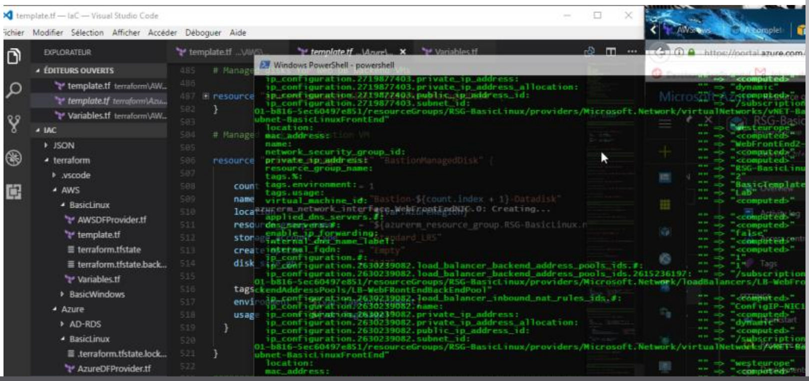


Teknews.cloud

Using Azure Budget to monitor a subscription consumption

David Frappart

01/03/2019

Contributor

Name	Title	Email
David Frappart	Cloud Architect	david@teknews.cloud

Version Control

Version	Date	State
1.0	2019/03/01	Final

Table of Contents

1 The cloud consumption paradigm

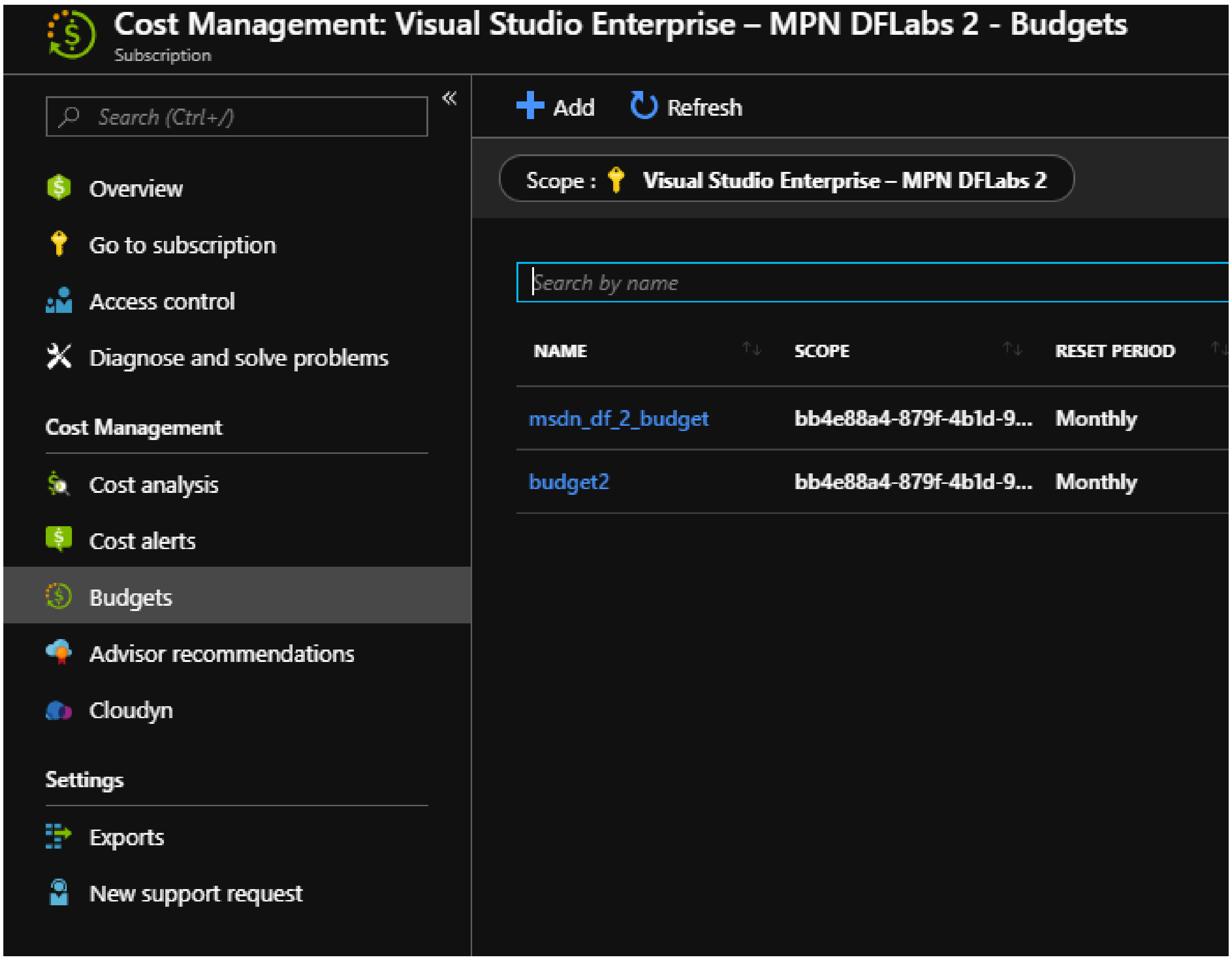
As everyone knows nowaday, one of the main change of paradigm with Cloud Computing is the charge back concept, or pay as you go principles...
Well call it as you want..
From a techical point of view, we have in the NIST definition of a Cloud service the concept of measured service, on which rely all Cloud services providers to, at the end of the day, be able to bill their client.
From a consumer point of view, it can be difficult to keep track of one's consumption, even if the planning was done carefully throug the use of a defined set of tools.
In this article, obviously, we speak about Azure and of one way to easily keep track of how much is consumed for a defined period.

2 A glimpse at the portal

First let's have a glimpse at the portal.

2.1 Azure Budget

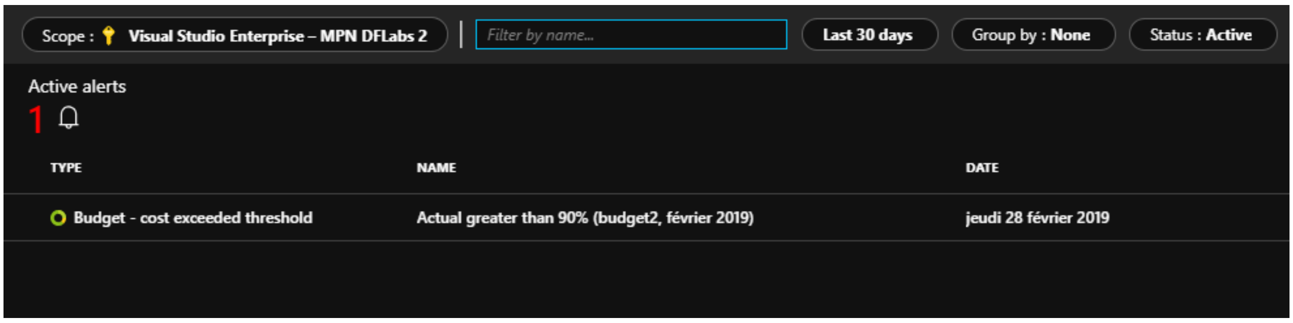
Azure budget is a feature provided in the cost management tool set of the Azure subscriptions.
It's accessible easily from the Subscriptions menu as follow:


Cost Management: Visual Studio Enterprise – MPN DFLabs 2 - Budgets
Subscription

Scope : Visual Studio Enterprise – MPN DFLabs 2

NAME	SCOPE	RESET PERIOD
msdn_df_2_budget	bb4e88a4-879f-4b1d-9...	Monthly
budget2	bb4e88a4-879f-4b1d-9...	Monthly

From a portal point of view, it’s easy to create, and the realted alerts are also visible in the Cost alerts blade:


Scope : Visual Studio Enterprise – MPN DFLabs 2

Active alerts
1

TYPE	NAME	DATE
Budget - cost exceeded threshold	Actual greater than 90% (budget2, février 2019)	jeudi 28 février 2019

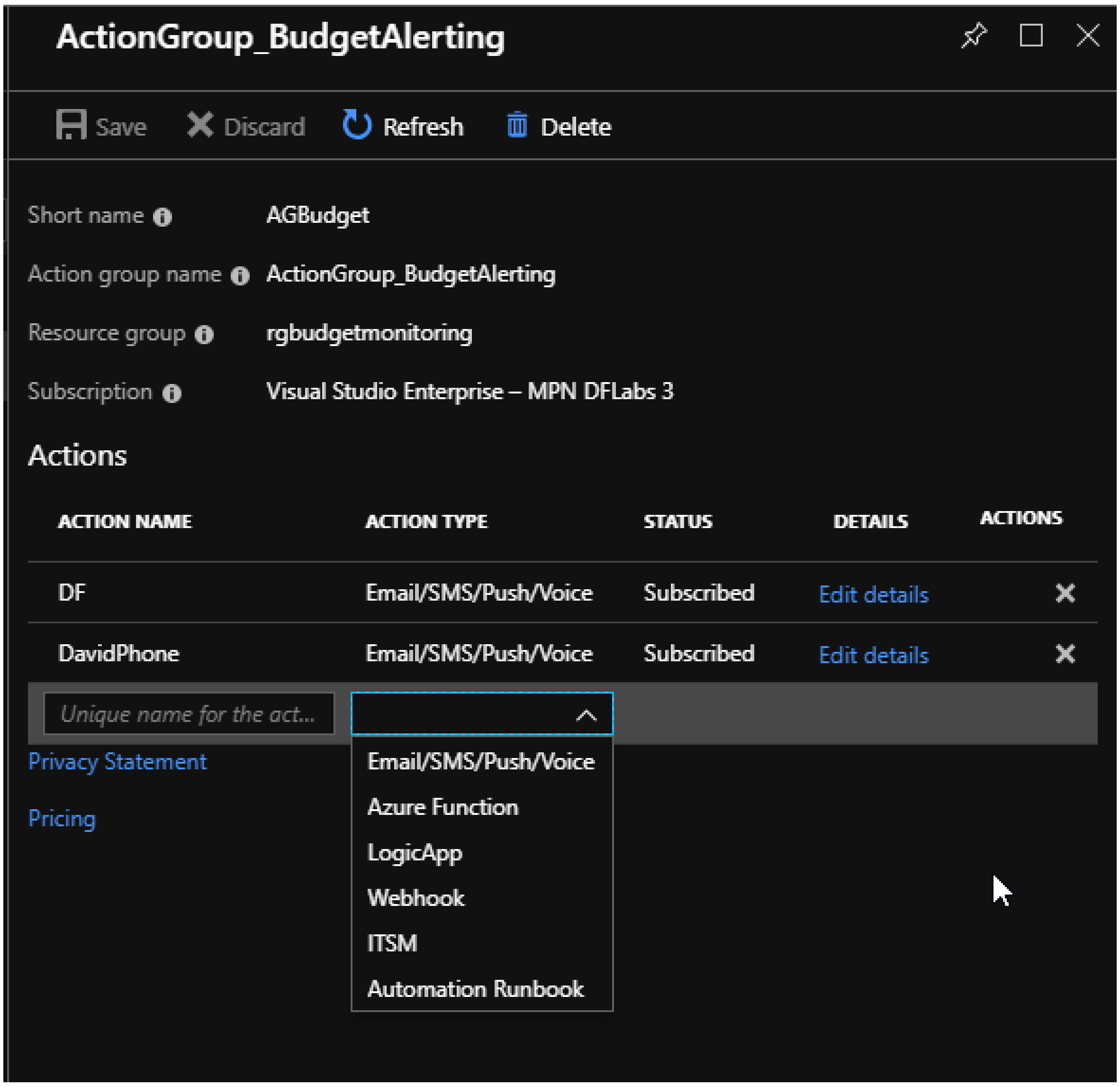
I will not detail how to create a budget in the portal, there are plenty of documentation for this and the process is quite straightforward.
Just remember the following:

- Azure budget can be configured to notify people when a condition is reach. This condition can be a consumption threshold or a percentage of the threshold, for example.

- Azure budget can be configured, in code, to filter the consumption on resources
- Azure budget can be configured to look the consumption monthly, quarterly or annually.
- Azure budget starts its monitoring at the beginning of the month, the format is always YYYY-MM-DD.

2.2 Action Group

Now, before starting some code deployment, let’s have a look a last time to the portal.
As we said, we have the capability of associating the Azure Budget with an action group, which is all the better because it allows us to centralized in one point the notification group in Azure, in the Azure Monitor section.
Also, the action group is richer in terms of notification options than the notification directly in the budget, with options for email, Text messages, webhook and connection to some ITSM, in addition to Azure services such as LogicApp, Azure function and Automation Runbook:


ActionGroup_BudgetAlerting

Short name	AGBudget
Action group name	ActionGroup_BudgetAlerting
Resource group	rgbudgetmonitoring
Subscription	Visual Studio Enterprise – MPN DFLabs 3

Actions

ACTION NAME	ACTION TYPE	STATUS	DETAILS	ACTIONS
DF	Email/SMS/Push/Voice	Subscribed	Edit details	✕
DavidPhone	Email/SMS/Push/Voice	Subscribed	Edit details	✕

And that's all for the GUI, because, GUI is not that fun ☺

3 Deploy with Terraform… and ARM

Now after the overview, it's time to deploy programatically. Since Azure Budget can associate itself with Azure Monitor Action Group, we will create those 2 resources for a test purpose.
At this time, the Azure budget is not available in the Terraform Azure provider, so we will call a little ARM template to workaround this limitation.
Also, be aware that the Azure budget, while available on MSDN subscription cannot be creaed by ARM unless the subscription is part of an EA.
I dit have an error message relating this limitation, along with sometimes a simple 401 Error also stating an unauthorized action:

```
Error: Error applying plan:

1 error(s) occurred:

* azurerm_template_deployment.Template-AZBudgetW: 1 error(s) occurred:

* azurerm_template_deployment.Template-AZBudgetW: Error waiting for deployment: Code="DeploymentFailed" Message="At least one resource deployment operation failed. Please list deployment operations for details. Please see https://aka.ms/arm-debug for usage details." Details=[{"code":"Unauthorized","message":"{\r\n \"error\": {\r\n \"code\": \"401\",\r\n \"message\": \"Unauthorized. Request ID: 78d71523-3d7b-4423-99fd-7a99f8d688c5\"\r\n }\r\n}"}]

Terraform does not automatically rollback in the face of errors.
Instead, your Terraform state file has been partially updated with
any resources that successfully completed. Please address the error
above and apply again to incrementally change your infrastructure.
```

On the other hand it works well on EA subcriptions, so it's still good to go.

3.1 Module for the Action Group

As discussed, this part is the easy part. It's working well in a module with this code:

```
#Creating a Action Group

resource "azurerm_monitor_action_group" "TerraLogTerraActionGroup" {
  name                        = "${var.AGName}"
  resource_group_name         = "${var.AGRGName}"
  short_name                  = "${var.AGSName}"
  enabled                     = "${var.IsEnabled}"

  email_receiver {
    name                = "${var.EmailReceiver1}"
    email_address       = "${var.EmailReceiver1Address}"
  }
/*
  email_receiver {
    name                = "${var.EmailReceiver2}"
    email_address       = "${var.EmailReceiver2Address}"
  }
*/
  sms_receiver {
    name              = "${var.SMSReceiver1}"
    country_code      = "${var.SMSReceiver1CC}"
    phone_number      = "${var.SMSReceiver1Number}"

  }
/*
  sms_receiver {
    name              = "${var.SMSReceiver2}"
    country_code      = "${var.SMSReceiver2CC}"
    phone_number      = "${var.SMSReceiver2Number}"
  }
*/
}
```

You can see that I commented for now the second targeted user of the Action Group.
It is possible to add more than just one so I had this ready for testing.
Something to be aware of, it is not possible to duplicate the target of the notification. If the same phine number is referenced twice, it will generate an error at the apply step.

3.2 Terraform and ARM for the Azure Budget

Now for the Azure Budget, as mentionned earlier, it is not available on MSDN subscription, neither is it at this time available in the Terraform provider. So the work around is to use the terraform resource azurerm_template_deployment. I use also the terraform template provider to hide from my .tf file the JSON part and reference it from a file somewhere else.
On this, a word of caution, template in terraform support only string for interpolation. In this case, i took the quick start template as a reference and this ARM template use an array fo rthe action groups. In Terraform wording, it means it’s a list that should be passed as a parameter and this does not work.
The workaround, because there is one obviously, is to pass a simple string from terraform and use the array() function in the ARM template. We have the parameter as follow:

```
      "contactGroup": {
        "type": "string",
        "metadata": {
          "description": "The action group to send the budget notification to when
the threshold is exceeded."
        }
```

And in the resources part we call the array() function to transform the string in an array:

```
          "notifications": {
            "First-Notification": {
              "enabled": true,
              "operator": "[parameters('operator')]",
              "threshold": "[parameters('threshold')]",
              "contactRoles": "[parameters('contactRoles')]",
              "contactGroups": "[array(parameters('contactGroup'))]"
```

Attentive readers may have noticed the contactRoles parameter reference. This one allows the budget to notify identified people on the subscription depending on their roles in RBAC. It takes also an array but, for this one, I just decided it would be nice to notify systematically the subscription owner, so no change here ☺

For curious, below is the code for the azurerm_template_deployment:

```
data "template_file" "AzureBudget" {
  template = "${file("./Templates/AzureBudget.json")}"
}

resource "azurerm_template_deployment" "Template-AZBudget" {
  name                = "azurebudget"
  resource_group_name = "${module.ResourceGroupMonitoring.Name}"

  template_body = "${data.template_file.AzureBudget.rendered}"

  parameters {
    "budgetName"        = "TestBudget"
    "amount"            = "100"
    "budgetCategory"    = "Cost"
    "timeGrain"         = "Monthly"
    "startDate"         = "2019-03-01"
    "operator"          = "GreaterThanOrEqualTo"
    "threshold"         = "90"
    "contactGroup"     = "${module.BudgetActionGroup.AGName}"


  }

  deployment_mode = "Incremental"
}
```

And the JSON file referenced:

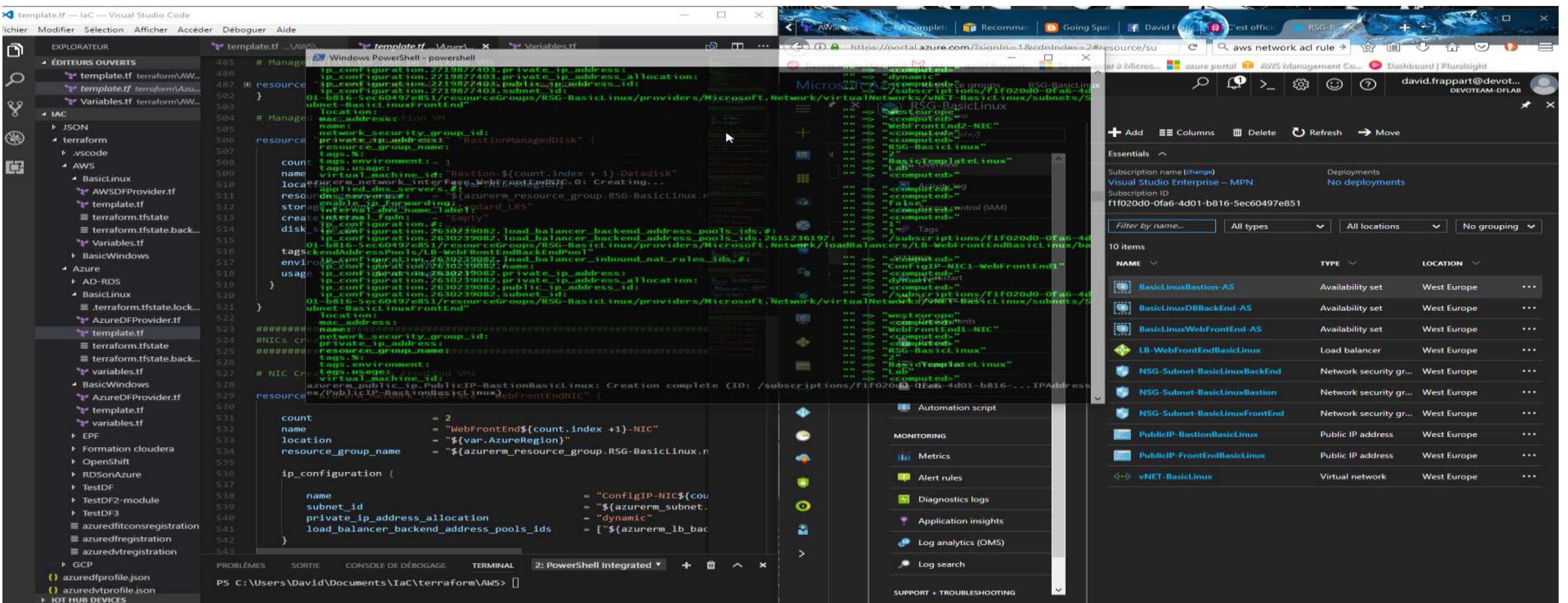
```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
      "budgetName": {
        "type": "string",
        "defaultValue": "MyBudget",
        "metadata": {
          "description": "Name of the Budget. It should be unique within a resource
group."
        }
      },
      "amount": {
        "type": "string",
        "defaultValue": "1000",
        "metadata": {
          "description": "The total amount of cost or usage to track with the
budget"
        }
      },
      "budgetCategory": {
        "type": "string",
        "defaultValue": "Cost",
        "allowedValues": [
          "Cost",
          "Usage"
        ],
        "metadata": {
          "description": "The category of the budget, whether the budget tracks
cost or usage."
        }
      },
      "timeGrain": {
        "type": "string",
        "defaultValue": "Monthly",
        "allowedValues": [
          "Monthly",
          "Quarterly",
          "Annually"
        ],
        "metadata": {
          "description": "The time covered by a budget. Tracking of the amount will
be reset based on the time grain."
        }
      },
      "startDate": {
        "type": "string",
        "metadata": {
          "description": "The start date must be first of the month in YYYY-MM-DD
format. Future start date should not be more than three months. Past start date
should be selected within the timegrain preiod."
        }
      },
      "operator": {
        "type": "string",
        "defaultValue": "GreaterThan",
        "allowedValues": [
          "EqualTo",
          "GreaterThan",
          "GreaterThanOrEqualTo"
        ],
        "metadata": {
          "description": "The comparison operator."
        }
      },
      "threshold": {
        "type": "string",
        "defaultValue": "90",
        "metadata": {
          "description": "Threshold value associated with a notification.
Notification is sent when the cost exceeded the threshold. It is always percent and
has to be between 0 and 1000."
        }
      },
      "contactRoles": {
        "type": "array",
        "defaultValue": [
          "Owner"
        ],
        "metadata": {
          "description": "The list of contact roles to send the budget notification
to when the threshold is exceeded."
        }
      },
      "contactGroup": {
        "type": "string",
        "metadata": {
          "description": "The action group to send the budget notification to when
the threshold is exceeded. It accepts array of strings."
        }
      }
```

```
    },
    "variables": {},
    "resources": [
      {
        "type": "Microsoft.Consumption/budgets",
        "name": "[parameters('budgetName')]",
        "apiVersion": "2018-01-31",
        "properties": {
          "category": "[parameters('budgetCategory')]",
          "amount": "[parameters('amount')]",
          "timeGrain": "[parameters('timeGrain')]",
          "timePeriod": {
            "startDate": "[parameters('startDate')]"
          },
          "notifications": {
            "First-Notification": {
              "enabled": true,
              "operator": "[parameters('operator')]",
              "threshold": "[parameters('threshold')]",
              "contactRoles": "[parameters('contactRoles')]",
              "contactGroups": "[array(parameters('contactGroup'))]"
            }
          }
        }
      }
    ],
    "outputs": {
      "budgetName": {
        "type": "string",
        "value": "[parameters('budgetName')]"
      }
    }
  }
```

4 Conclusion

In this article, we described a simple way to deploy systematically alert on budget thanks to the Azure budget and Action group feature.
For now, it is not completely dry but it should give some help to those who have to manage many subscription and need to have alerting on the consumption.
I did not at this point had a look at the filtering options, but it could be nice to add this kind of granularity. I'll wait for the terraform provider update before doing this kind of thing.
For those interested, the code is available on github here.

Hope you liked it ☺



```
ip_configuration.2719877403.private_ip_address:
ip_configuration.2719877403.private_ip_address_allocation:
ip_configuration.2719877403.public_ip_address_id:
ip_configuration.2719877403.subnet_id:
01-b816-5ec60497e851/resourceGroups/RSG-BasicLinux/providers/Microsoft.Network/virtualNetworks/vNET-BasicLinux/subnets/S
ubnet-BasicLinuxFrontEnd"
location:
mac_address:
name:
network_security_group_id:
private_ip_address:
resource_group_name:
tags.%:
tags.environment:
tags.usage:
virtual_machine_id:
azurerm_network_interface.WebFrontEndNIC.0: Creating...
applied_dns_servers.#:
dns_servers.#:
enable_ip_forwarding:
internal_dns_name_label:
internal_fqdn:
ip_configuration.#:
ip_configuration.2630239082.load_balancer_backend_address_pools_ids.#:
ip_configuration.2630239082.load_balancer_backend_address_pools_ids.2615236197:
01-b816-5ec60497e851/resourceGroups/RSG-BasicLinux/providers/Microsoft.Network/loadBalancers/LB-WebFrontEndBasicLinux/ba
ckendAddressPools/LB-WebFRontEndBackEndPool"
ip_configuration.2630239082.load_balancer_inbound_nat_rules_ids.#:
ip_configuration.2630239082.name:
ip_configuration.2630239082.private_ip_address:
ip_configuration.2630239082.private_ip_address_allocation:
ip_configuration.2630239082.public_ip_address_id:
ip_configuration.2630239082.subnet_id:
01-b816-5ec60497e851/resourceGroups/RSG-BasicLinux/providers/Microsoft.Network/virtualNetworks/vNET-BasicLinux/subnets/S
ubnet-BasicLinuxFrontEnd"
location:
mac_address:
name:
network_security_group_id:
private_ip_address:
resource_group_name:
tags.%:
tags.environment:
tags.usage:
virtual_machine_id:
azurerm_public_ip.PublicIP-BastionBasicLinux: Creation complete (ID: /subscriptions/f1f020d0-0fa6-4d01-b816-...IPAddress
es/PublicIP-BastionBasicLinux)
```

```
    count                    = 2
    name                     = "WebFrontEnd${count.index +1}-NIC"
    location                 = "${var.AzureRegion}"
    resource_group_name      = "${azurerm_resource_group.RSG-BasicLinux.n

    ip_configuration {

        name                                        = "ConfigIP-NIC${cou
        subnet_id                                   = "${azurerm_subnet.
        private_ip_address_allocation               = "dynamic"
        load_balancer_backend_address_pools_ids     = ["${azurerm_lb_bac
    }
```

```
PS C:\Users\David\Documents\IaC\terraform\AWS>
```

Subscription name (change)
Visual Studio Enterprise – MPN
Subscription ID
f1f020d0-0fa6-4d01-b816-5ec60497e851

Deployments
No deployments

10 items

NAME	TYPE	LOCATION
BasicLinuxBastion-AS	Availability set	West Europe
BasicLinuxDBBackEnd-AS	Availability set	West Europe
BasicLinuxWebFrontEnd-AS	Availability set	West Europe
LB-WebFrontEndBasicLinux	Load balancer	West Europe
NSG-Subnet-BasicLinuxBackEnd	Network security gr...	West Europe
NSG-Subnet-BasicLinuxBastion	Network security gr...	West Europe
NSG-Subnet-BasicLinuxFrontEnd	Network security gr...	West Europe
PublicIP-BastionBasicLinux	Public IP address	West Europe
PublicIP-FrontEndBasicLinux	Public IP address	West Europe
vNET-BasicLinux	Virtual network	West Europe