PLUS3

IT SYSTEMS

Developing Infrastructure for Using Guacamole

Mark Lidd

mark.lidd@plus3it.com

Community Over Code 2023

New Capability: Developer Desktop

- Preconfigured Virtual Desktop for Software Development
  - No need to take time to setup developer environment
    - Software development applications are already loaded
    - Jetbrains IDE, Eclipse, GitHub/GitLab, Visual Studio, AI Plugins
- Accessible from a Browser
  - No special client downloads
- Entirely OpenSource
  - No restrictions due to licensing
- Multiple O/S workstations available
  - Centos7 initially; Other Linux & Windows later
- Platform
  - Instance-based initially
  - Container-based (AKA Kubernetes) later for applications not requiring a full O/S
- Developers find same/similar environment on various projects/networks

PLUS 3 IT SYSTEMS

Simple Concept: Virtual Developer Workstation

Requirements

- Centos7 based virtual workstation
  - Accessible from available windows/linux clients
    - Use RDP wrapped in TLS
  - Preconfigured with “typical” developer apps
    - Eclipse, compilers, editors, Docker, et. al
    - Meets security requirements
- Multi-Cloud “capable”
- Give developers “something”
  - That is easy to use - preconfigured
  - Saves them time from having to configure a workstation themselves
  - Reasonably secure
    - But doesn’t stop them from configuring their unique development environment
- Scale to hundreds of users

Solution

- Build a centos7 Instance
  - Preconfigured w/things a “developer” needs
    - Eclipse, Editors, Desktop
    - Development environment (Java, C++, etc.
    - Use Ansible to configure workstation
  - Set of tools preconfigured to meet security requirements
  - Save developer configuration time
- Access from a web browser
  - Use HTML5 (Most Web clients)
  - Apache Guacamole
  - Considered RDP client with TLS tunnel, but didn’t meet audit requirements

PLUS 3 IT SYSTEMS

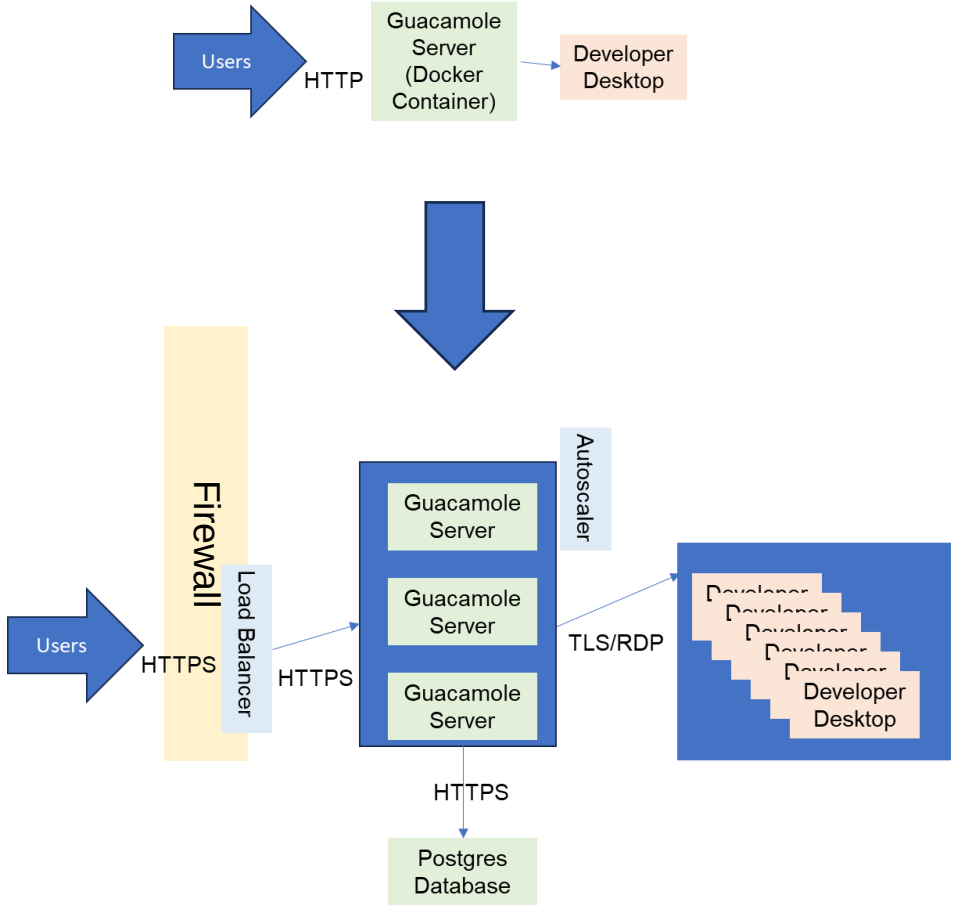
Really want to avoid

OK. I've got this great Open Source product. I've tested it and it works great. What do I do to provide this awesome functionality to my hundreds of users.

- All too typical:
  - Load onto an accessible instance
  - Update instance size when users complain of slowness
  - Reboot if problems
  - Spend inordinate amount of time with users; changing passwords, rebooting, general support, updating public/private keys, etc.
  - "Been there; Done that"
  - Security is an afterthought

PLUS 3 IT SYSTEMS

Talk's Focus



Proof-of-Concept

Other AWS Services:
Cloudwatch: Statistics, Alerts & Alarms
AWS Secrets
IAM
Security
Cloudformation
AWS RDS

Production

Maintainable: secure, key rotation, updates, new versions

AWS focused now but vendor neutral growth

Load balancer does a health check and if failure then Autoscale will in turn fail and regenerate the instance

AWS RDS will automatically backup and update the Postgres database

PLUS 3 IT SYSTEMS

Use a Commercial Product ? Possible But…

- Typical Issues
  - Workstation not general enough of customization not sufficient
  - Not secure enough
  - Needs custom client download
  - License fees
  - Lack of control
  - Not multi-cloud
  - Proprietary

- Our Needs
  - We want to create a paradigm to support ease of use for the developer
  - Want flexibility
  - Not force a paradigm on the developer

PLUS 3 IT SYSTEMS

Guacamole:
Aside from a really yummy dip for chips…

- Guacamole is:
  - A protocol
  - A Clientless Remote Desktop Gateway
    - Clientless: Only a web browser required - no plugins, no software installs
    - Remote Desktop: Support for common remote desktop protocols
      - Kubernetes
      - RDP
      - SSH
      - Telnet
      - VNC
    - Gateway: Web-based, authentication, and permission control

- Started by Michael Jumper and James Muehlner, circa 2010
- Initially developed on SourceForge
- Entered Apache Software Foundation (ASF) “Incubator” project in 2016
- Graduated to ASF Top Level Project (TLP) in late 2017
- Licensed under the Apache 2.0 License

Slide contents provided by Nick Couchman/Apache Guacamole

PLUS 3 IT SYSTEMS

What “Production” means (to me)

OpenSource project provides function “Buffet” to meet production needs (Oauth, OpenID, SAML, etc.)

- Configuration: Understand configuration options
- Traceability
  - Code
  - Glue code
  - Logging
- Cost Control
- Maintainability
  - Public/private keys; UserIDs/passwords
  - Administration
    - Infrastructure
    - Users
    - Support
- OpenSource Software is “Ready to Use”
  - All project CM done by the opensource project
  - Only configuration file(s) need to be generated
- Add “stuff” not supplied by the project repo
  - Site Specific Information & Localization
    - Branding
    - Security needs
    - Infrastructure
- Other Considerations
  - Updates
    - Project software as needed
    - Other software
      - Scripts
      - Downloaded applications
  - Security in general
  - Keep project software load “clean”; no changes
  - UserIDs/passwords, public/private keys secure

PLUS 3 IT SYSTEMS

Utilize AWS for Production Environment Integrations

- Log File Histories:
  - AWS Cloudwatch & AWS S3
- Statistics:
  - AWS Cloudwatch
- Alerting (Alarms & Notifications)
  - AWS Cloudwatch
- Automation: Building & Management
  - CloudFormation
  - AutoScaling
  - Load Balancer
  - Other AWS Infrastructure: Security groups, IAM Policy, VPC/subnets, AMI, instances
- Secrets
  - AWS Secrets

PLUS 3 IT SYSTEMS

Additional Needs for Production (1/3)

- Guacamole needs a datastore to store user & connection data
  - Default is a data file for connection parameters and UserIDs/passwords
    - Other methods supported: we use AWS RDS (Postgres) and OpenID
- Issues Management:
  - Git
  - Software Development Management: Jira
- Manage & store public/private keys:
  - Update & Rotate Keys
  - Includes signed public keys not generally in keystores
  - Key Stored in AWS Secrets
    - Managed access via AWS IAM policies

PLUS 3 IT SYSTEMS

Additional Needs for Production (2/3)

- Congestion Handling
  - AWS Autoscaling
  - AWS Load Balancer
- Create base AMIs
  - Currently: run scripts manually
  - Planned: Create monthly via AWS Lambda
- Configuration Management
  - “Vanilla” repository
    - Scripts, templates, infrastructure (Cloudformation)
  - Site Specific Repository
    - Location Unique identifiers repository (yaml/json files)
- Blob Repository
  - Nexus

PLUS 3 IT SYSTEMS

Additional Needs for Production (3/4)

- Location Unique Credentials:
  - AWS Secrets
  - Scripts to generate/access these Credentials
  - Infrastructure parameters (VPC, Subnet, AMI IDs, IAM Policy. etc.)
- Log File Histories; Statistics; Alerting (Alarms & Notifications)
  - AWS Cloudwatch & AWS S3
  - Use the unified CloudWatch agent
- Automation
  - Building
    - Should be scripted; parameters in yaml file
  - Management

PLUS 3 IT SYSTEMS

Additional Needs for Production (4/4)

- General Security
- Architecture:
  - Isolated with security groups in separate zones/subnets
    - Developer Desktop instances
    - Guacamole & Database
    - Load balancer (Accepts only HTTPS (port 443))
- Isolation via subnets; AWS security groups
- Encrypted data at rest
- Encryption in transit

- AWS SMS
  - Periodic updates; Periodic security scans

TLS Encryption:

Guacamole and Guacd
User and load balancer
Load balancer and Guacamole
Guacd and User's Desktop Instance
Guacamole and database
All AWS services connections

PLUS 3 IT SYSTEMS

Feature Build Breakdown

- Build custom AMIs
  - Create DD AMI w/ needed software for Developer Desktop Instance
  - Create Guacamole AMI w/Guacamole &Guacd
  - Use Hashi's Packer w/ Ansible
  - Template file driven python scripts
    - For DD EC2 deployment
    - For CloudFormation Guacamole deployment

PLUS 3 IT SYSTEMS

Secrets & Management

Secrets

- 6 public/private key pairs to rotate (for TLS)
  - Load Balancer <-> user
  - Tomcat <-> Load Balancer
  - Servlet (guacamole.war) <-> Guacd
  - Servlet (guacamole.war) <-> Database (SQL)
  - Guacd <-> User's Instance (RDP)
  - OpenID server <-> Servlet (Guacamole.war)
- Database UserID/password
- Other tokens/public keys
  - Nexus Blob Repository token (needed during AMI build)
  - AWS public keys
- User Login to Guacamole managed with OpenID server
- Desktop User Logins/Passwords also stored by Guacamole

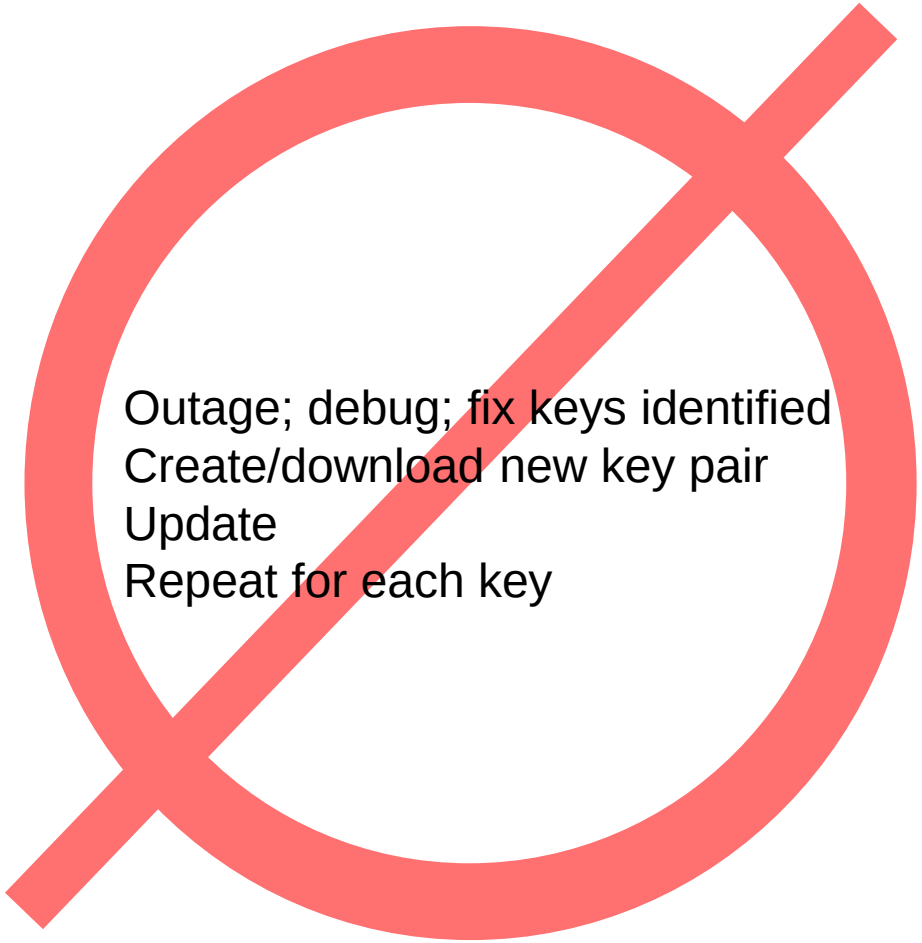
Secrets Management

- Scripts to push keys into AWS Secrets
  - Run as needed
  - Can create self-signed or use signed keys as needed
- Public keys added to keystore during AMI build (Packer/Ansible)
- Private keys read from AWS Secrets using Cloud-Init during instance deploy
- Refresh keys done by
  - Pushing keys to AWS Secrets
  - Rebuilding AMI
    - w/public keys
  - Re-deploying using AWS Autoscaler
    - private keys added

PLUS 3 IT SYSTEMS

Encryption Key Updates

- Public key updates need new AMI
  - Update keys in AWS Secrets
  - Rebuild AMI
- Private key updates need new deploy
  - Update keys in AWS Secrets
  - Redeploy (rolling update)
    - Allows existing connections to drain
    - New connections are routed to new instance



PLUS 3 IT SYSTEMS

Keys/Passwords Scripts: Calls & Returns

- Wrote script to either read keys from files or generate a self-signed key pairs and load keypairs into AWS Secrets

- ./gen-guac-ssk
  - -c <true if create self-signed keys | false to read keys from files>
  - -p < true to push keys and passwords to AWS Secrets manager | false to export keys and passwords to env>
- To rotate keys:
  - Run: source ./gen-guac-ssk -c true -p true
  - Then use AWS Autoscaling to terminate current Guacamole instance and deploy new

PLUS 3 IT SYSTEMS

AMI Generation

- Packer scripts with Ansible plugin
  - Site specific variables defined in Packer HCL yaml file
  - Site specific variables defined in Ansible Vars directory yaml file
  - Parameters (source AMI Id, et. al.) in bash wrapper script

- Separate but similar scripts for Developer Desktop AMI & Guacamole (Guacamole & Guad) AMI

PLUS 3 IT SYSTEMS

AMI Scripts: Calls & Returns

- Developer Desktop AMI

more dd-ami

```
#!/usr/bin/bash
if [[ "x$1" == "x" ]]; then echo "var file not present -exiting"; exit 1; fi
echo "using \"$1\" as packer var file"
read -p "Are you sure? " -n 1 -r
if [[ ! $REPLY =~ ^[Yy]$ ]]; then echo -e "\nresponse must be \"Y\" or \"y\" to continue.  Exiting"; exit 2; fi
echo
#create new version number
MAJOR="0"
MINOR=$(($(cat DD_version)+1))
echo $MINOR >DD_version
PAD=""
if [[ MINOR -lt 10 ]]; then PAD="0"; fi
if [[ MINOR -lt 100 ]]; then PAD="0$PAD"; fi
DD_version="V${MAJOR}.${PAD}${MINOR}"
echo "NEW AMI Version: $DD_version"
#Run Pack
#packer validate -var "ami_force_deregister=true" ami
#get certs for DD nexus repo
. ./get_certs
export DD_ADMIN='DEFAULT_USER'
packer build -var-file "$1" -var "dev_wkstn_version=$DD_version" ami.dd
```

- Guacamole AMI

more dguac-ami

```
#!/usr/bin/bash
if [[ "x$1" == "x" ]]; then echo "var file not present -exiting"; exit 1; fi
echo "using \"$1\" as packer var file"
read -p "Are you sure? " -n 1 -r
if [[ ! $REPLY =~ ^[Yy]$ ]]; then echo -e "\nresponse must be \"Y\" or \"y\" to continue.  Exiting"; exit 2; fi
echo
#create new version number
MAJOR="0"
MINOR=$(($(cat Guac_ami_version)+1))
echo $MINOR >Guac_ami_version
PAD=""
if [[ MINOR -lt 10 ]]; then PAD="0"; fi
if [[ MINOR -lt 100 ]]; then PAD="0$PAD"; fi
Guac_ami_version="V${MAJOR}.${PAD}${MINOR}"
echo "NEW Guac AMI Version: $Guac_ami_version"
#Run Packer
#packer validate -var "ami_force_deregister=true" ami
#get certs for DD nexus repo
pushd .. && . ./get_certs && popd
export DD_ADMIN='DEFAULT_USER'
packer build -var-file "$1" -var "guac_server_version=$Guac_ami_version" ami.guac
```

Note: var file contains files of variables in YAML format
YAML files provide configuration management

PLUS 3 IT SYSTEMS

User Desktop Deployment

- Bash script with call to generalized Python script EC2 deployment
  - Generates the User's instance thru calls to EC2 API (BOTO3)
  - Updates Guacamole database
    - Adds user data
    - Adds associated instance connection data
  - EC2 Deployment is template driven

- Details:
  - Script arguments include instance UserID, Password, public key
  - Reads a template yaml file containing subnet ID, AMI ID, userdata, etc.
  - Creates yaml file with all particulars for visual verification and CM; also logs deploy data for historical record.

PLUS 3 IT SYSTEMS

User Desktop Deployment: Calls

- wrapper script for a python program that deploys an EC2
  - Creates a user's DD instance and updates the guacamole database:
- ./dd-deploy -u user -p password -k "ssh-rsa AAAA...." -a "ami-0000000000000"
- YAML files provide configuration management

- Example parameter file for runec2:
  - more DD_deploy_parameters.yaml.readme
  - REGION : 'us-east-1'
  - EC2-OPERATION : 'create'
  - EC2-PARAMETERS :
  - - SecurityGroupIds :
  - - 'sg-0000000'
  - - SubnetId : 'subnet-0000000000'
  - - InstanceType : 't3.medium'
  - - KeyName : 'my-pem'
  - - ImageId : ''
  - - MaxCount : 1
  - - MinCount : 1
  - - TagSpecifications :
  - - ResourceType : 'instance'
  - Tags :
  - - Key : 'Name'
  - Value : 'DD-'
  - - Key : 'ID'
  - Value : '00001'
  - - UserData : |
  - #!/usr/bin/bash
  - #
  - #
  - #setup named user

PLUS 3 IT SYSTEMS

Guacamole Capacity

- AWS Load balancer scales as needed to handle users volume
- Autoscaling used to create more guacamole instances as needed
  - Average CPU usage > 60% triggers autoscaling up

PLUS 3 IT SYSTEMS

Guacamole Deployment

- Bash script with call to generalized Python script Cloudformation deployment
  - Specifies the Guacamole instance thru autocale template init and userdata Instance
  - Setup up statistics collection, alarms and TBD
  - Parameter driven using yaml/json file to provide site parameters
    - VPC ID, subnet ID, AMI ID, autoscale parameters, IAM policy ID

- Details:
  - Script arguments include CloudFormation update, create, delete.
  - Reads a parameter file containing VPC ID, subnet ID, AMI ID, autoscale parameters, IAM policy ID, etc.
  - Produces log file for record

PLUS 3 IT SYSTEMS

Guacamole Deploy: Calls & Returns

- Deploy cloudformation template:

python runstk.py --log-level DEBUG -pf parameters-guac-server.yaml

Sample parameters-guac-server.yaml:

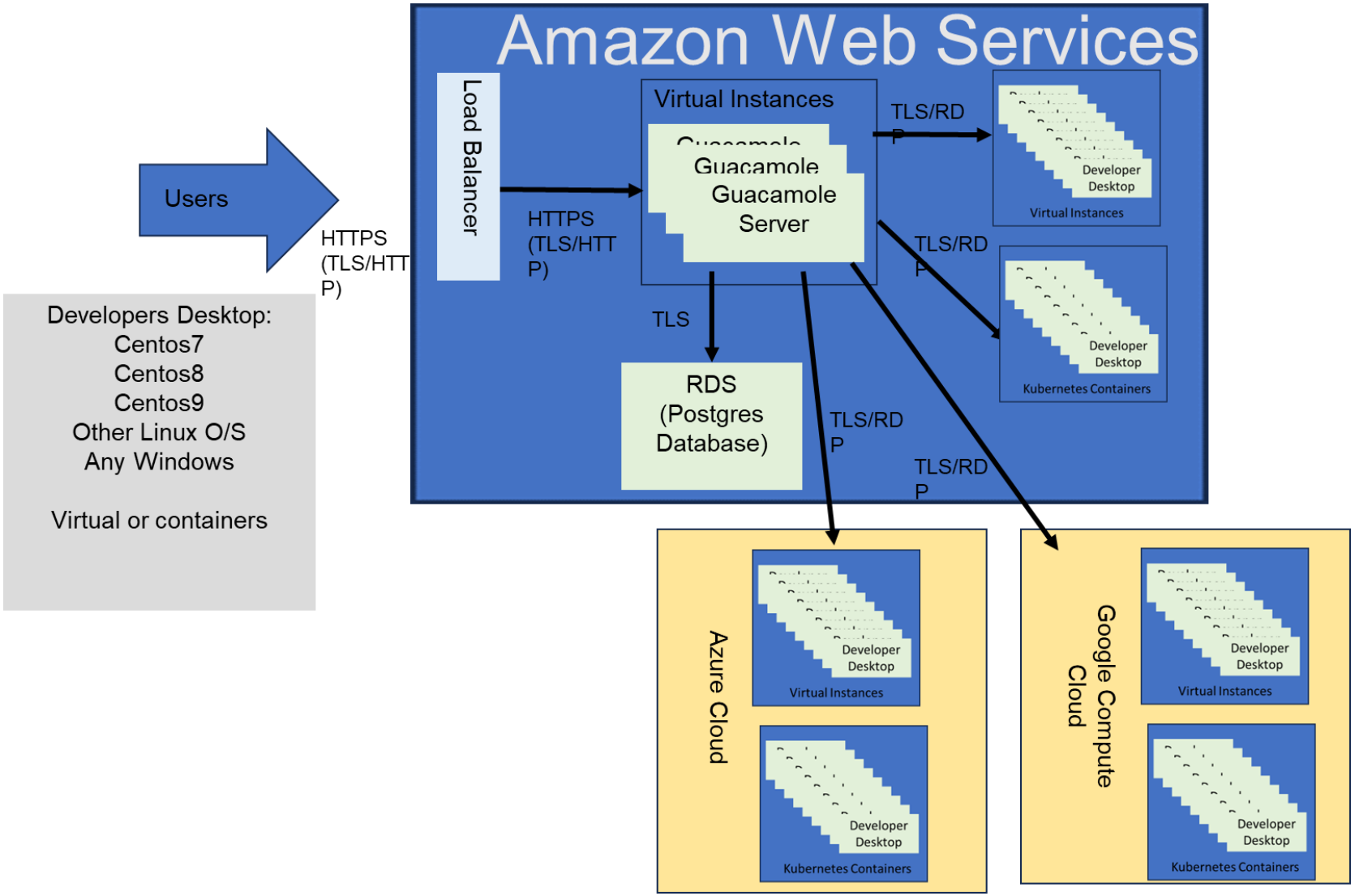
```
Region : 'us-east-1’
StackName : 'My_Stack’
DisableRollback : 'True’
CAPABILITIES : 'CAPABILITY_IAM’
TEMPLATE-FILE : 'my_cfn.yaml’
TEMPLATE-OPERATION : 'create’
STACK-PARAMETERS :
 - VPCId : 'vpc-000000000’
 - VPCIdCidr : "10.10.10.0/24“
 - SubnetID : "subnet-0000000000000“
 - SecurityGroups : 'sg-000000000000, sg-000000000001, sg-0000000000002’
 - SystemsManagerAccess : 'false’
 - KeyName : "my-pem“
 - RedirectURL : 'internal-dev-00000000000.us-east-1.elb.amazonaws.com’
 - RedirectURLPort : ‘’
 - InstanceType : "t3.med"
```

Planned

- Currently all scripts return error codes upon exit. But more is needed
  - Functionality tests needed to determine validity
    - DD AMI; Guacamole AMI
    - DD instance after deployment
    - Guacamole instance after deployment
      - Currently LB does a simple HTTPS ping but doesn't test functionality
      - Test Guacamole/Tomcat functionality
      - Test Guacd functionality
- Other Developer instances
  - Other versions of Linux instances; windows versions
  - Other versions of K8s O/S instances

PLUS 3 IT SYSTEMS

Next Generation Architecture



PLUS 3 IT SYSTEMS

Developing Infrastructure for Using Guacamole

Thank You

Questions?

PLUS 3 IT SYSTEMS