Teleport

Terraform not Jira Tickets

Pass your compliance audits the DevOps way and actually improve your security too

Teleport

Disclaimer

- This advice applies to SOC2 Type II Certifications
- PCI 27001 is about 90% similar
- NIST and SOX are much stricter
- Ticketing Systems are important for Support Teams
- Your IaC Pipelines have powerful API keys that can be exfiltrated if not careful

Hackers don’t care about your change management process

Teleport

Why did we even start doing tickets?

- Because we use Jira for planning
- Because IT/Systems work is considered a service organization
- ITIL Philosophies

Teleport

~~Jira~~ Github Issues for Planning and Change Management

- Planning
  - Assign to project boards (Kanban)
  - Linked with Pull Requests
  - Tag for automation and audit
  - Assign to Milestones for releases
- Change Management
  - PR's = Change Tickets
  - Testing is Automatic
  - **Code Reviews are better than approvals**
  - Auto Roll Back
  - Full Audit Trail

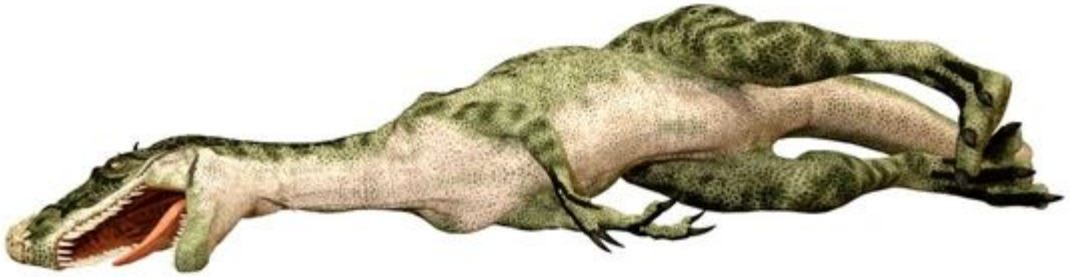
Teleport

~~IT~~ DevOps is a ~~service organization~~ Platform Team

- Every Team included slows the process down
  - Changes should be able to be made by 2 devs on the same team whenever possible
  - Only need approval from a second teams for strictest compliance requirements
- Requests don't scale
  - Good DevOps requires shipping fast
  - Requires 24x365 Support Desk

Teleport

ITIL



shutterstock.com · 1161033826

Teleport

ITIL

- ITIL processes we created to manage error prone human driven processes
- Physical servers required coordination from multiple teams
- Manual Processes need manual approvals, manual testing, manual roll back plans
- The cloud means we can automate so stop following

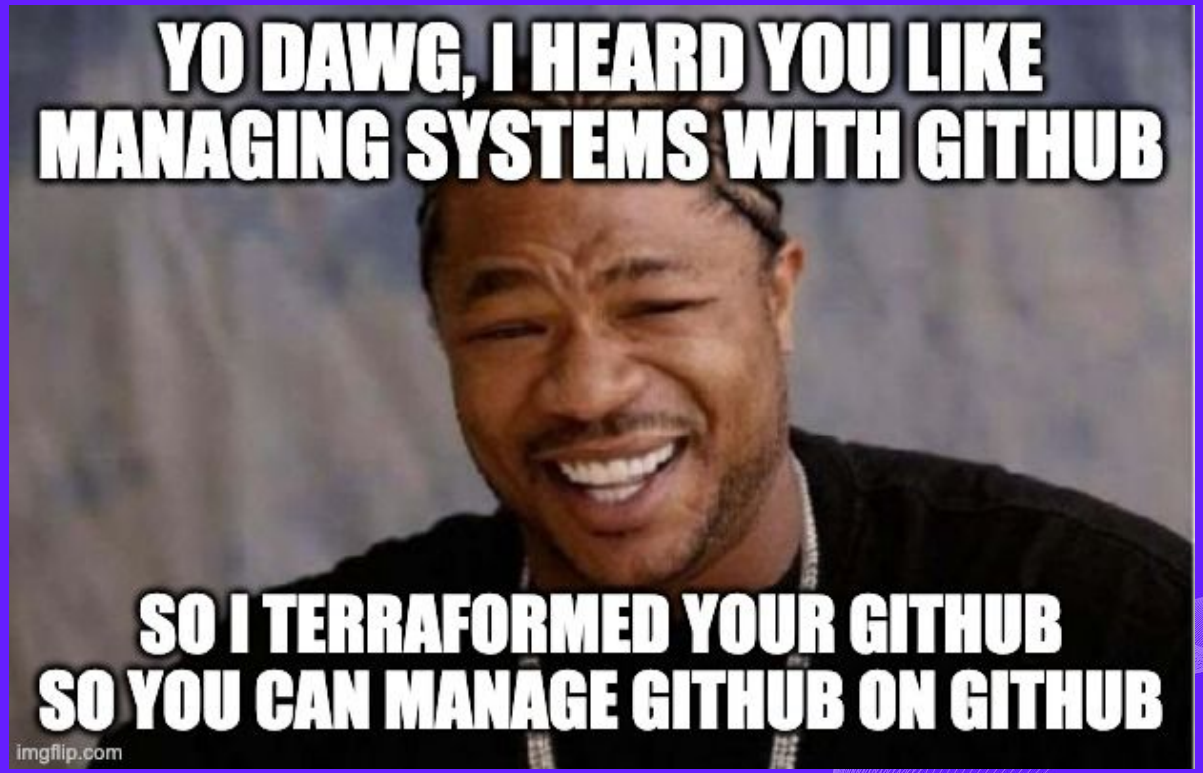
Stop following IT Philosophies from the Pre-Cloud Era

Teleport

Applying IaC to Manage SaaS Apps for Compliance

- IaC systems like Terraform work great for passing compliance audits for infrastructure but they can also be used for managing configuration of critical SaaS apps like Github and Okta which traditionally still needed tickets for manual changes by IT admins

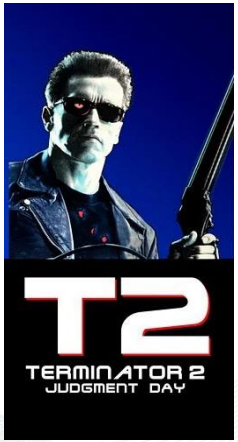
If you use github to manage your infrastructure then a compromised github admin owns your infrastructure


YO DAWG, I HEARD YOU LIKE
MANAGING SYSTEMS WITH GITHUB
SO I TERRAFORMED YOUR GITHUB
SO YOU CAN MANAGE GITHUB ON GITHUB
imgflip.com

Teleport

Applying IaC to Manage SaaS Apps for Compliance

- All Changes Must Come from IaC
- Remove Console Access from the the entire stack including:
  - Okta
  - Github
  - Terraform Cloud



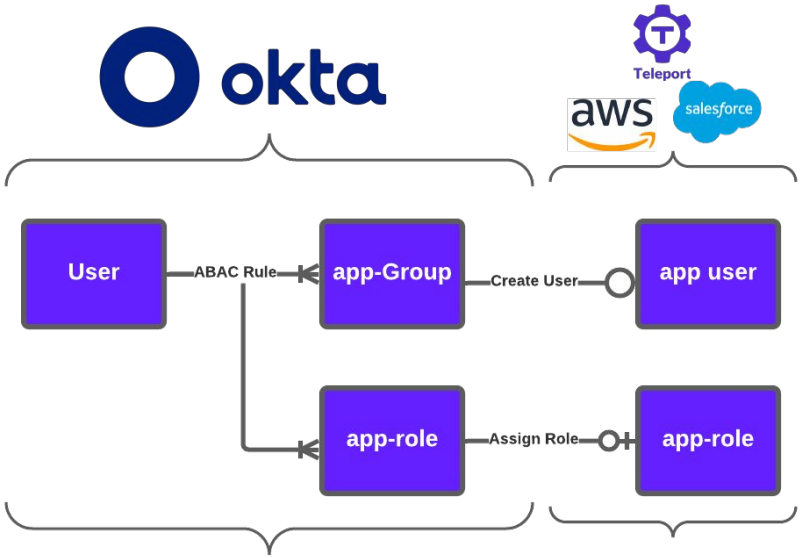
Teleport

Terraforming Okta

Applying ABAC directory rules via Terraform to eliminate Jira tickets for access requests in just 3 easy steps..

Teleport

Step 1: Create a group for every app and role



group-apps.tf

Users > travisgary > group-apps.tf

```
locals {
  apps = {
    "Salesforce"  = { rule = join(" ", [
        "user.department == \"Sales\" OR",
        "user.department == \"Marketing\""
        ])
    },
    "Salesforce-Admin"  = { rule = join(" ", [
        "User.login == \"travis@goteleport.com\" OR", # IT Director
        "User.login == \"henning@goteleport.com\""  # Sales Operations Manager
        ])
    }
  }
}

resource "okta_group" "app" {
 for_each = local.apps
    name        = "app-${each.key}"
    description = "Do Not Edit, RBAC"
    #Do not add users here, use rules only
}

resource "okta_group_rule" "app" {
 for_each = local.apps
    name              = "rbac-app-${each.key}"
    status            = "ACTIVE"
    group_assignments = [okta_group.app[each.key].id]
    expression_type   = "urn:okta:expression:1.0"
    expression_value  = each.value.rule
}

```

Teleport

Step 1: Create a group for every app and role

- Create groups for groups and roles that are not yet managed by code
- Changes to those groups can be used as the request/approval/audit system
- Future Proof for when those apps do support Provisioning/RBAC

Teleport

Step 2: Remove Group Admin Access

- Create a roles for IT users such as IT-Helpdesk-Admin
- Restrict ‘Group-Admin’ permissions to only allow managing groups not managed by Terraform
- All groups that affect access should be in terraform

Teleport

Step 3: Alert on Changes outside Terraform

- Connect Okta to your SIEM (Security Information Events Monitoring) Platform
    - Use Webhooks and Zapier if you don’t have a SEIM
- Write an alert to fire anytime a group change is made by anyone other than the terraform service user

splunk>

panther

Teleport

Step n: Repeat until 100% Terraform Coverage

- Continue to Terraform more resources that admins usually configure via console until all are codified
- Remove console access entirely
- Create a breakglass user for console access
    - Alert in 1password if user credentials accessed
- Alert on Okta changes made outside terraform
- Stop needing change management tickets

Tickets are only for changes made outside of code.

No changes outside code, no tickets.