice to switch the ports to higher nonstandard ports in order to add an extra level of obscurity and stop port scanners from easily picking up the open connections.

Regardless of the chosen strategy, always make sure that only the minimum level of access is granted.

If the database server is only accessed by the web server, and only on port 3306 or whichever port is designated, make sure that the only IP that has access to that machine is the web server IP on the database port. To grant direct access for engineers, the web server can be used as a tunnel for the database service.

Never allow all inbound traffic or all outbound traffic on all ports in production environments (0.0.0.0/0). This might be dangerous even in staging environments if separate VPCs are not created from staging, because a computer in a network might compromise all other computers even though all other security group rules are properly configured.

Note that if there is more than one rule for a specific port, the most permissive rule is applied. For example, if a rule allows access to port 22 from a specific IP address and another rule allows access to port 22 from everywhere, the more permissive rule applies and everyone has access to TCP port 22.

2. Always Add Descriptions for Each Rule

In summer 2017, Amazon offered the option to add descriptions to the rules in security groups. A description can be up to 255 characters in length, and the allowed characters are a-z, A-Z, 0-9, spaces, and ._-:/()#,@ []+=;{}!$*. Although this allows for a lot of flexibility in writing descriptions, it is best to just add simple descriptions like “Monitoring server” or “John’s home IP”.

There is a limit of 50 inbound and outbound rules per security group. This number is usually very easy to reach. There is also a default limit of 5 security groups per network interface, which also doesn’t require a lot of work to achieve.

Adding descriptions will help identify IP addresses at later stages. It serves as documentation for long IP lists, which otherwise would be just cryptic lists that need to be researched every time they get full. By simply reviewing the list of IPs and descriptions, it is simple to make decisions about what to eliminate or move to other groups.

3. Divide Security Groups Based on Scope

An organized and clear approach to having security groups is to divide the definitions based on scope.

It's a good idea to create groups like "Office IPs" if your company has multiple offices around the world or "Employee home IPs" if you allow remote work in your company. Another approach would be to add one security group called "SSH" and another group called "Database" in order to make a clean distinction between who has SSH access and who has access to the database.

This sort of approach makes working with security groups easy. Having "Employees home IPs"-style security groups also helps to avoid duplication, because an employee would have to add his or her IP to only one security group and would then have access to all the servers that are configured to be accessible by employees from home. It also applies to, for example, a web servers security group: Adding the security group with all the IPs of the web servers to the database servers would grant all of them access to the database in a simple and clean fashion. Dividing groups based on scope also makes it simple to remove servers from security groups when needed. For example, suppose one monitoring server gets infected and its access needs to be immediately revoked. If it has been granted access with a special security group only for monitoring servers, administrators can just remove the IP from that security group, and it will propagate in every place where the security group has access.

The same applies if the IP is updated. If we want to completely remove SSH access from a server, we can just remove the "SSH" security group.

4. Be Afraid of Pinging

Hackers are on the lookout for servers they can use as robots to perform DDoS attacks. One way hackers discover vulnerable servers is by pinging random IPs, seeing which ones respond, and then exploring different types of vulnerabilities in the hope that the server can be infected, remotely controlled, and added to their botnet.

Only enable server pinging if there is a very good reason to do so. In general it is good enough if monitoring systems just check the open ports – we can even use a specialized endpoint for reporting back status instead of pinging.

If it is absolutely necessary to enable pinging, make sure to narrow down the source of the ping requests as much as possible. Ping responses are generally a sign of an old or improperly configured server.

5. Leverage Security Groups and Recursivity

When designing a security group, the options available are to add an IP address or to add another security group.

Adding a rule based on a security group means that all the servers inside that security group will have access to that type of connection. This can lead to a variety of different configuration possibilities, such as allowing web servers to connect to database servers.

But it is also a valid option to add the security group to its own security group. This would mean allowing all servers to access a specific connection. For example, all database servers and backend servers can be added to a security group called “internal-db,” and that security group can have access to itself on port 3306 (MySQL). This is a simple and efficient way to allow access between machines in the same VPC without having to manage huge lists of IP addresses.

This recursive pattern can save a lot of trouble when it comes to autoscaling systems. Having an “Access-to-mysql” security group where IPs get added every time scale-up events occur would mean that sooner or later that security group would get full.

Indeed, this has some limitations, due to the fact that only Amazon servers from the same account can be managed this way. Just use this pattern to avoid managing huge lists of IPs.

6. Validate IP Sources

Inbound and outbound IPs can pile up. Sometimes it is necessary to allow different partners or contractors access to the organization's servers, along with the home devices of company employees. Those IP addresses might become infected over time, and so it is a good idea to do an audit from time to time and validate the reputation of all public IPs.

OPSWAT's Metadefender Cloud API offers IP reputation analysis that checks IP addresses using multiple IP reputation services. This can be used as an indicator to identify compromised servers that have access to your infrastructure.

In our efforts to make the internet a safer place, OPSWAT has developed a tool that goes to Amazon and automatically checks the IP addresses, performing an automated audit of some or all of the security group inbound and outbound rules. Instructions for installing and configuring can be found here. Please make sure to go to OPSWAT Portal and register to obtain a Metadefender Cloud API key before using the tool.

The GitHub repository for the tool can be found at https://github.com/opswat/mdcloud-go

The simplest way of installing the tool on a Linux environment:

```
sudo wget -q
https://github.com/OPSWAT/mdcloud-go/releases/download/1.0.0/mdcloud-go_linux_amd64 -O
/usr/local/bin/mdcloud && sudo chmod +x /usr/local/bin/mdcloud
```

Just make sure that /usr/local/bin is in your path.

Visit **this page** for a list of alternative downloads.

Before running the tool, please make sure you have a Metadefender Cloud API key. If not, please go to Metadefender.com and click the "Sign up" button.

After obtaining an API key, you need to specify it in the command line by setting the "MDCLOUD_APIKEY" environment variable, or by passing it as an argument to the tool with --apikey like so:

```
$> mdcloud --apikey <command>
```

To see possible options, run:

```
$> mdcloud --help
Metadefender Cloud API wrapper

Usage:
  mdcloud-go [command]

Available Commands:
  help        Help about any command
  sglist      List security groups IPs
  sgscan      Scan security groups using IP Scan API
  version     Print the version number of mdcloud-go

Flags:
  -a, --apikey string   apikey token (default is MDCLOUD_APIKEY env variable)
  -h, --help            help for mdcloud-go

Use "mdcloud-go [command] --help" for more information about a command.
```

In order to scan a security group, use:

```
$> mdcloud sgscan
109.103.100.28 : OK
35.162.110.83 : OK
35.165.42.65 : OK
54.183.119.45 : OK
62.231.66.135 : OK
79.114.3.225 : OK
113.161.86.113 : OK
14.161.18.101 : OK
195.56.42.178 : OK
90.174.2.6 : OK
54.67.15.72 : OK
90.174.2.50 : OK
95.76.19.167 : OK
76.14.65.6 : OK
52.12.101.151 : OK
52.12.182.222 : OK
52.12.86.29 : OK
52.12.88.107 : OK
```

This command relies on the fact that Amazon credentials are already configured in ~/.aws/credentials. The --include flag can be used to specify a comma delimited list of security groups to scan. By default all groups are scanned.

To see a list of possible security groups, use:

```
$> mdcloud sglist
109.103.100.28
35.162.110.83
35.165.42.65
54.183.119.45
62.231.66.135
79.114.3.225
113.161.86.113
14.161.18.101
195.56.42.178
90.174.2.6
54.67.15.72
90.174.2.50
95.76.19.167
76.14.65.6
```

Conclusion

Properly constructed security groups are crucial when it comes to configuring infrastructure. The best practices described in this guide are mostly common sense patterns, easy to apply and extend. Maintaining cloud infrastructure for the long term requires a well-organized approach, and one of the most important aspects of infrastructure is security. Make sure the configuration is clean, and continually use the Metadefender Cloud scanning tool to audit all IPs and maintain security.