For every issue, there is a solution.

Infrastructure is no different—due to its intricacies, complexity and the purpose it serves, troubles are guaranteed. Especially when it comes to Infrastructure as Code, as even though this concept was created to fix many extremely difficult problems it did create a few new ones.

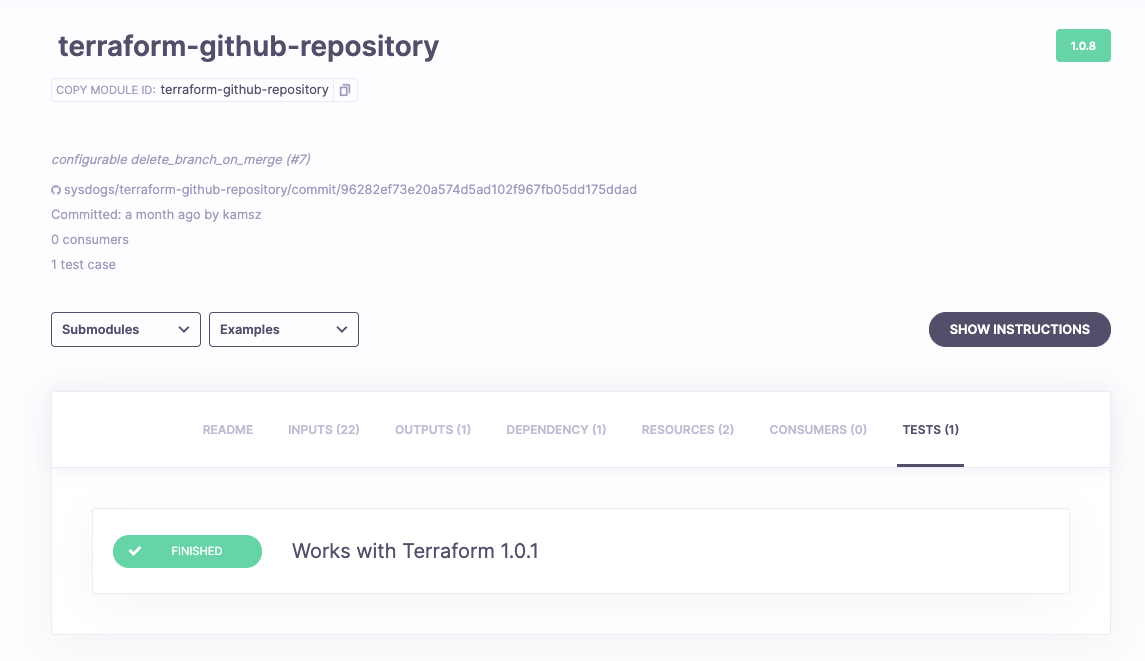
For those,**Spacelift comes to the rescue**. Let’s talk about five infrastructure problems that Spacelift solves for you.

**1. Broken Modules Tearing Down Your Configurations**

Although the modular approach is great, it presents a little quirk that you might find out about only when it’s too late. A painful one, at that. Modules are clean, efficient, granular… but not immortal. Things in the infrastructure field tend to change and some of those changes might end up breaking your modules—one unfortunate version bump, and things collapse.

Spacelift provides **automated testing for the configuration modules** it manages and I cannot emphasize enough how significant this is. So significant, in fact, that we would want to dedicate an entire article just to this outstanding functionality (stay tuned for an in-depth explanation of the registry testing).

**To test is to know what to expect**. Knowing what to expect gives you peace of mind and confidence that things will work out correctly. Even better—you don’t have to lift a finger to run comprehensive tests for your modules and infrastructure.



**2. Drifting Away From What You Had Defined**

Inanimate objects and things have an irritating tendency to not stay as you left them. If you have ever lost your keys, you will surely relate.

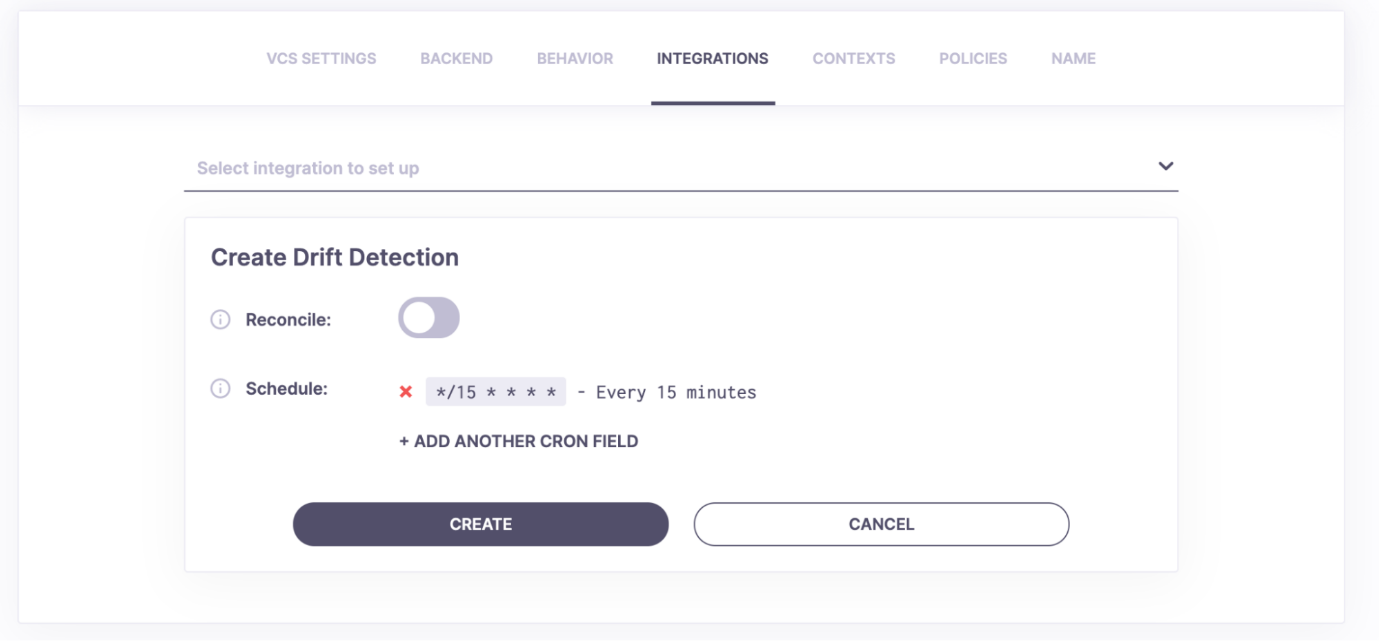
Infrastructure is no different in this matter. Unexpected situations, manual tampering or the workings of *force majeure* could lead to differences between the configuration you have applied and the current state of the system.

Sometimes the reasons are understandable. For example, when in an emergency your engineer manually introduces crucial infrastructure changes to avoid downtime or other damages; sometimes it’s not just a result of a script or person doing what they shouldn’t be.

Spacelift introduces [**automatic drift detection runs**](https://docs.spacelift.io/concepts/stack/drift-detection.html). That’s right—periodically (how often? You decide or trust the defaults), Spacelift can check your infrastructure, detect drifts, provide you information about the resources they impact:

* What was impacted?
* When did this happen?

Then, you can properly introduce them into your state in a process called *reconciliation* if you decide that they are beneficial or required.



**3. Lack of Security & Compliance**

Security is an important factor of every infrastructure, Big or small—if insecure, it won’t stand long. Unfortunately, **maintaining security and keeping compliance is cumbersome**, to say the least. There are many things to take into consideration and even more that you might forget about… until they come back to haunt you.

With some smart tricks, Spacelift makes security much easier—for example, it [**provides support for policies**](https://docs.spacelift.io/concepts/policy/). And what are those? Internally, Spacelift leverages the well-known [*Open Policy Agent*](https://spacelift.io/blog/what-is-open-policy-agent-and-how-it-works) and the Rego language. This combination allows you to define policies, affecting various aspects of daily operation such as logging in, starting Tasks and Runs, or command execution.

You can enforce compliance *by code*.

Another example is **data handling**. Data that you exchange with Spacelift is encrypted both at rest and in transit, and temporary credentials are used where needed—the secret data is protected by multiple layers of additional security.

And when it comes to shared workers and their credentials, granting static, permanent access to your AWS, Azure Cloud or GCP wouldn’t exactly be a good practice. Thanks to Spacelift’s integration with identity management systems, dynamically-generated temporary credentials can be used to provide necessary access during runs.

Whether you don’t trust shared workers or just prefer to provide and use your own with Spacelift —this is also possible.

**💡 You might also like:**

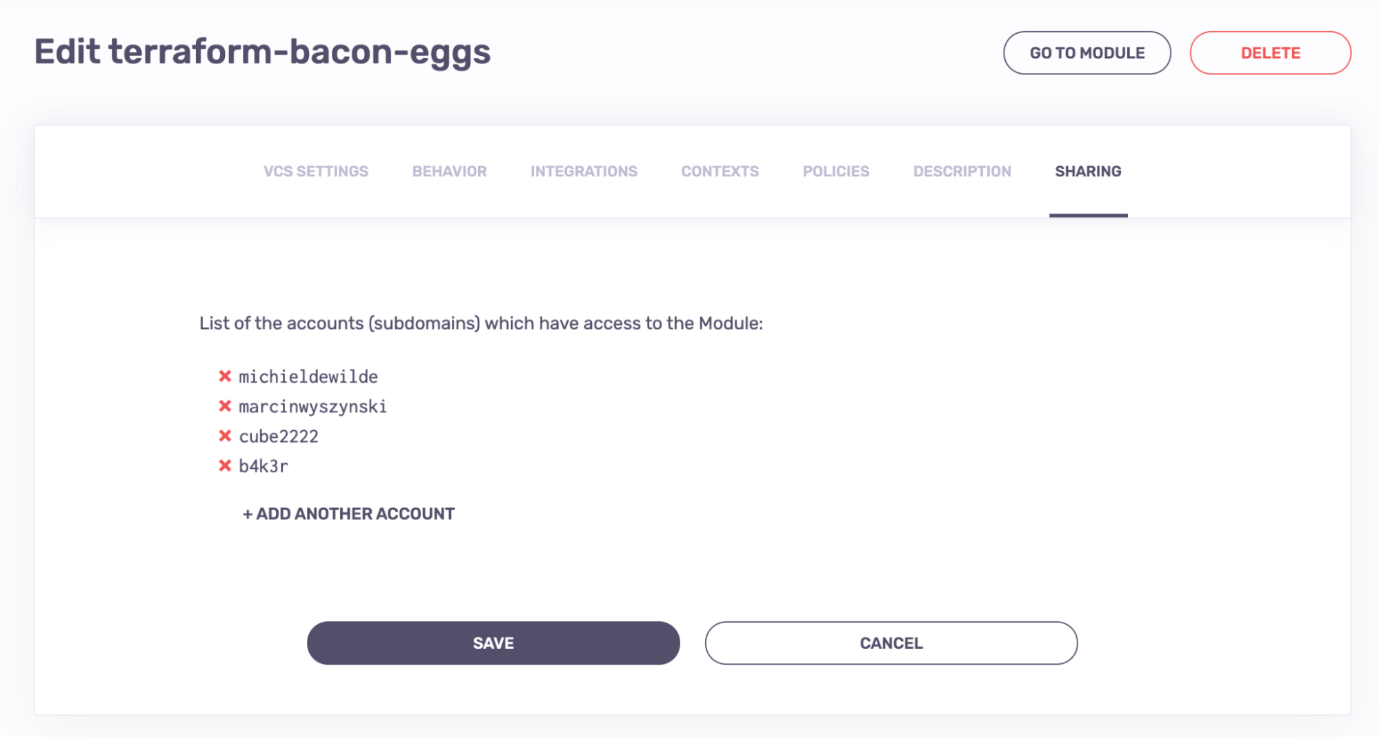
* [Why DevOps Engineers Recommend Spacelift](https://spacelift.io/blog/why-devops-engineers-recommend-spacelift)
* [How to Use Spacelift Module Registry](https://spacelift.io/blog/spacelift-module-registry)
* [Alternative to Atlantis – Spacelift](https://spacelift.io/blog/alternative-to-atlantis)

**4. Troublesome Collaboration**

[Infrastructure as Code](https://spacelift.io/blog/infrastructure-as-code) significantly improves collaboration. It is much easier to work with someone on a piece of code than on a server spinning up, but there are still a few peculiarities that you might find out about along the way.

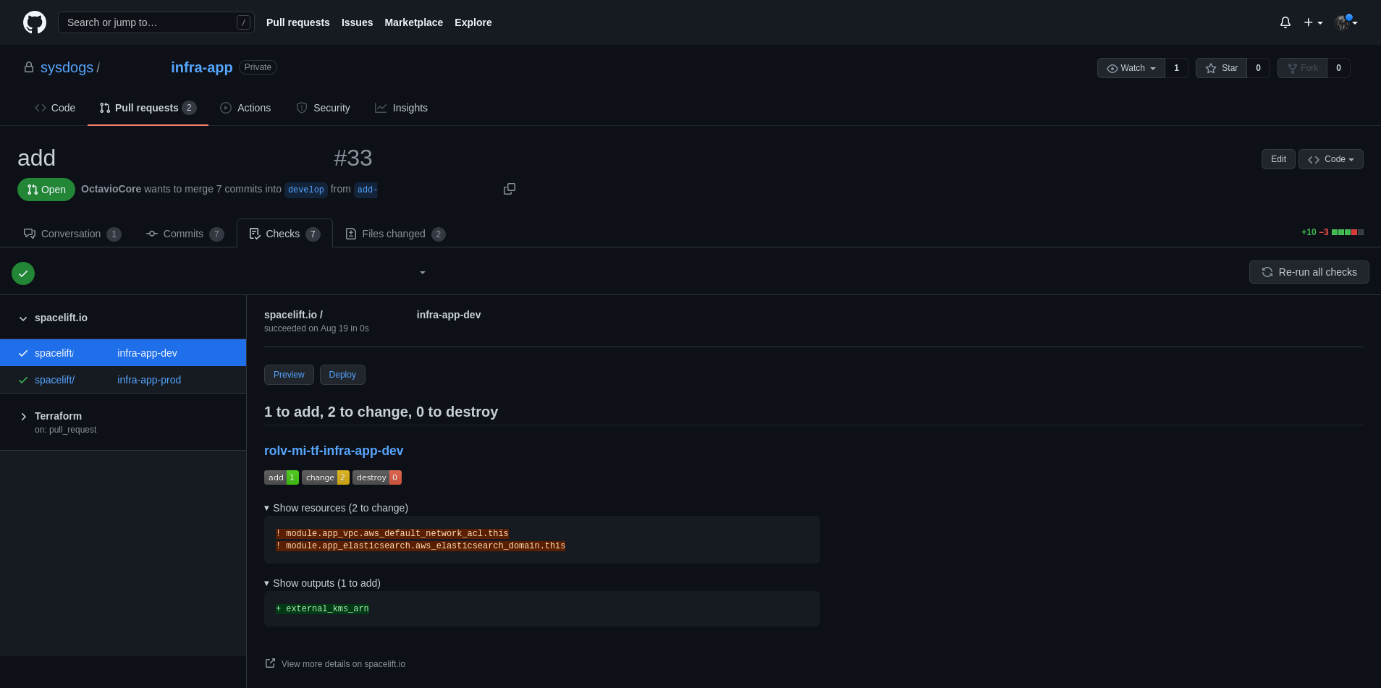
Take this example. Let’s assume you’ve got some configuration elements, like environment variables, that you wish to use as shared—between various groups of resources. Normally, you’d need to individually provide those values to each resource set that needs them by defining them multiple times, in different places. To significantly ease this process, Spacelift introduces the [**concept of contexts**](https://docs.spacelift.io/concepts/configuration/context.html)—grouped declarations of variable values and useful files that you can mount to your stacks. You declare them once and use them where you need, with no security disadvantages or time wasted on useless repetition.

Another situation that you might run into, would be when you’d want to use your [**Terraform modules**](https://spacelift.io/blog/what-are-terraform-modules-and-how-do-they-work)**in configurations outside your organization**. For example, you have five customers. Each one has a separate account, and ideally, you’d want to retain control over the configuration modules you made, to make sure no one breaks anything. But—you also want to use the configuration you already have, for repetitive tasks between the infrastructures of those customers. Spacelift makes this possible:



Just provide the names of accounts you wish to share your code with, and let those accounts build their infrastructure with your curated resources. Easy!

Still unconvinced? **There’s much more**. Seamless integration with popular VCS providers such as GitHub or BitBucket, push status notifications… and of course, preview runs, showing you what will change as a result of the pull request.



As you can see, there’s something here for everyone. **Spacelift is developed with collaboration in mind**. It blends in very well but provides features that stand out. Even if you don’t really like using external tools or technologies, you won’t be disappointed with the results.

**5. Budgets Out of Hand**

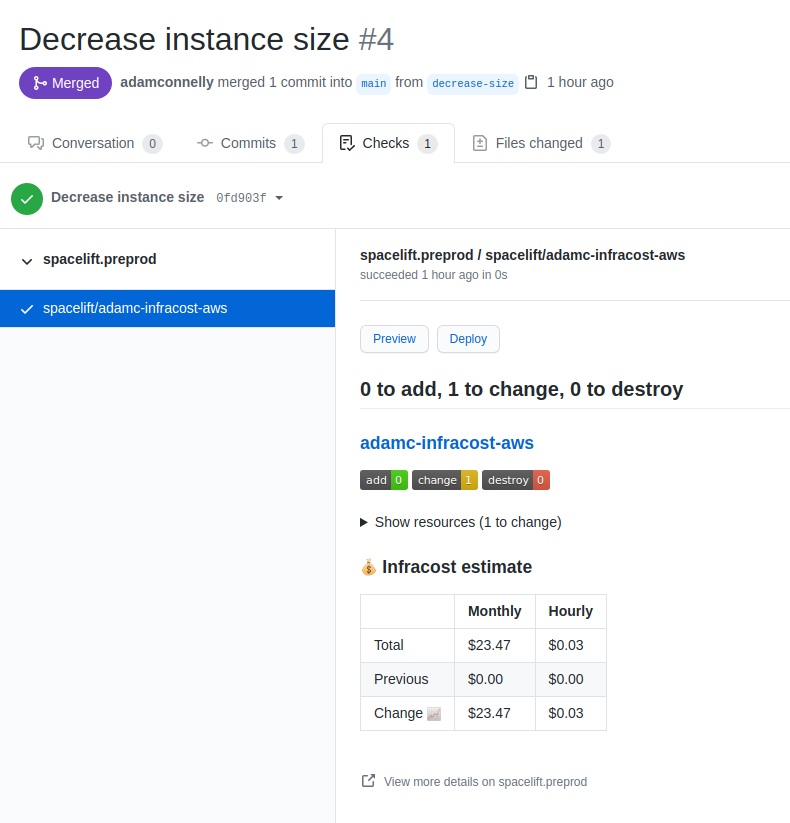
Everyone at least once in their lifetime bought something and then watched in disbelief as it diminished before their eyes. The pain of a huge Amazon Web Services bill is definitely real.

AWS bills are often mentioned in the countdowns of some of the most massive things in existence—somewhere between cargo ships, black holes, and the NodeJS *node\_modules* directories. Some engineers even say that you don’t really need coffee in your life if you’re provisioning AWS infrastructure—the bill will raise your blood pressure just the same.

Spacelift integrates with [Infracost](https://www.infracost.io/" \t "_blank), to provide you with [**cost estimates of your runs**](https://docs.spacelift.io/vendors/terraform/infracost.html). This way your pull request can tell you how much it will cost to apply the configuration you have designed. It also allows you to set up proper **cost barriers** to make sure that you won’t spend more than expected just by misconfiguration or burn the entire monthly budget in a week by one unfortunate deployment.

Read more about it in our [Terraform Cost Estimation with Infracost](https://spacelift.io/blog/terraform-cost-estimation-using-infracost) article.

The setup is fairly simple, but to make it even easier you can create a context mentioned in an earlier section and attach it to the stacks you wish to monitor in terms of budget.



**Key Points**

As you’ve seen, while the Infrastructure as Code approach definitely solves many issues, there’s still a huge lot to go around. Every bit of assistance or automation is precious and can provide tremendous benefits.

If you wish to make the management of your infrastructure easier, less time-consuming, the infrastructure itself more robust and dependable… and also keep everything as secure and as under control as possible—**[Spacelift](https://spacelift.io/) is the way to go**. I’m sure you will love it.