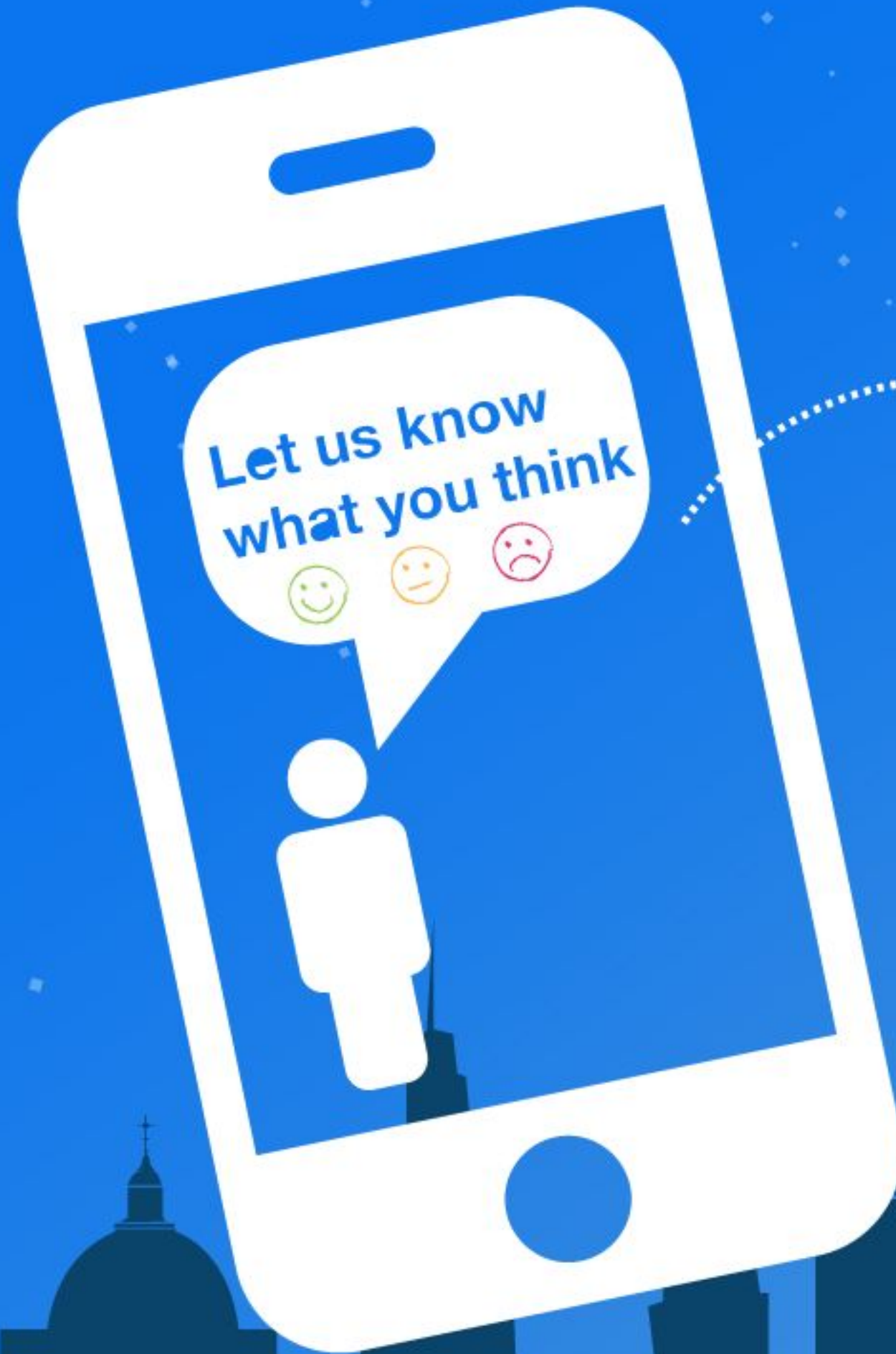


Terraform - Past, Present (0.12) and Future

Radek Simko





**Click 'Rate Session'
to rate session
and ask questions.**





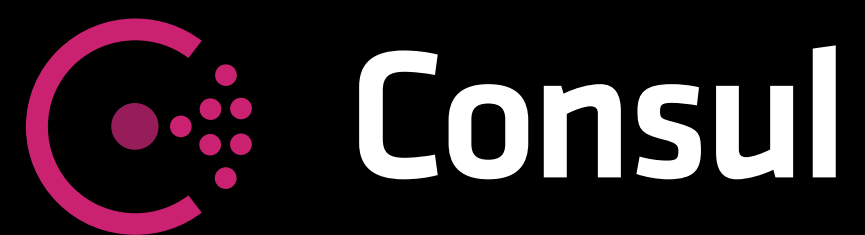
Radek Simko

Software Engineer at HashiCorp
(since Jan 2017)

*Terraform Contributor
(since Feb 2015)*

@radeksimko







What motivated 0.12?

Main Themes

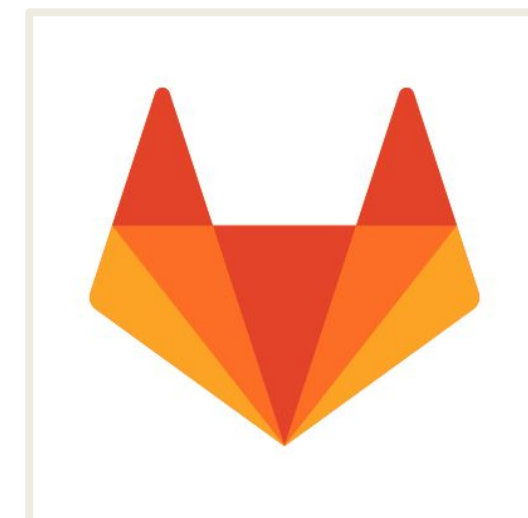
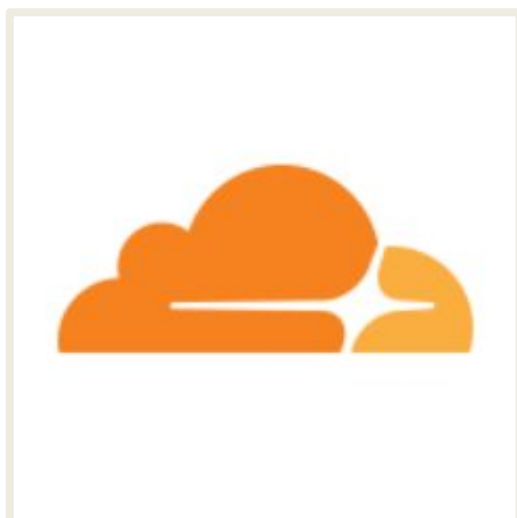
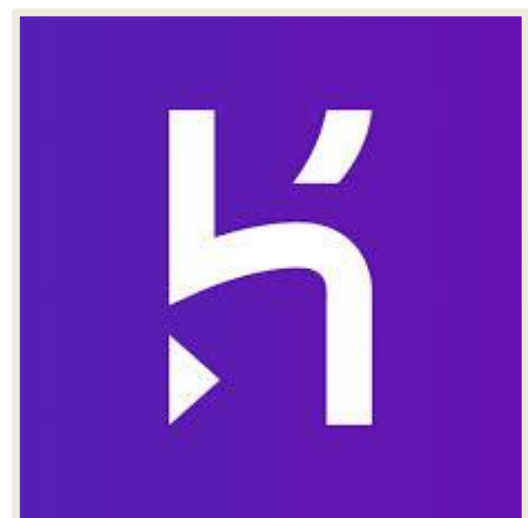
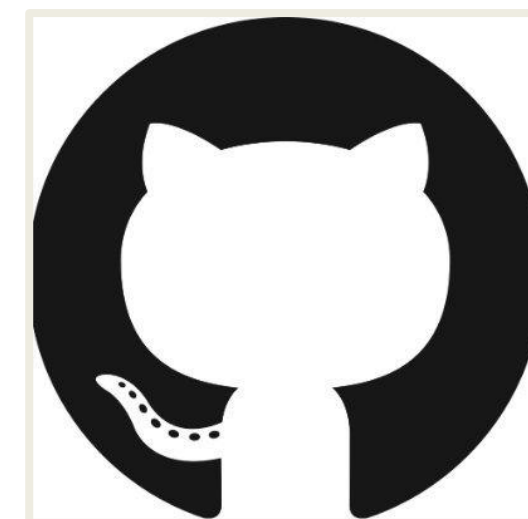
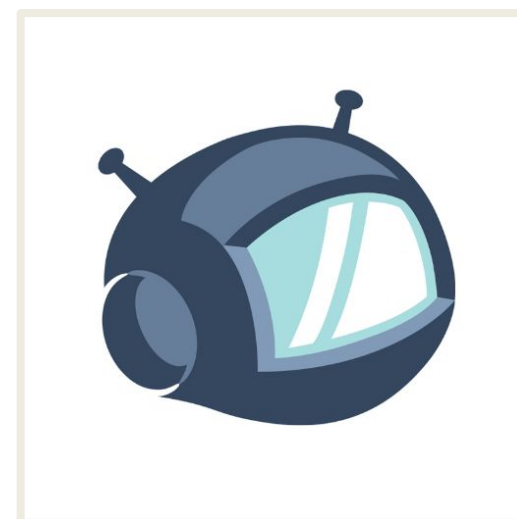
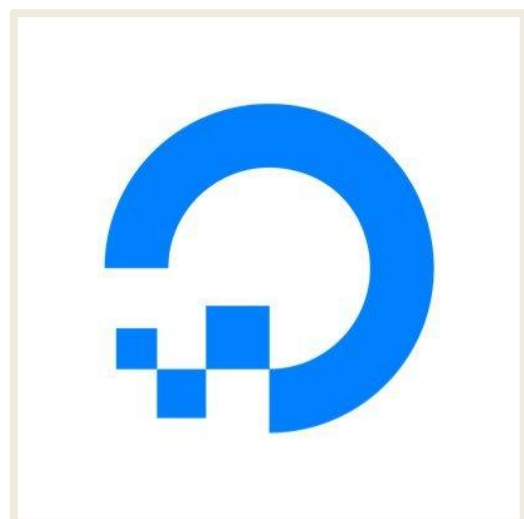
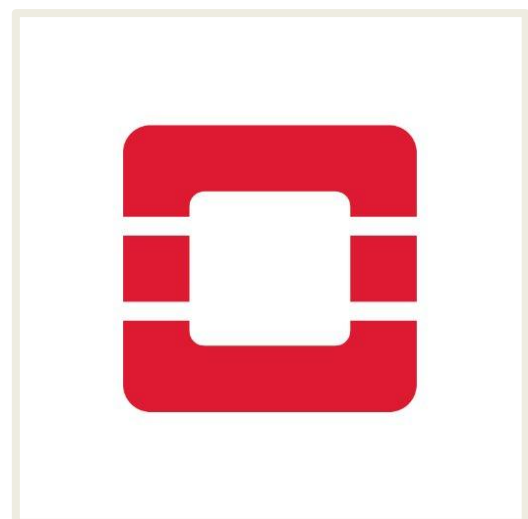
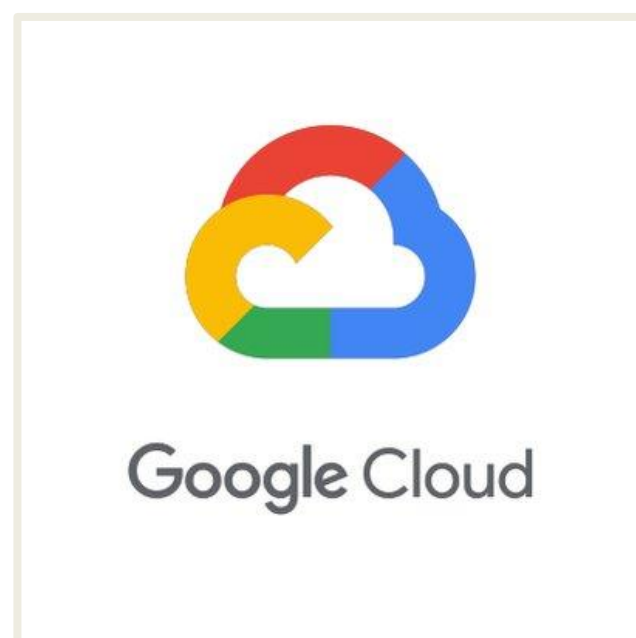
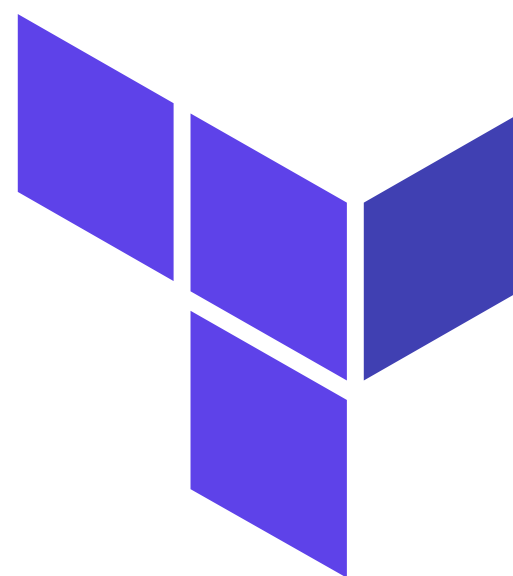


1. Provider discovery & distribution
2. Configuration Language
3. Machine-readable output
4. Plugin SDK



1 / 4

Provider Discovery & Distribution





```
provider "aws" {
  version = ">=2.0.0"
  region  = "us-west-2"
}

data "aws_ami" "ubuntu" {
  filter {
    name      = "name"
    values    = ["ubuntu/.../ubuntu-trusty-14.04-amd64-server-*"]
  }
  # ...
  owners = ["099720109477"] # Canonical
}

resource "aws_instance" "example" {
  ami           = "${data.aws_ami.ubuntu.id}"
  instance_type = "t2.micro"
}
```




TERMINAL

```
$ terraform init
```


=< 0.6.0



June 2015

- **1 binary** per provider for each OS/arch
- **1 binary** for core
- **10-30 providers**
- **monorepo** w/ providers in `/builtin/...`

=< 0.6.0



June 2015

- ~**40** binaries distributed as ZIP for each OS/arch
 - ◆ core **21MB** + providers **12-36MB** each (~**600MB**)
- **monorepo** w/ providers in `/builtin/...`

0.7.0

August 2016



- “E_TOO_MANY” binaries!
- Big binaries
- Sub-optimal installation experience

0.7.0

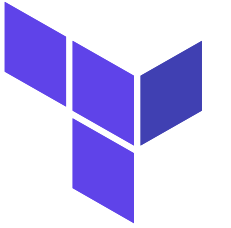
August 2016



- **one** binary for each OS/arch
- still monorepo

0.10.0

August 2017



→ **“E_TOO_MANY_PROVIDERS”**

→ **“E_TOO_MANY_CHANGES”**

0.10.0

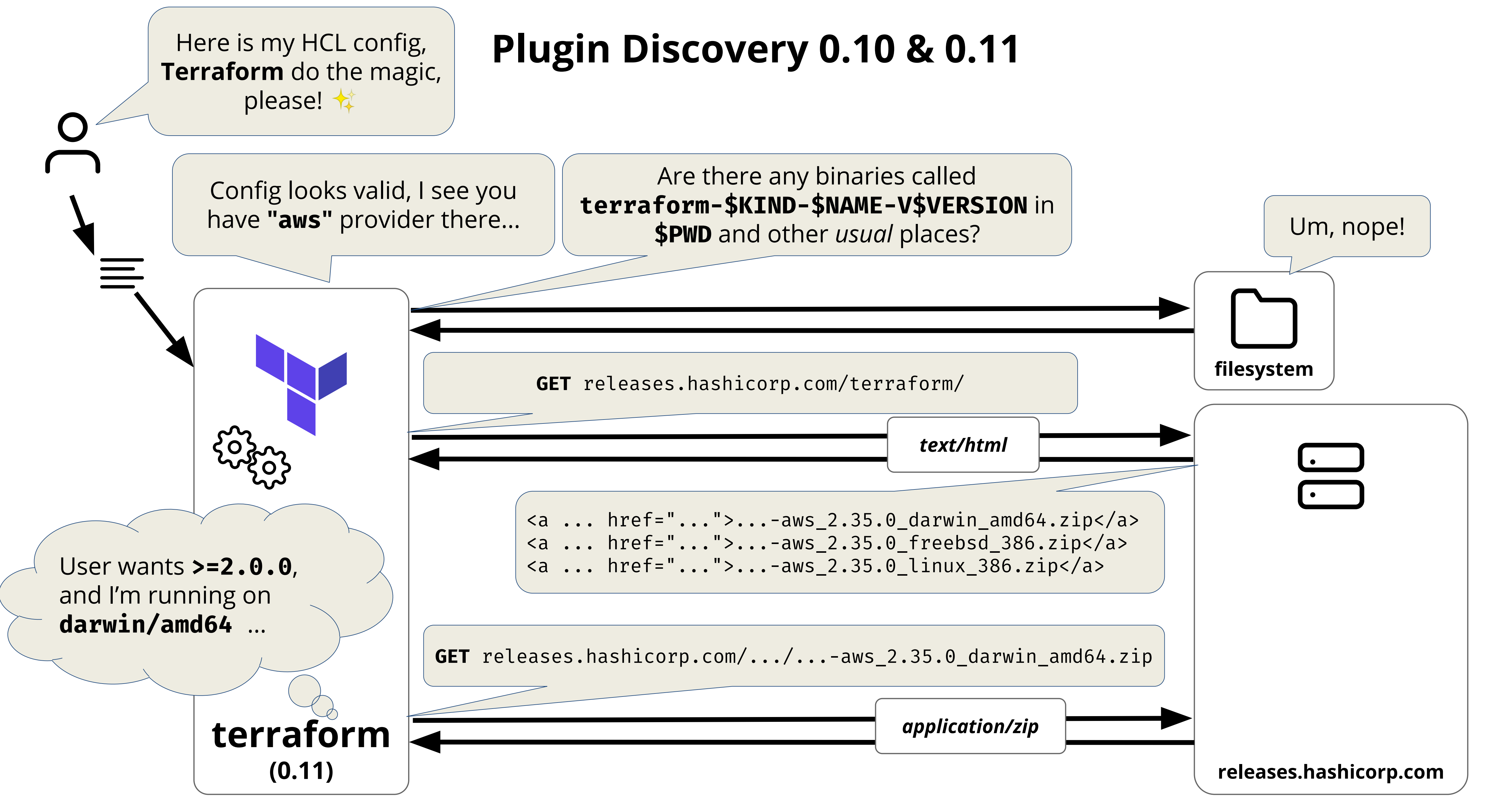
August 2017



→ **one** binary with just core

- ◆ **providers** in separate binaries **downloaded on-demand**
- ◆ **providers** versioned separately (**multi-repo**)
- ◆ provider **constraints**
- ◆ `terraform init` practically mandatory

Plugin Discovery 0.10 & 0.11



terraform init (0.10 & 0.11)



```

$ terraform init

...

- Downloading plugin for provider "aws" (2.37.0)...
```


terraform init (0.12)



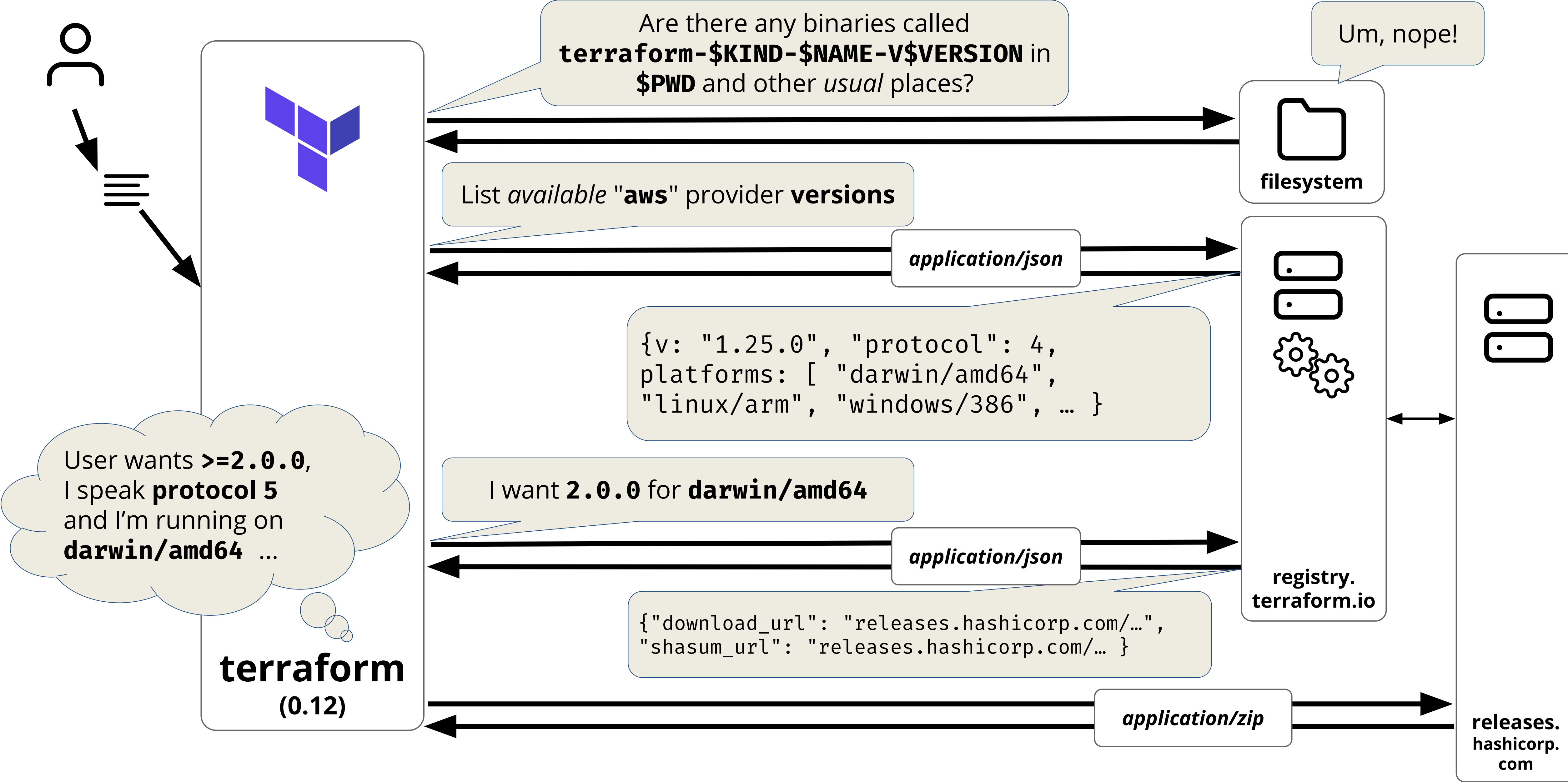
```

$ terraform init

...

- Downloading plugin for provider "aws" (hashicorp/aws)
2.37.0...
```

Plugin Discovery in 0.12.0



3rd Party Providers Today



→ HashiCorp effectively gatekeeper

- ◆ Consistency

- ◆ Healthiness

- ◆ E2E testing

→ “E_TOO_MANY_PROVIDERS”

- ◆ Not very scalable

- ◆ Providers want independence

Future: Terraform Registry



- Better story for 3rd party providers
 - ◆ No need to be hosted on `releases.hashicorp.com`
 - ◆ Discoverable via `terraform init`
 - ◆ Hosted documentation



HashiCorp

Terraform

Registry

Providers

/ hashicorp

/ aws

/ Version 2.35.0



Latest Version

aws



aws



Official

by:



HashiCorp

VERSION

2.35.0



PUBLISHED

5 days ago



INSTALLS

20819624



SOURCE

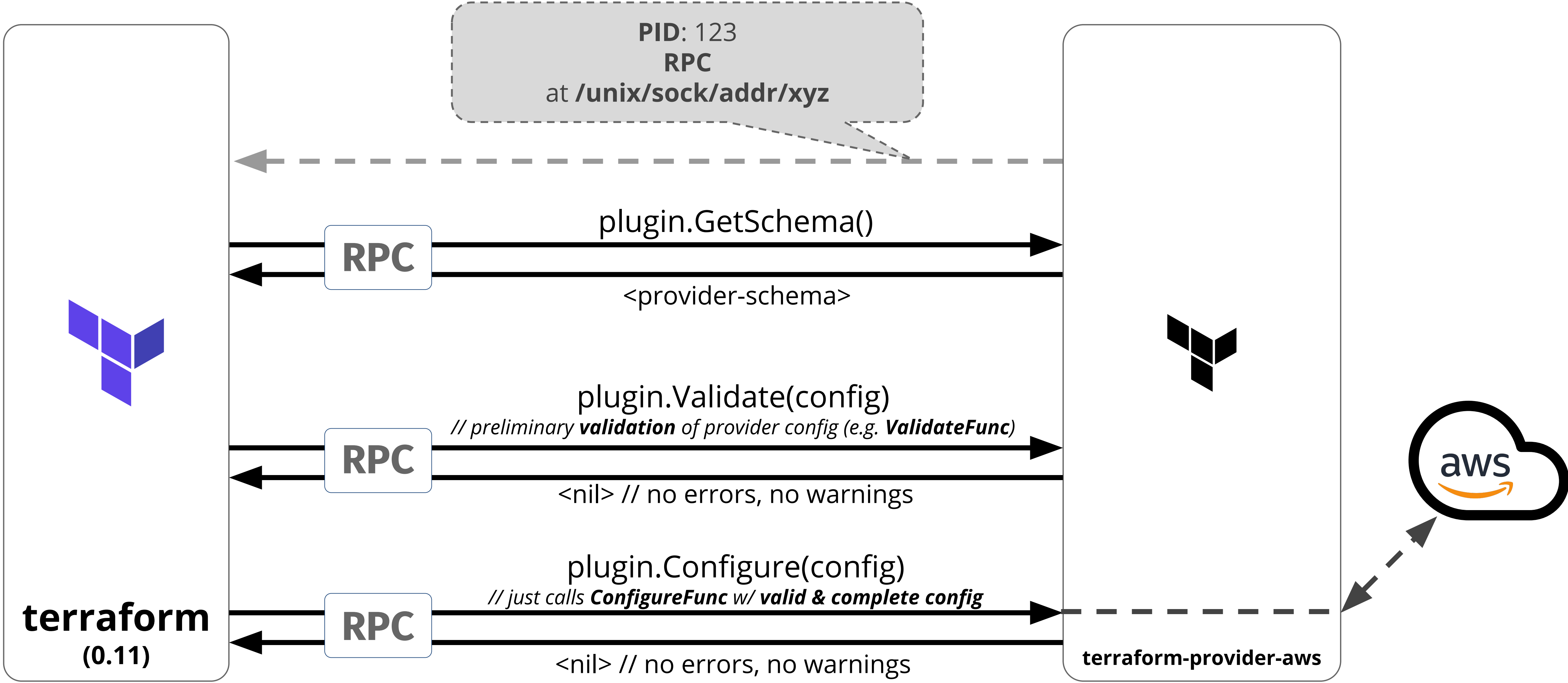
[terraform-providers/terraform-provider-aws](https://github.com/hashicorp/terraform-provider-aws)



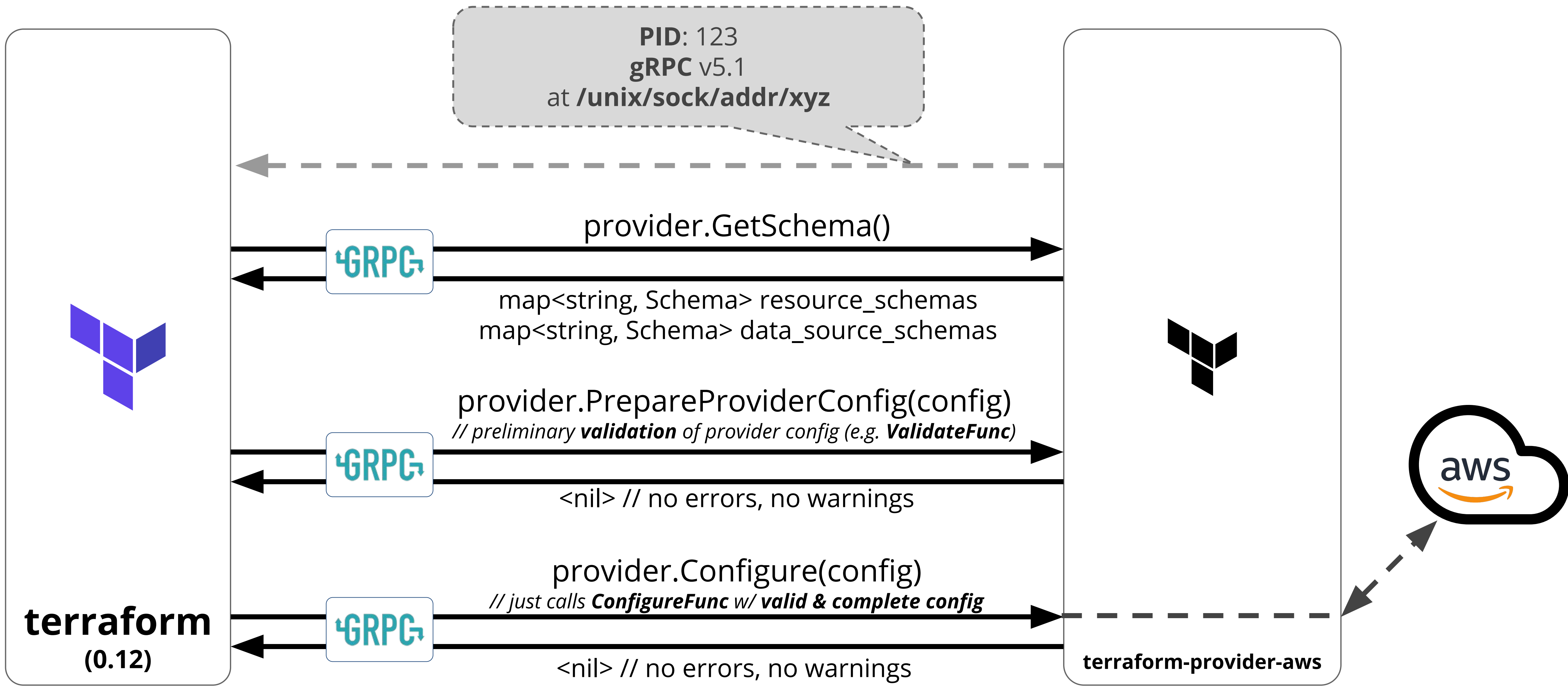
2 / 5

Plugin Protocol

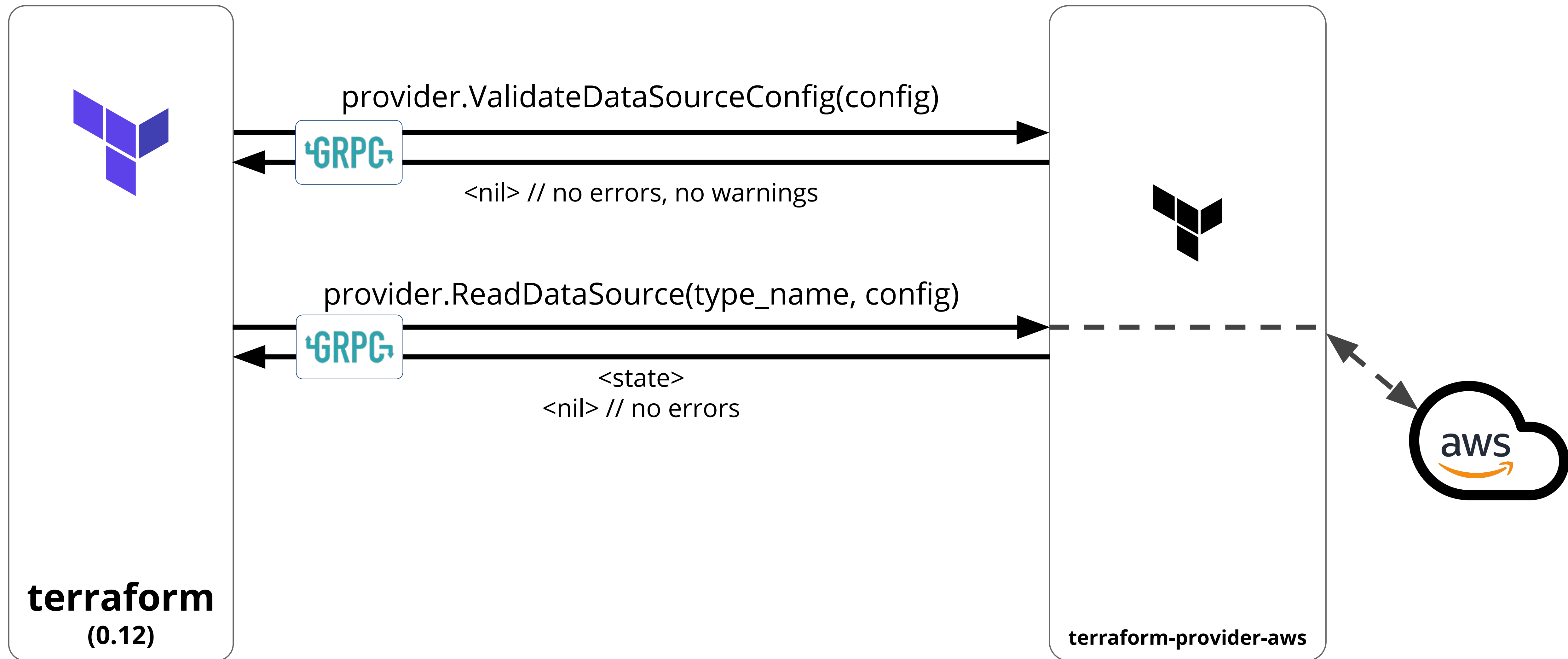
Plugin Communication (0.11) - Provider



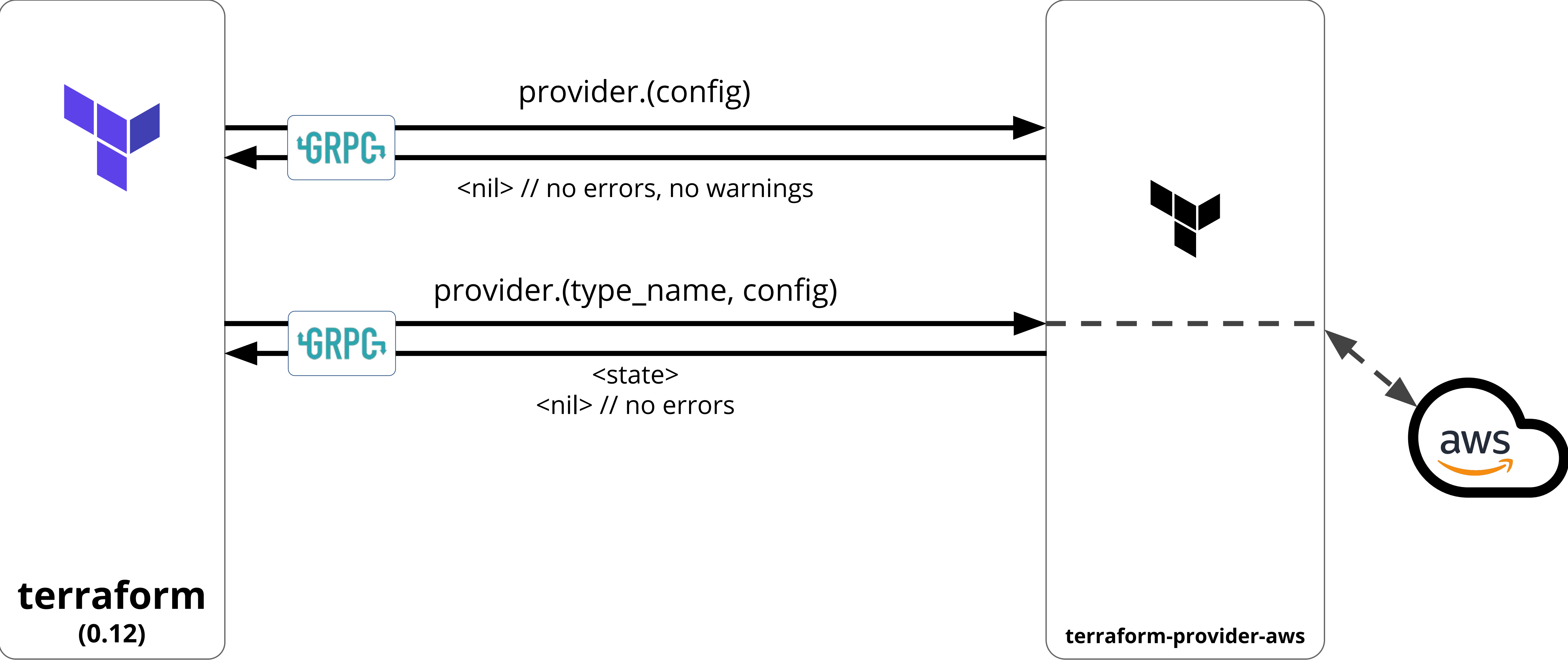
Plugin Communication 0.12 (provider)



Plugin Communication 0.12 (data source)



Plugin Communication 0.12 (resource)





2 / 4

Configuration Language

core: Change string list representation so we can distinguish empty, single element lists #2504

 **Merged** phinze merged 4 commits into `master` from `f-string-list`  on 26 Jun 2015

 Conversation **14**

 Commits **4**

 Checks **0**

 Files changed **9**



phinze commented on 26 Jun 2015

Member



This encapsulates and enhances the internal representation of lists within terraform's core.

To work around limitations of the current architecture, lists need to be "flattened" into a string as they're passed around the config interpolation.

Given a list `["foo", "bar", "baz"]`, the previous format for these strings was: `foo(special delimiter)bar(special delimiter)baz`.

This format made it impossible to distinguish between:

- `["foo"]` and `"foo"`
- `[]` and `""`
- `[""]` and `""`

Reviewers

No reviews

Assignees

No one—ass

Labels

None yet

Projects

None yet

Milestone

config: Add map() interpolation function #7832



Merged

phinze merged 1 commit into `master` from `f-map-func` on 27 Jul 2016



Conversation 4



Commits 1



Checks 0



Files changed 3



phinze commented on 27 Jul 2016

Member



- `map(key, value, ...)` - Returns a map consisting of the key/value pairs specified as arguments. Every odd argument must be a string key, and every even argument must have the same type as the other values specified. Examples:
 - `map("hello", "world")`
 - `map("us-east", list("a", "b", "c"), "us-west", list("b", "c", "d"))`



1

core: allow non-computed data source values in "count" #11482

Merged mitchellh merged 5 commits into `master` from `f-computed-count` on 30 Jan 2017

Conversation 12 Commits 5 Checks 0 Files changed 16



mitchellh commented on 28 Jan 2017 • edited ▾

Member



Based on [#10418](#), but simpler due to new graphs (more LOC change due to tests).

This disables the computed value check for `count` during the validation pass. This enables partial support for [#3888](#) or [#1497](#): as long as the value is non-computed during the plan, complex values will work in counts.

Notably, this allows data source values to be present in counts!

The "count" value can be disabled during validation safely because we can treat it as if any field that uses `count.index` is computed for validation. We then validate a single instance (as if `count = 1`) just to make sure all required fields are set.

Review



Assign

No on

Label

core

enha

Project

None



2 / 4 - Configuration Language

Behind the Scenes

Relevant Projects



- HCL1 & HIL (~ before 0.12)
- HCL2 & cty (~ 0.12+)

HCL1 & HIL



HashiCorp {Config, Interpolation} Language

- HCL for configuration, not interpolation
- Complex types (maps, lists, sets) stored as strings
- HIL as separate language for string interpolation



```
provider "aws" {  
  version = ">=2.0.0"  
  region  = "us-west-2"  
}  
  
data "aws_ami" "ubuntu" {  
  filter {  
    name      = "name"  
    values    = ["ubuntu/.../ubuntu-trusty-14.04-amd64-server-*"]  
  }  
  # ...  
  owners = ["099720109477"] # Canonical  
}  
  
resource "aws_instance" "example" {  
  ami           = "${data.aws_ami.ubuntu.id}"  
  instance_type = "t2.micro"  
}
```





```
provider "aws" {
  version = ">=2.0.0"
  region  = "us-west-2"
}

data "aws_ami" "ubuntu" {
  filter {
    name      = "name"
    values    = ["ubuntu/.../ubuntu-trusty-14.04-amd64-server-*"]
  }
  # ...
  owners = ["099720109477"] # Canonical
}

resource "aws_instance" "example" {
  ami           = "${data.aws_ami.ubuntu.id}"
  instance_type = "t2.micro"
}
```


HCL2 & cty



→ HCL2

- ◆ Interpolation as part of the language

→ **cty** to deal with types and operations on types

- ◆ no more string weirdness



2 / 4 - Configuration Language

0.12

First-Class Expressions

0.12: First-Class Expressions



- Expressions are native to HCL2
 - ◆ Functions and variables can be used outside of string interpolation (`"${}"`)
- Lists and maps can be used directly
 - ◆ Less `"${list("")}"`, more `[]`
 - ◆ Less `"${map("key", "val")}"`

Terraform 0.11



```
data "docker_registry_image" "consul" {  
  name = "consul"  
}
```

```
resource "docker_image" "consul" {  
  name          = "${data.docker_registry_image.consul.name}"  
  pull_triggers = ["${data.docker_registry_image.consul.sha256_digest}"]  
}
```

```
resource "docker_container" "consul" {  
  name = "consul"  
  image = "${docker_image.consul.latest}"  
}
```


Terraform 0.12



```
data "docker_registry_image" "consul" {  
  name = "consul"  
}
```

```
resource "docker_image" "consul" {  
  name          = data.docker_registry_image.consul.name  
  pull_triggers = [data.docker_registry_image.consul.sha256_digest]  
}
```

```
resource "docker_container" "consul" {  
  name = "consul"  
  image = docker_image.consul.latest  
}
```

Terraform 0.11



```
variable "instance_type" {}  
variable "vpc_security_group_id" {  
    type      = "string"  
    default = ""  
}
```

```
resource "aws_instance" "example" {  
    ami      = "ami-123456789"
```



```
    instance_type = "${var.instance_type}"
```



```
    vpc_security_group_ids =  
        "${var.vpc_security_group_id != "" ?  
         [var.vpc_security_group_id] : list("")}"
```

```
}
```


Terraform 0.12



```
variable "instance_type" {}  
variable "vpc_security_group_id" {  
    type      = string  
    default = ""  
}
```

```
resource "aws_instance" "example" {  
    ami      = "ami-123456789"
```

👁 ☐ instance_type = var.instance_type

👁 ☐ vpc_security_group_ids =
 var.vpc_security_group_id != "" ?
 [var.vpc_security_group_id] : []
}



2 / 4 - Configuration Language

0.12

Rich Value Types

0.12: Rich Value Types



→ Complex Values

- ◆ **Complex objects** can be passed to child **modules** as **inputs**, and returned to parent **modules** as **outputs**.

→ **Resources** and **Modules** can be used as **Values**

- ◆ Entire resources and modules can be used as object values within configuration, including returning them as outputs and passing them as input variables.

Terraform 0.12



```
variable "consul_ports" {  
  type = list(object({  
    internal = number,  
    external = number,  
    protocol = string  
  }))  
  default = [  
    {  
      internal = 8300  
      external = 8300  
      protocol = "tcp"  
    } ...  
  ]  
}
```

CODE EDITOR

Terraform 0.12



```
CODE EDITOR

// Inside the "consul" module
resource "docker_container" "consul" {
    ...
}
output "consul_container_name" {
    value = docker_container.consul.name
}
output "consul_container_ips" {
    value = docker_container.consul.network_data.*.ip_address
}
output "consul_container_hostname" {
    value = docker_container.consul.hostname
}
```

Terraform 0.12



```
CODE EDITOR

// Inside the "consul" module
resource "docker_container" "consul" {
    ...
}

// Fun fact: the resource docker_container
// exposes 53 attributes
output "consul_container" {
    value = docker_container.consul
}
```




2 / 4 - Configuration Language

0.12

Improved Error Messages

Terraform 0.11



```
$ terraform plan
```

```
Error: Error parsing main.tf: object expected closing  
RBRACE got: EOF
```


Terraform 0.12



```
$ terraform plan
```

```
Error: Argument or block definition required
```

```
on main.tf line 24:  
24: :q!
```

```
An argument or block definition is required here.
```

Terraform 0.12



```
$ terraform plan
```

```
Error: Invalid operand
```

```
on outputs.tf line 16, in output "example":
```

```
16:   value = 1 + var.foo
```

```
Unsuitable value for right operand: a number is  
required.
```


Terraform 0.12



```
$ terraform plan
```

```
Error: Unsupported block type
```

```
on main.tf line 36, in output "item":
```

```
1: output "item" {
```

```
Blocks of type "output" are not expected here. Did you  
mean "output"?
```




2 / 4 - Configuration Language

0.12

For Expressions

0.12: For Expressions



→ `for`

- ◆ Used for **list** and **map** transformation

→ `for_each`

- ◆ Iterate over **sets** or **maps**

→ `dynamic`

- ◆ Used with "**for_each**" to dynamically construct a collection of nested blocks

0.12: for expression



```
$ terraform console
> {for s in var.list : s => upper(s)}
{
  "a" = "A"
  "b" = "B"
  "c" = "C"
}
> [for s in var.list : upper(s) if s != "b"]
[
  "A",
  "C",
]
```


0.12: Nested Block



```
resource "docker_container" "consul" {  
    # ...  
    ports {  
        internal = 8600  
        external = 8600  
        protocol = "tcp"  
    }  
    ports {  
        internal = 8600  
        external = 53  
        protocol = "udp"  
    }  
    ports { }  
    ports { }
```

CODE EDITOR

0.12: Dynamic Nested Block



```
resource "docker_container" "consul" {  
    # ...  
    dynamic "ports" {  
        for_each = var.consul_ports  
        content = {  
            internal = ports.value.internal  
            external = ports.value.external  
            protocol = ports.value.protocol  
        }  
    }  
}
```




3 / 4

Machine-Readable Output

Machine-readable output



1. `terraform show -json`
2. `terraform-config-inspect`
3. `terraform providers schema -json`



3 / 4 - Machine Readable Output

terraform show


```
$ terraform show
```

```
# module.consul_container.data.docker_registry_image.consul:
data "docker_registry_image" "consul" {
  id          = "sha256:e70c788854dd6e8ee0aa3d4c3a6a8c717ce74ed3c18b1665d983bf7e5fff5486"
  name        = "consul"
  sha256_digest = "sha256:e70c788854dd6e8ee0aa3d4c3a6a8c717ce74ed3c18b1665d983bf7e5fff5486"
}

# module.consul_container.docker_container.consul:
resource "docker_container" "consul" {
  attach      = false
  env         = [
    "CONSUL_HTTP_TOKEN=supersecure",
  ]
  gateway     = "172.23.0.1"
  hostname    = "consul"
  id          = "ce9a2197e65de5cf9174389f4209ac5e331f482e708a7edf71474aef747bf922"
  image       = "sha256:f136343b75e00664a7aa4c477a6d1e7b40a9708610fb6e483e884e0cce691599"
  ip_address  = "172.23.0.2"
  ip_prefix_length = 16
  log_driver  = "json-file"
  logs        = false
  must_run    = true
  name        = "consul-noted-pig"
  network_data = [
    {
      gateway          = "172.23.0.1"
      ip_address       = "172.23.0.2"
      ip_prefix_length = 16
      network_name     = "ministack"
    }
  ]
}
```



```
$ terraform show -json | jq
{
  "format_version": "0.1",
  "terraform_version": "0.12.1",
  "values": {
    "root_module": {
      "child_modules": [
        {
          "resources": [
            {
              "address": "data.docker_registry_image.consul",
              "mode": "data",
              "type": "docker_registry_image",
              "name": "consul",
              "provider_name": "docker",
              "schema_version": 0,
              "values": {
                "id": "sha256:e70c788854dd6e8ee0aa3d4c3a6a8c717ce74ed3c18b1665d983bf7e5fff5486",
                "name": "consul",
                "sha256_digest":
"sha256:e70c788854dd6e8ee0aa3d4c3a6a8c717ce74ed3c18b1665d983bf7e5fff5486"
              }
            },
            {
              "address": "docker_container.consul",
              "mode": "managed",
              "type": "docker_container",
              "name": "consul",
              "provider_name": "docker",
              "schema_version": 1,
              "values": {
```

Machine-Readable Output: state



```

$ terraform show -json | \
> jq '.values.root_module.child_modules[].resources[].address'

"data.docker_registry_image.consul"
"docker_container.consul"
"docker_image.consul"
"null_resource.consul_directory"
"random_pet.pet_name"
"data.docker_registry_image.vault"
"docker_container.vault"
"docker_image.vault"
"random_pet.pet_name"
"docker_network.private_network"

```


Machine-Readable Output: plan



```

$ terraform plan -out tf.plan
$ terraform show -json tf.plan | \
> jq "[.resource_changes[] | { resource: .address, changes:
.change.actions }]"
[
  {
    "resource": "module.consul_container.docker_container.consul",
    "changes": ["create"]
  },
  {
    "resource": "module.consul_container.docker_image.consul",
    "changes": ["no-op"]
  },
  {
    "resource": "module.consul_container.null_resource.consul_directory",
    "changes": ["no-op"]
  },
  ...

```

Machine-Readable Output: plan (config)



```

$ terraform show -json tf.plan | \
> jq ' .configuration.root_module.module_calls[] | {module:
.source, variables: .module.variables}'

{
  "module": "./modules/vault",
  "variables": {
    "network": {
      "description": "An entire `docker_network` resource"
    }
  }
}
```




3 / 4 - Machine Readable Output

terraform-config-inspect

terraform-config-inspect



```
$ terraform-config-inspect --json
{
  "path": ".",
  "variables": {},
  "outputs": {
    "consul_container": {
      "name": "consul_container",
      "pos": {
        "filename": "main.tf",
        "line": 16
      }
    }
  },
  ...
}
```




3 / 4 - Machine Readable Output

terraform providers schema

Machine-Readable Output: schemas



```
$ terraform providers schema -json | jq '.provider_schemas | keys'
```

```
[  
  "docker",  
  "null",  
  "random"  
]
```



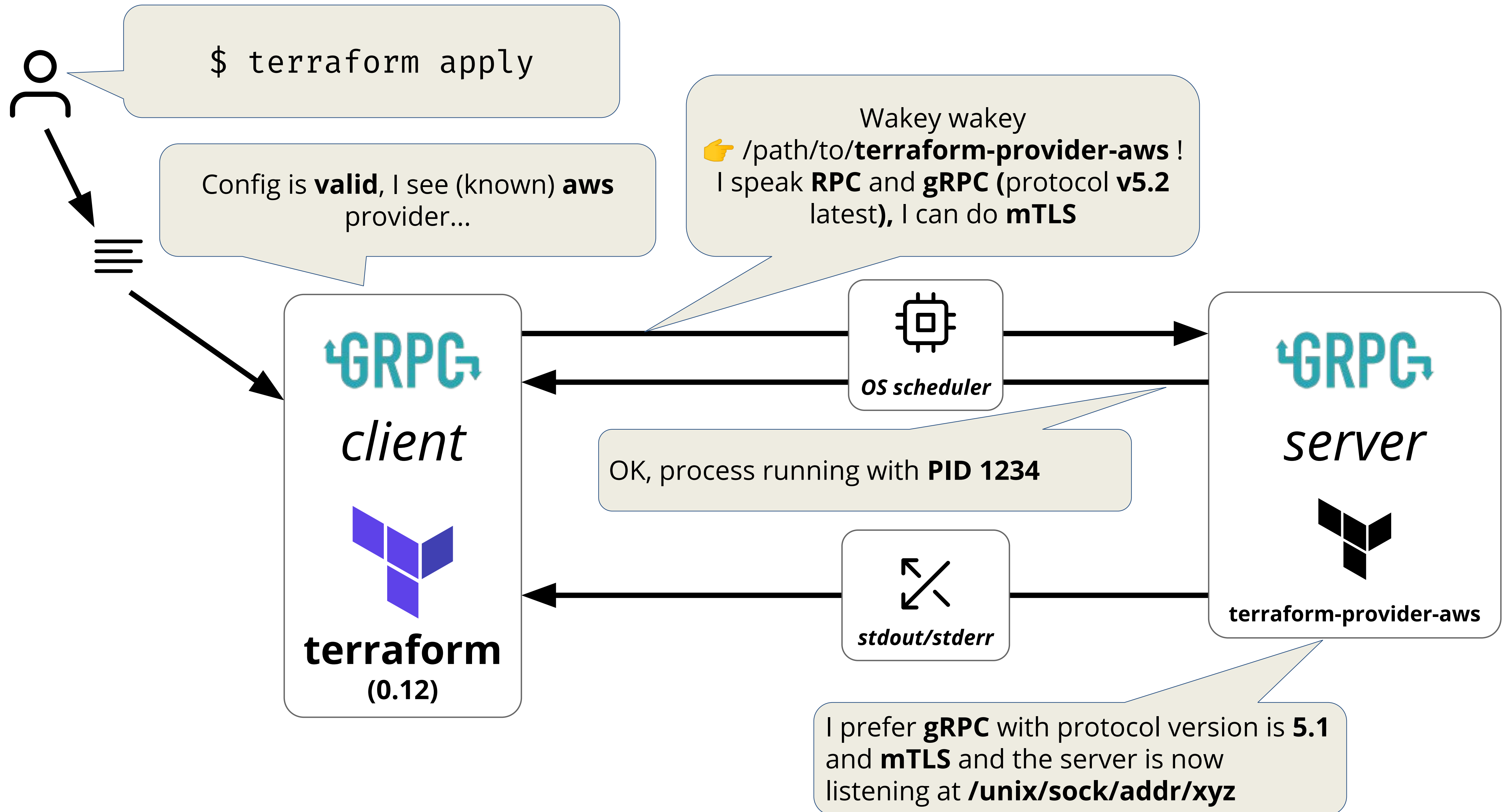
```
terraform providers schema -json | jq \
> '.provider_schemas["docker"].resource_schemas["docker_container"].block | \
> .attributes, .block_types.attributes'
...
  "network_alias": {
    "type": [
      "set",
      "string"
    ],
    "description": "Set an alias for the container in all specified networks",
    "optional": true
  },
  "network_data": {
    "type": [
      "list",
      [
        "object",
        {
          "gateway": "string",
          "ip_address": "string",
          "ip_prefix_length": "number",
          "network_name": "string"
        }
      ]
    ],
  },
]
```




4 / 4

Plugin SDK

Launching plugin (after discovery)



What is Plugin SDK?



- Hides implementation details, such as
 - ◆ gRPC itself
 - ◆ Protocol built on gRPC
 - ◆ Low-level operations
- Allows providers to talk in high-level primitives
 - ◆ Resource's and data source's schemas
 - ◆ CRUD

SDK "<1.0.0"



Before it was called SDK

- Loosely coupled packages inside Terraform's `/helper/...` namespace
- Versioned alongside Terraform core
- Made providers depend on Terraform core
- Allowed providers to be *"too creative"*

SDK 1.0.0



Released in September 2019

- More strictly separated namespace
 - ◆ Prevent import of non-SDK packages
- As compatible w/ `<1.0.0` as possible
- Migration tool available for convenience
- Versioned separately from core

SDK 1.0.0



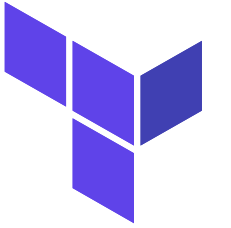
→ Dedicated **team**

- ◆ Radek Simko (me)
- ◆ Katy Moe
- ◆ Alex Pilon

→ Dedicated GitHub **repository**

- ◆ `github.com/hashicorp/terraform-plugin-sdk`

Future: SDK



→ Coming Soon:

- ◆ Deprecating 0.11 (protocol 4) support

→ May come later:

- ◆ Provisioners
 - under providers
- ◆ Backends under providers



Thank You

@radeksimko

www.terraform.io

www.hashicorp.com



Please

**Remember to
rate this session**

Thank you!



Did you **remember**
to rate the previous
session ?

