What are the security benefits of defining cloud infrastructure as code?

There are multiple security benefits to using Infrastructure as Code. In my first project. I used Microsoft Azure to create an ELK Stack server which I used as a cloud monitoring system for the web servers (Pen-testing servers) that I had created previously in my bootcamp.

Considering the alternatives such as a traditional infrastructure which requires physical hardware, location and ongoing maintenance. I decided that using Microsoft Azure to create my infrastructure was most suitable to what I was trying to create within the given parameters, that is due to the benefits of having your infrastructure in the cloud as code. I have listed some of these benefits below.

* Routines and repetitive tasks related to system/server changes can be extensively and thoroughly automated.
* Members of different teams are able to define and manage their own resources, allowing them to work at their own pace and speeding up the entire development process.
* Improvements are continuously integrated into the infrastructure.
* The deployment process is automated, so individual teams can fire their own deployments at any time.
* The deployment process is safer and faster.
* IAC source(s) can be stored in version control systems.
* Every change in the infrastructure state can be validated.
* The infrastructure can be divided into separate, reusable modules for testing and/or modification purposes.

Once I was done creating the ELK Stack on Microsoft Azure. I configured Filebeat to monitor the log files and collect log events then I tested it by doing a SSH barrage to generate a high amount of failed SSH login attempts and then checked if Kibana is picking up these attempts. I then configured Metricbeat and tested it by doing a stress test to generate a high amount of CPU usage and a wget-DoS attack to generate a high amount of web requests to make sure that Kibana was picking them up.

Despite its advantages, Infrastructure as Code has potential disadvantages. It requires additional tools, such as a configuration management system. Errors can spread quickly across servers. Therefore, it is important to monitor version control and perform comprehensive pre-testing.