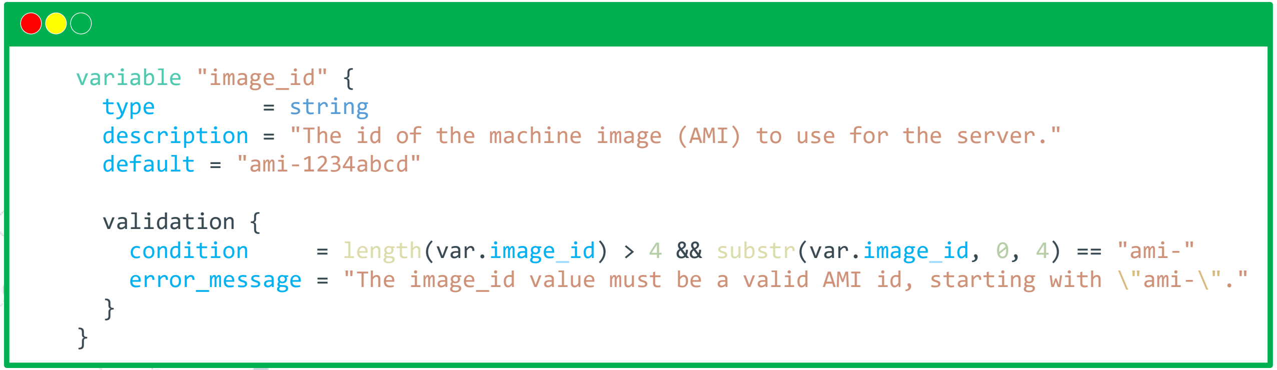
Packer Variables

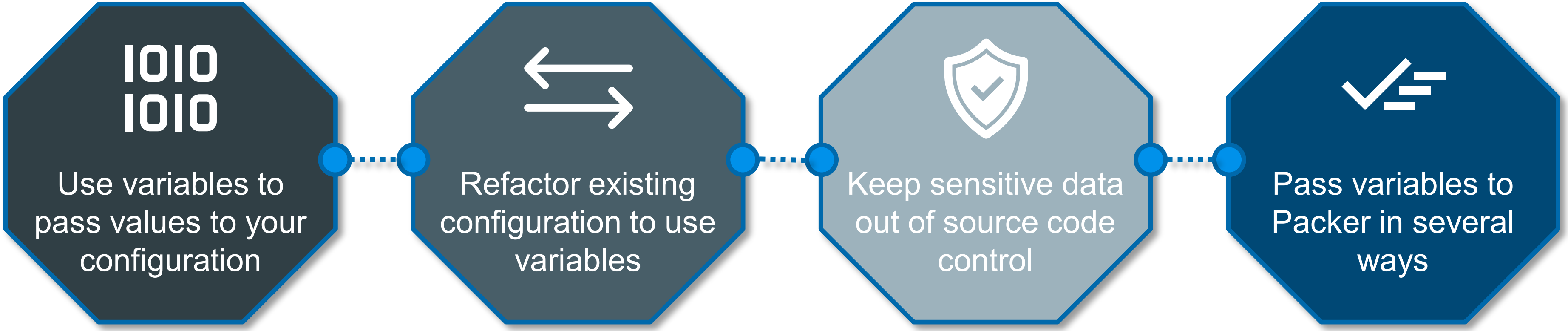
Introduction to Variables

- HashiCorp Packer can use variables to define defaults and values during a build
- Work a lot like variables from programming languages
- Allow you to remove hard coded values and pass parameters to your configuration
- Can help make configuration easier to understand and increase reusability
- Must always have a value. Variables are never optional, but they can have a default value



```
variable "image_id" {
  type        = string
  description = "The id of the machine image (AMI) to use for the server."
  default = "ami-1234abcd"

  validation {
    condition     = length(var.image_id) > 4 && substr(var.image_id, 0, 4) == "ami-"
    error_message = "The image_id value must be a valid AMI id, starting with \"ami-\"."
  }
}
```

What can we do with Packer Variables?



Use variables to pass values to your configuration

Refactor existing configuration to use variables

Keep sensitive data out of source code control

Pass variables to Packer in several ways

Declaring Variables

- Variables can be declared and defined in a .pkrvars.hcl file or .auto.pkrvars.hcl, the default .pkr file, or any other file name if referenced when executing the build.

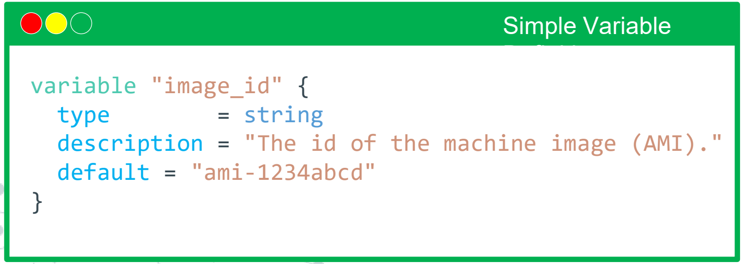
- You can also declare individually using the –var option

- Declared variables can be access throughout the template where needed within expressions

- The type is a constraint, meaning that's the only value that will be accepted for that variable

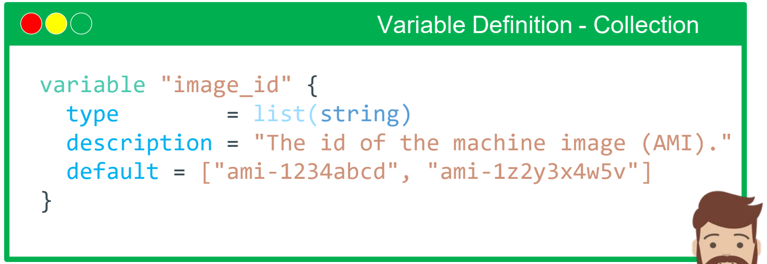
Understanding Variable Types

- The most common variable types are `string`, `number`, `list`, and `map`
- Other support types include `bool` (true/false), `set`, `object`, and `tuple`
- If type is omitted, it is inferred from the default value
- If neither type nor default is provided, the type is assumed to be `string`
- You can also specify complex types, such as collections



Simple Variable

```
variable "image_id" {
  type        = string
  description = "The id of the machine image (AMI)."
  default = "ami-1234abcd"
}
```

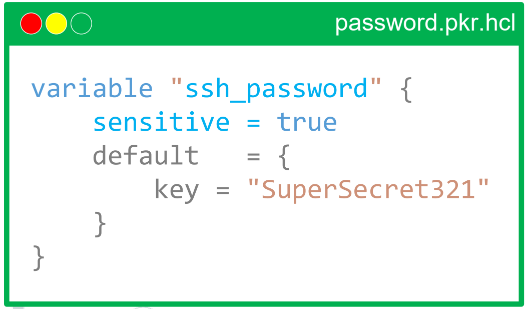


Variable Definition - Collection

```
variable "image_id" {
  type        = list(string)
  description = "The id of the machine image (AMI)."
  default = ["ami-1234abcd", "ami-1z2y3x4w5v"]
}
```

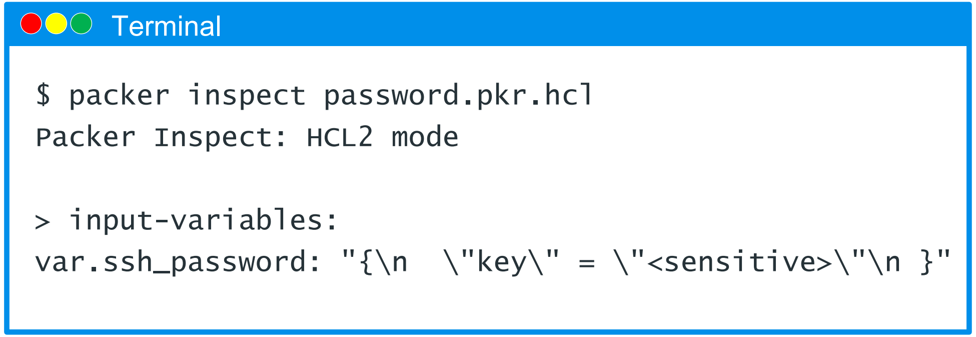
Sensitive Variables

- A variable can be marked as sensitive, if needed
- Packer will obfuscate the value in Packer's output



password.pkr.hcl

```
variable "ssh_password" {
    sensitive = true
    default   = {
        key = "SuperSecret321"
    }
}
```



Terminal

```
$ packer inspect password.pkr.hcl
Packer Inspect: HCL2 mode

> input-variables:
var.ssh_password: "{\n  \"key\" = \"<sensitive>\"\n }"
```

Using Variables in Packer within Expressions

General Referral in Packer

```
var.<name>
```

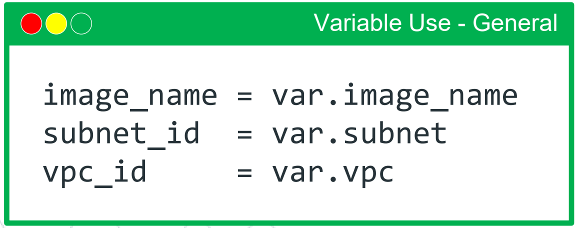
Interpolation within a String

```
"${var.<name>}"
```

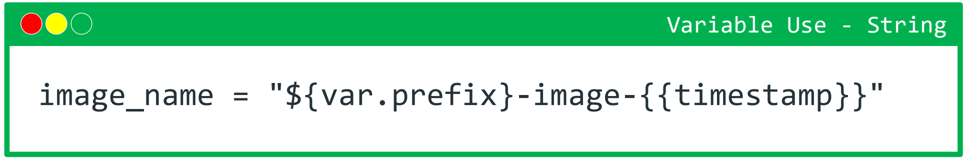
Using Variables in Packer within Expressions

General Referral in Packer

Interpolation within a String



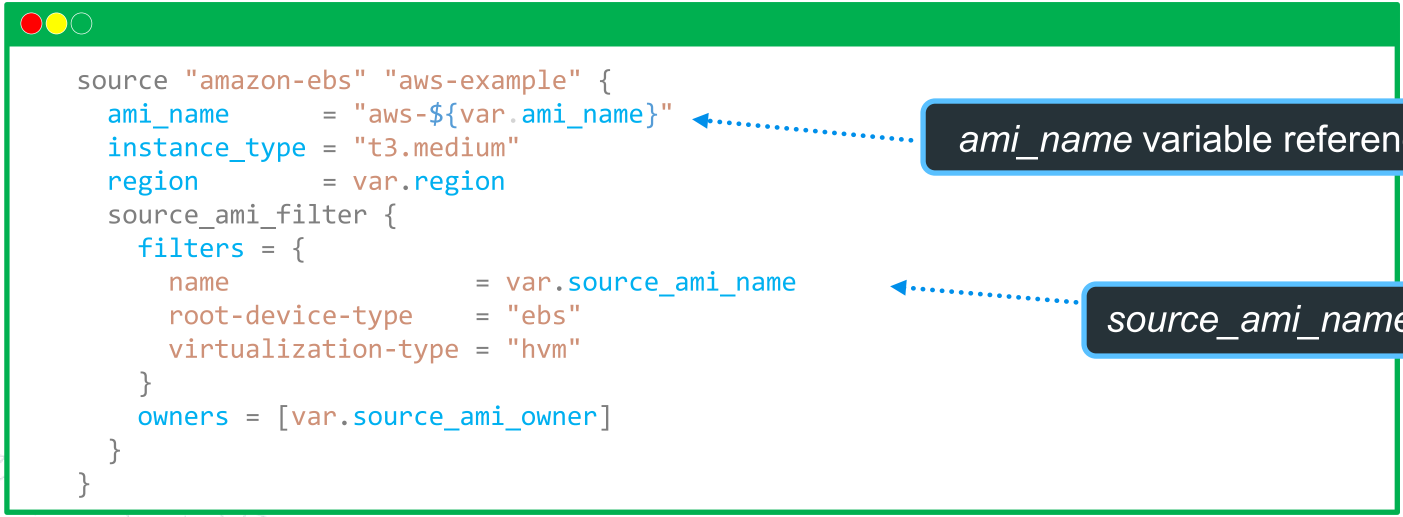
```
image_name = var.image_name
subnet_id  = var.subnet
vpc_id     = var.vpc
```



```
image_name = "${var.prefix}-image-{{timestamp}}"
```

Variables

- Declared variables can be accessed throughout the template where needed
- Reference variables using expressions such as var.<name>or "${var.<name>}"



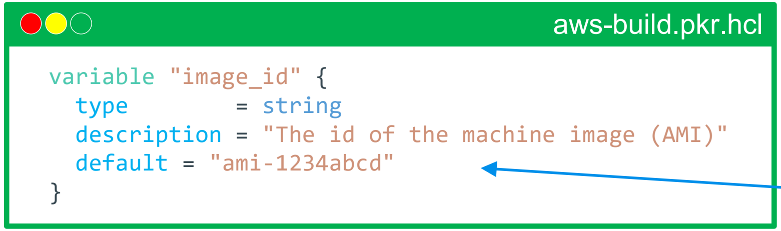
```
source "amazon-ebs" "aws-example" {
  ami_name      = "aws-${var.ami_name}"
  instance_type = "t3.medium"
  region        = var.region
  source_ami_filter {
    filters = {
      name                = var.source_ami_name
      root-device-type    = "ebs"
      virtualization-type = "hvm"
    }
    owners = [var.source_ami_owner]
  }
}
```

ami_name variable reference

source_ami_name variable reference

Variables

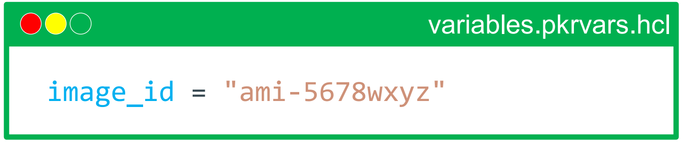
Setting a Value for a Variable


aws-build.pkr.hcl

```
variable "image_id" {
  type        = string
  description = "The id of the machine image (AMI)"
  default = "ami-1234abcd"
}
```

What if we didn't want to use the default value of ami-1234abcd?

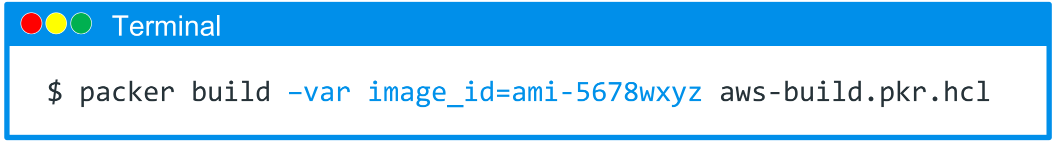
Options to Set Value for Variable:


variables.pkrvars.hcl

```
image_id = "ami-5678wxyz"
```

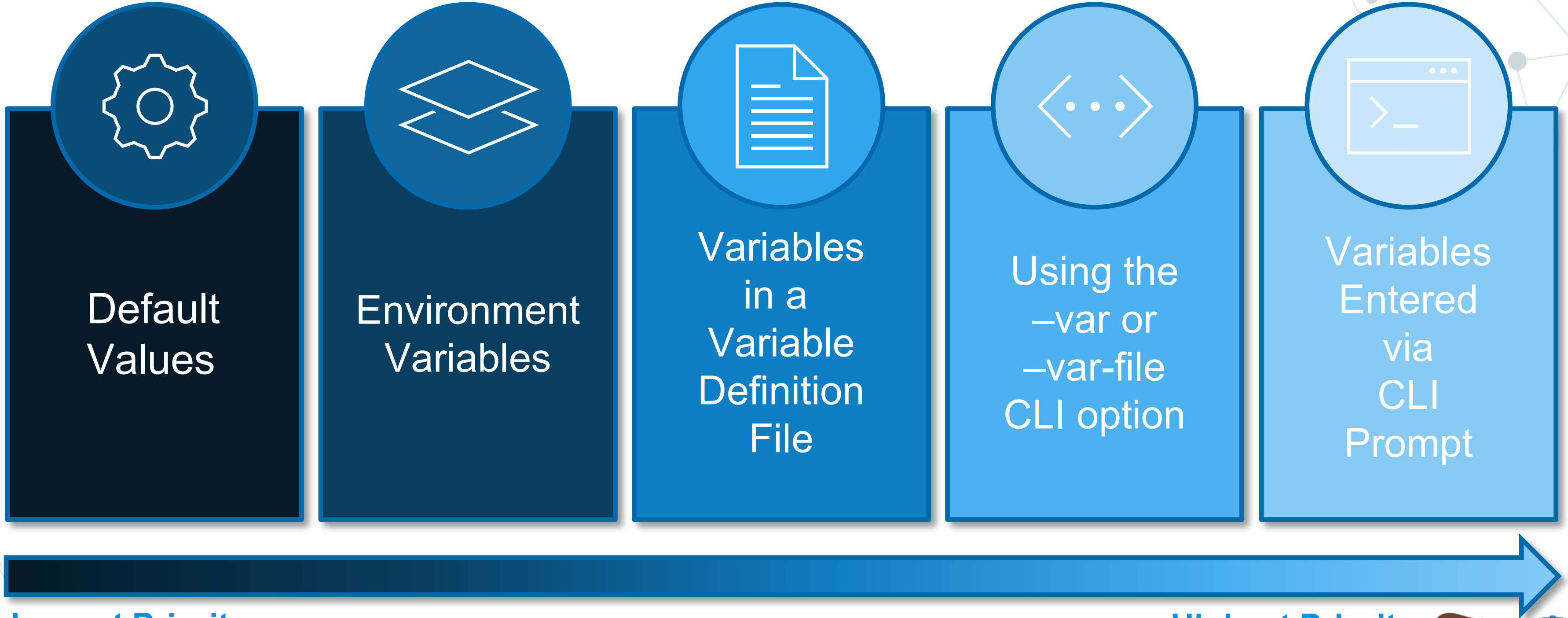
Define the value in another file

Declare the value on the command line during build


Terminal

```
$ packer build -var image_id=ami-5678wxyz aws-build.pkr.hcl
```

Specifying Variable Values (Order of Operation)



Default Values

Environment Variables

Variables in a Variable Definition File

Using the –var or –var-file CLI option

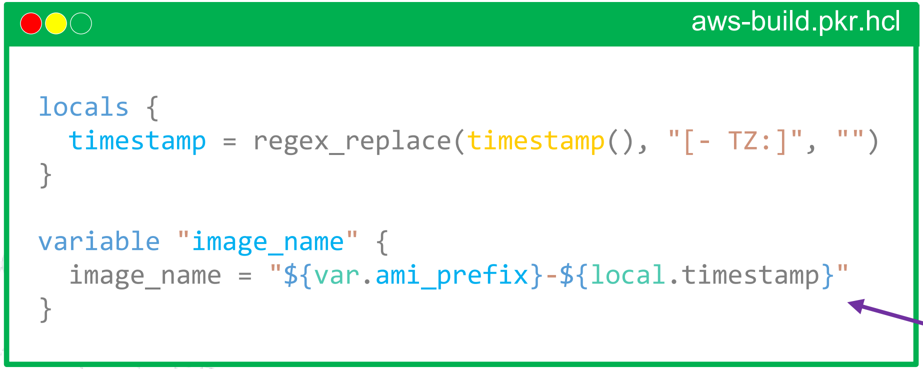
Variables Entered via CLI Prompt

Lowest Priority

Highest Priority

Locals

- Similar to input variables, assign a name to an expression or value
- Locals cannot be overridden at runtime – they are constants
  - Can use a local { } or locals { } block – can mark local as sensitive
  - Using locals { } is more compact and efficient
- Referenced in a Packer file through interpolation – local.<name>



aws-build.pkr.hcl

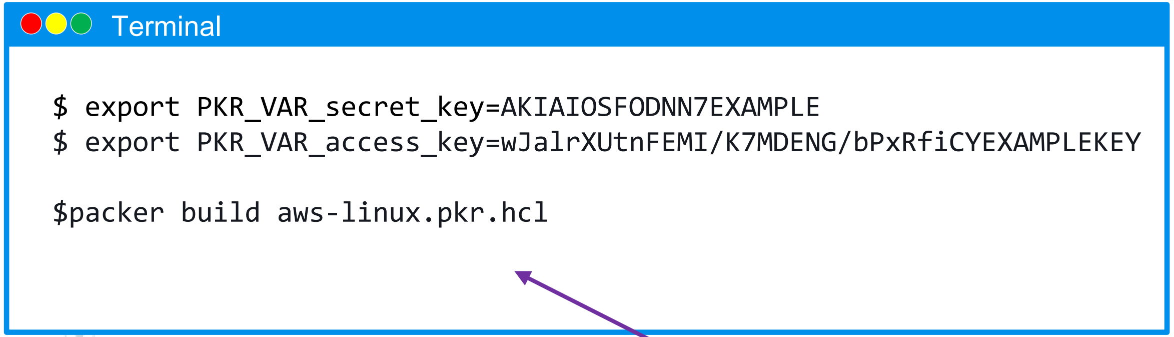
```
locals {
  timestamp = regex_replace(timestamp(), "[- TZ:]", "")
}

variable "image_name" {
  image_name = "${var.ami_prefix}-${local.timestamp}"
}
```

Define a local variable

Reference a local variable

Environment Variables

- Variables can also be set using environment variables
- Great solution for setting credentials or variables that might change often
- Packer will read environment variables in the form of `PKR_VAR_<name>`



```
$ export PKR_VAR_secret_key=AKIAIOSFODNN7EXAMPLE
$ export PKR_VAR_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

$packer build aws-linux.pkr.hcl
```

Set environment variables

Run Packer build that will use ENV

Default Environment Variables

- Packer will automatically define certain commonly used environment variables at build time that can be referenced

`PACKER_BUILD_NAME` – set to the name of the build that Packer is running.

`PACKER_BUILD_TYPE` – set to the type of builder that was used to create the machine

`PACKER_HTTP_ADDR` – set to the address of the http server for file transfer (if used)

END OF SECTION