Securing your AWS Infrastructure

"accelerated" short edition

Derek Hill – MBA, CISSP

CSO Security Manager – HP Inc.

derek@dh-solutions.com

@secureITtoday

Teaching SANS MGT 414 (CISSP Prep) this summer:

http://www.sans.org/event/53700

Presentation(s) can be found here

https://github.com/derekhillhp/AWS-Security-Class

Objectives

- A big picture view of security in AWS (30,000 ft vs. 3 ft)
  - Don’t worry, some links are included to dive deeper into subject areas
- Give you a better understanding on the various AWS technologies
- How to secure the various technologies?
  - Going to address the big ones
  - Just scratching the surface here
  - Give you real world examples and hopefully some ideas on how you can implement them in your organization
  - Not all issues are technical
- Have a discussion
  - Not meant as a lecture, but rather collaborative

Agenda

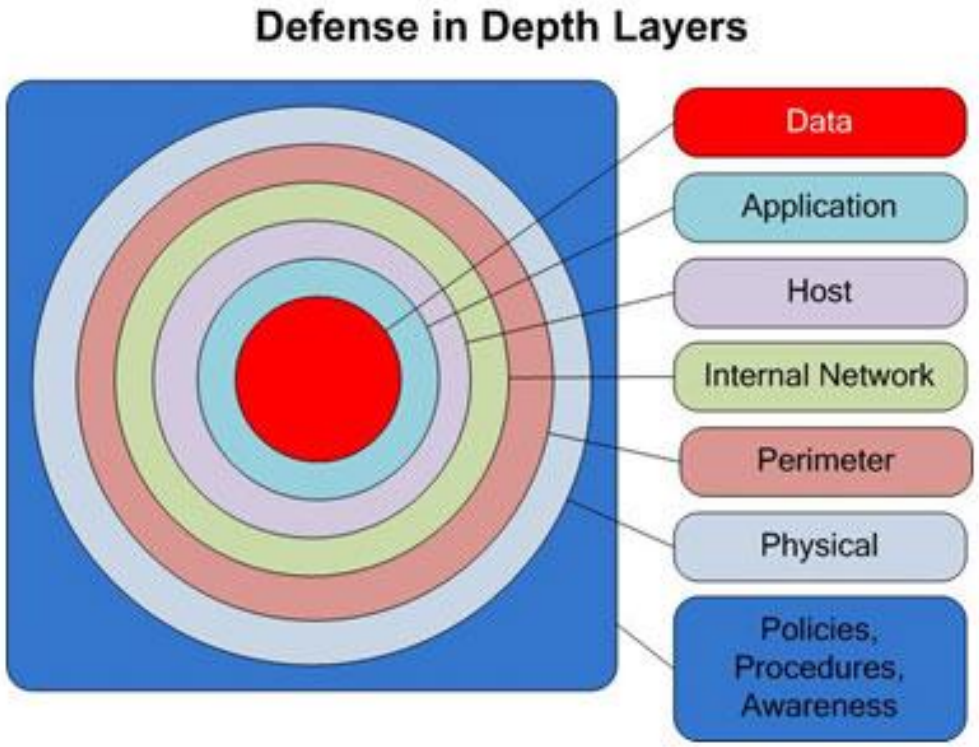
- Concepts
- Instances
- Account Management
- Logging
- Tools
- Vulnerability scanning / pen testing
- Software
- Data Security

Concepts

Laying a basic foundation

Defense in Depth

- Layers of security
- Have defenses in more than one layer
- Don’t put all your eggs in one basket – what happens when that defense fails?
- Applies to the cloud, however, the terminology is different
- Can’t implement all layers as you don’t have access to all of them – think physical layer in AWS or shared responsibilities
- Limit the blast radius, keep your applications / networks compartmentalized, the smaller the better and require authentication to move between compartments



Roles and Responsibilities

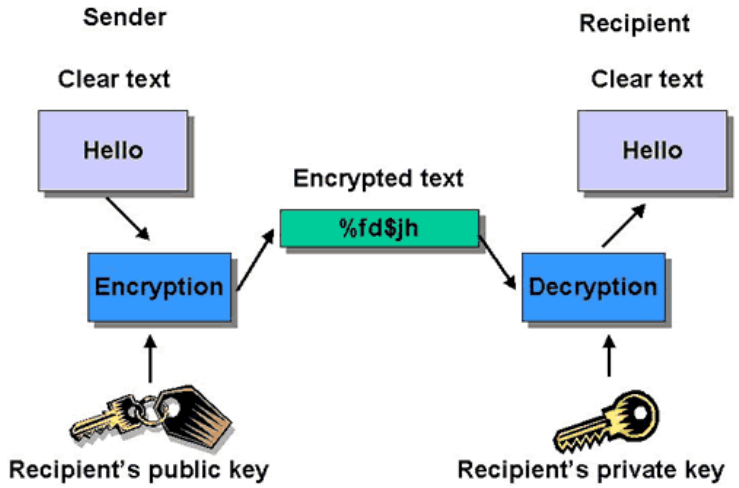
- Have personnel defined for certain tasks, hold them accountable and give them authority
  - A person responsible for the security architecture
  - A person responsible for vulnerability management
  - etc……
- Implement peer review for changes
  - Changes are easy and often the left hand doesn't know what the right hand is doing, especially the case with distributed teams (more on next slide)
- Have incident handling plan (out of scope for today)
  - IR/IH is an entire topic by itself, too much to cover here
- Generally try to understand and follow practices around team organization for your particular industry/project type
- Adopt a RACI model – clarify who is responsible for what
  - https://en.wikipedia.org/wiki/Responsibility_assignment_matrix

Change Management and the challenges

- Even more important in the cloud as it is extremely easy to make changes
- Changes are almost too easy, how do you keep things under control
- How do you handle distributed teams? Establish and maintain good relationships with remote/distributed teams.
- If you have to follow compliance rules, how do you know what you have? Who made what change? How long has it been this way? ... [This might become a much larger issue in the not so distant future]
- Even if you maintain your environments with automation, who is making changes to those scripts/tools? Are those changes tested?

Encryption

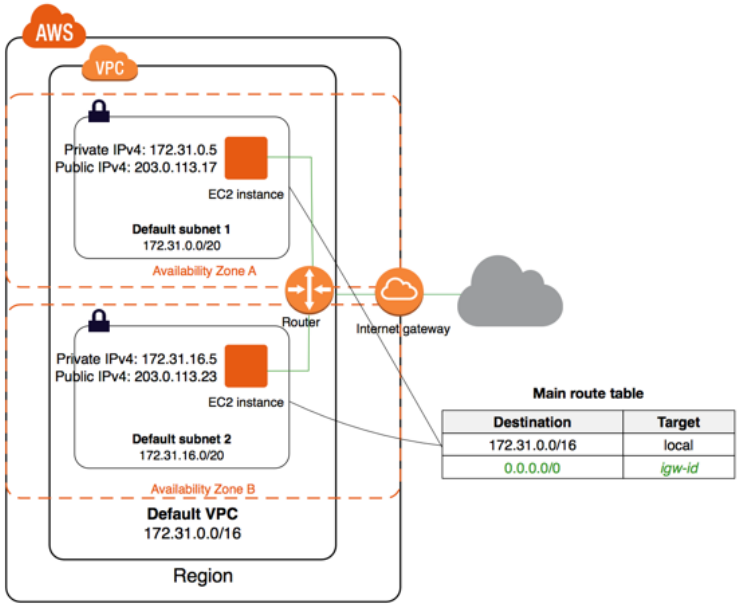
- Always use “open” encryption such as AES-256, SHA-2, RSA 4096 bits or what you are required by your company, gov’t agency, compliance, etc.
  - Never roll your own encryption, you want something that has been publicly vetted for a decade or more
- Encryption is only as good as the key
- It always comes down to key management, where is it stored, who has access to the key(s)
- Encryption at rest – disk encryption
- Encryption in motion – transport encryption (SSL/TLS) – SSL is broken, use TLS 1.1 or later
- When in doubt, encrypt – err on the side of security vs. performance, nobody has been sued for negligence for encrypting data vs. unencrypted data – make sure you use strong encryption and good key management

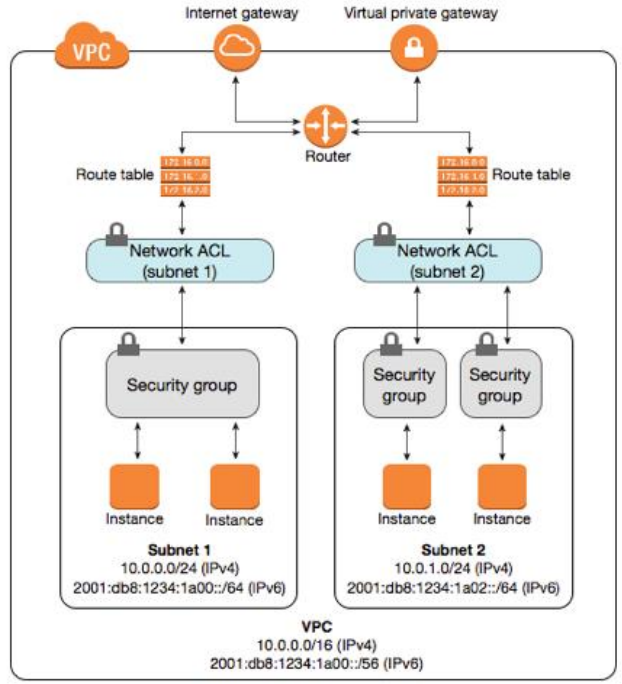


Virtual Private Cloud (VPC)

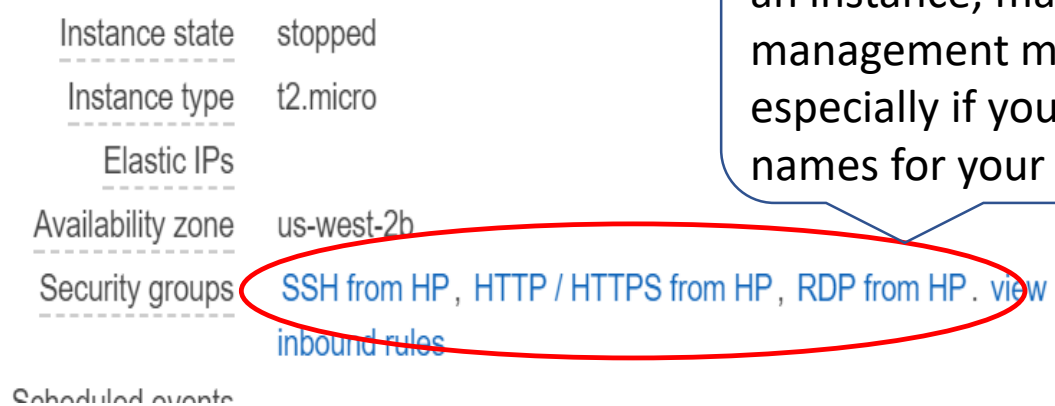
- What is a VPC?
  - Think of it as a VLAN if you need a traditional network reference
  - You can create multiple isolated subnets inside a VPC
  - A VPC can span multiple Availability Zones (AZ), this can provide you with local redundancy
  - A VPC cannot span multiple regions, i.e. US-West-2 and US-East-1 cannot share a VPC
    - https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Introduction.html

- How to secure VPC's?
  - Use security groups (stateful) to control INSTANCE ingress and egress of your VPC, you can also use ACL's (stateless) if you want to control access at the SUBNET level
  - Use VPC flow logs to collect traffic coming and leaving your VPC (similar to spanning in a traditional network) – you can send those to an IDS
    - https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Security.html
  - Enable CloudTrail to monitor changes to VPC
  - Better to have more and smaller VPC's, rather to have all your resources in a large VPC (blast radius)





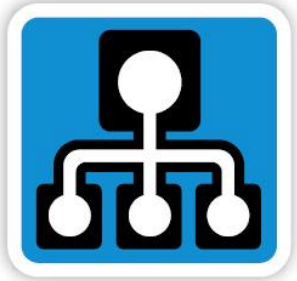
Security Groups (SG)



You can add multiple SG's to an instance, makes management much easier, especially if you descriptive names for your SG's.

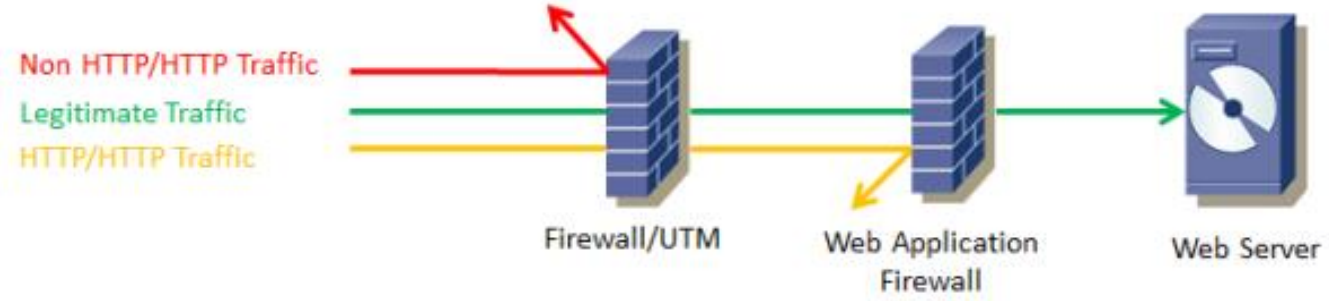
- Basically a Stateful firewall
- Ingress (inbound) and egress (outbound) filtering at the instance level
- Easily configured – almost too easy
- Need to be continually reviewed for effectiveness (remember submitting request for firewall changes)
- Create security groups for individual protocols/services, layer multiple groups onto an instance – it will help you see what is applied to each instance, without having to dig into each group
- Remove the launch-wizard groups, not descriptive at all and often wide open
- Label all your groups as to what type of traffic is allowed and where it is coming from

Load Balancers (LB)



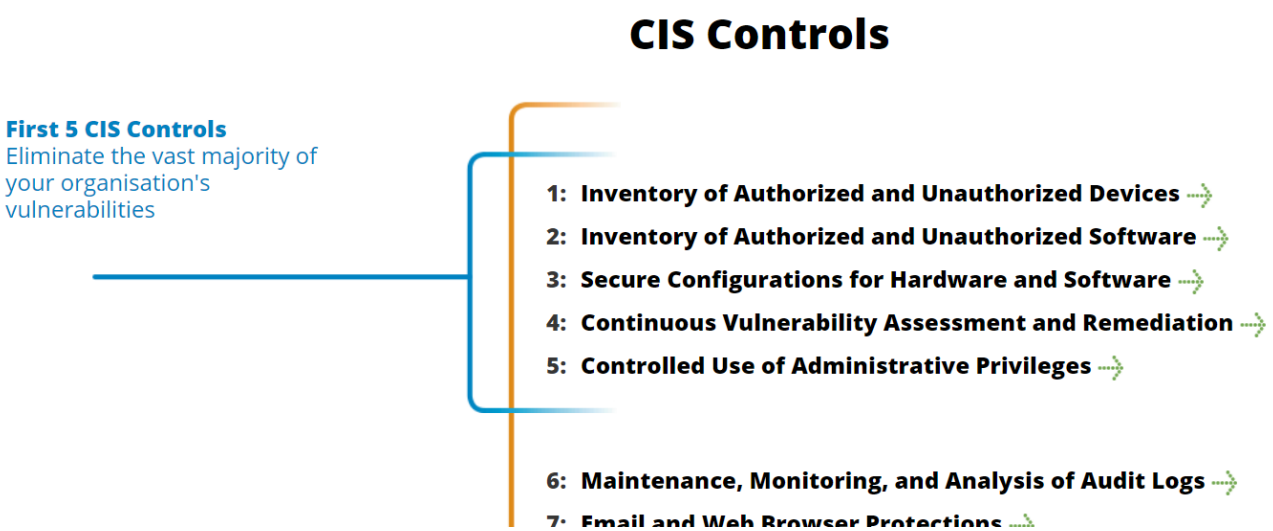
- Elastic Load Balancer (ELB) [now renamed to Classic] Layer 4 & 7 (X-headers) – Terminate or Pass-through
- Application Load Balancer (ALB) Layers 7 – ELB with a WAF integrated and more – Terminate Only, option to re-encrypt for last leg
- Network Load Balancer [released 6 months ago (Sep 7) – not covered here]
- DNS Load Balancing (DNS failover)
  - https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html
  - https://aws.amazon.com/elasticloadbalancing/details/#compare
  - https://aws.amazon.com/blogs/aws/new-network-load-balancer-effortless-scaling-to-millions-of-requests-per-second/

Web Application Firewall (WAF)



- Firewall for HTTP applications that operates at the Application layer of the OSI model (Layer 7)
  - It is designed to block and filter out undesirable traffic between the HTTP client and server – based on rules you apply
- Often considered a reverse proxy that protects servers, not clients
- AWS has WAF offerings, however, you have to create your own rules – often a trial and error
  - Sample rules: https://github.com/awslabs/aws-waf-sample
  - AWS WAF with OWASP Top 10 rules: https://aws.amazon.com/about-aws/whats-new/2017/07/use-aws-waf-to-mitigate-owasps-top-10-web-application-vulnerabilities/
- OWASP provides a basic set of rules free of charge:
  - https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project
- When initially implementing a WAF, do not put it into blocking mode, otherwise some part of the site may be broken.

Critical Security Controls



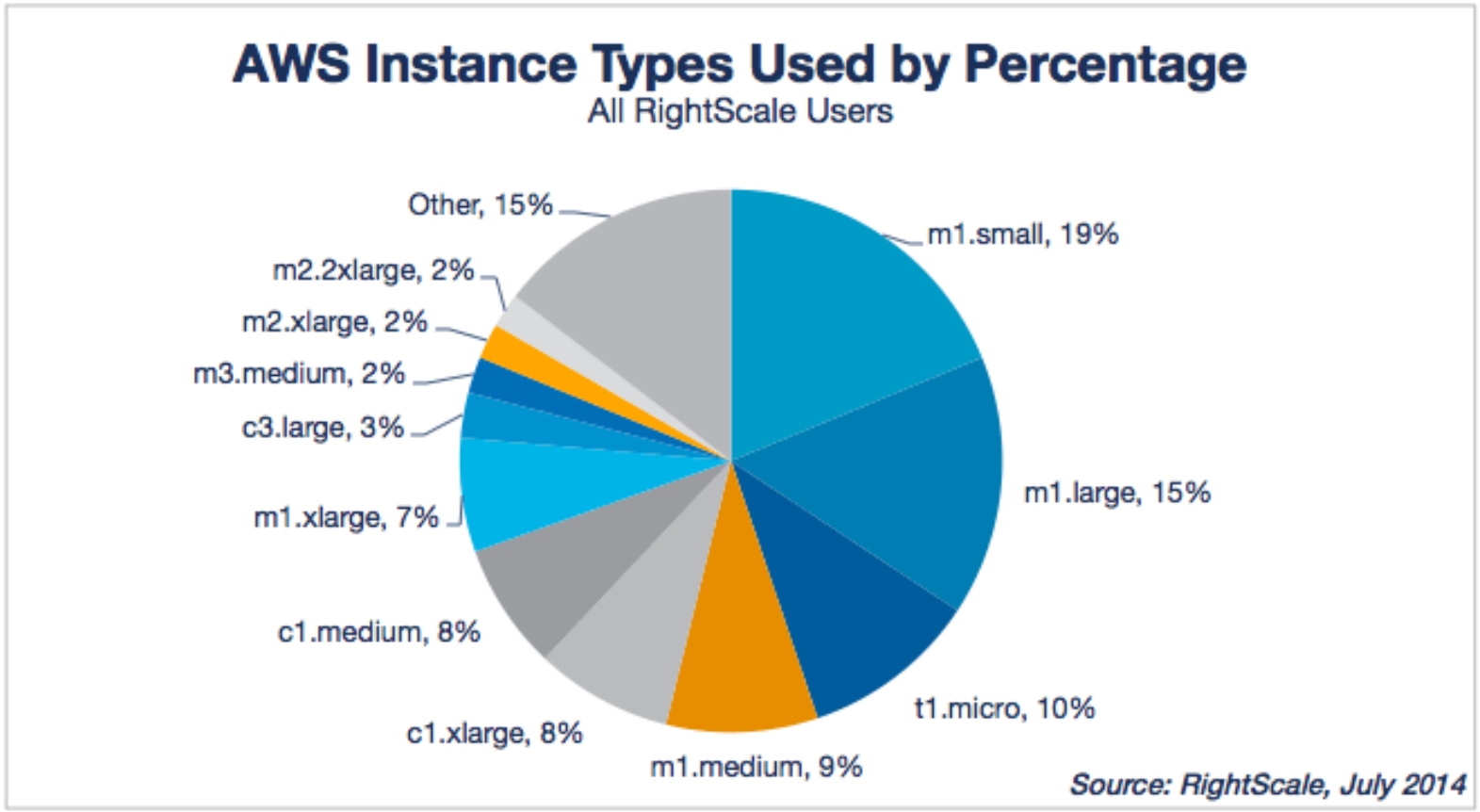
- Total of 20 controls
- Usually updated every 12-18 months
- Most of the controls can be automated
- Implementing the top 5 will get you into pretty good shape – not perfect, but you will protect against the most common types of attacks (85-95% depending on who you ask)
- https://www.cisecurity.org/controls/
- https://www.sans.org/critical-security-controls/guidelines
- https://www.sans.org/media/critical-security-controls/critical-controls-poster-2016.pdf

CSC – Controls 6-20

6: **Maintenance, Monitoring, and Analysis of Audit Logs**
7: **Email and Web Browser Protections**
8: **Malware Defenses**
9: **Limitation and Control of Network Ports**
10: **Data Recovery Capability**
11: **Secure Configurations for Network Devices**
12: **Boundary Defense**
13: **Data Protection**
14: **Controlled Access Based on the Need to Know**
15: **Wireless Access Control**
16: **Account Monitoring and Control**
17: **Security Skills Assessment and Appropriate Training to Fill Gaps**
18: **Application Software Security**
19: **Incident Response and Management**
20: **Penetration Tests and Red Team Exercises**

Instances

VM’s in AWS



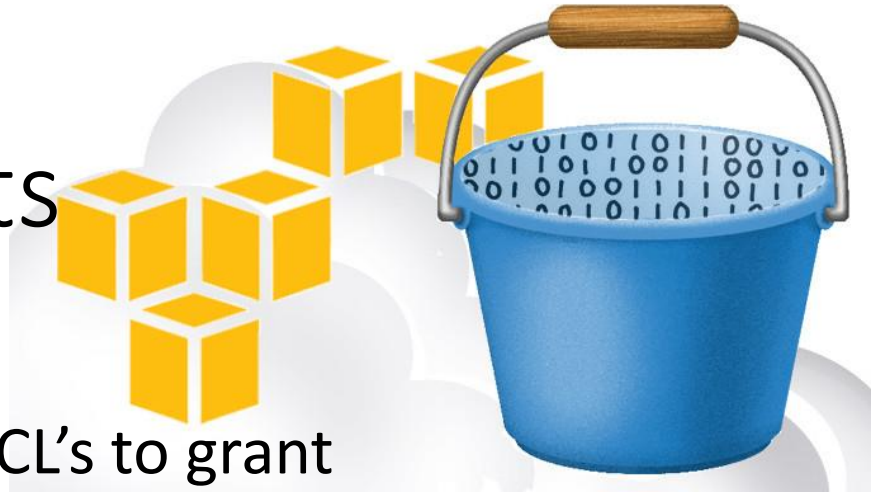
Admin Access

- Some instances are under your full control, you have full admin rights (root) – these should be the easiest to secure …. or are they?
- Some instances are under somebody else's control inside your AWS account – your key is not installed, however, somebody in your org does have root – How do you know what is installed, what that instance is doing, etc?
- Are you using 3rd party instances? For example, Virtual Appliance, Vendor provided AMI.
- Many AWS services don't give you root access, think RDS, RedShift, etc.
- Remove / disable user keys from instances once they leave the company

Encrypted Volumes

- As of recently you can now encrypt your root volumes in Elastic Block Storage (EBS) (didn't use to be the case) – might have to go rebuild older instances that were built previously
- You can have Amazon manage the key or you can manage it, you decide
  - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html
- Who has control over the key vs. the increased complexity

Simple Storage Service (S3) Buckets



- Enhance default security by using bucket policies and ACL's to grant permission
  - https://docs.aws.amazon.com/AmazonS3/latest/dev/s3-access-control.html
  - These can be scripted using JSON and applied to multiple buckets quickly to deploy an organizational policy
- Encrypt your S3 buckets using either Amazon's encryption options or manage it yourself (server side vs. client side)
  - Pick the one that fits your risk profile and your organization guidelines
  - https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html
- Consider using VPC Endpoints to access your S3 buckets. This will restrict access to only certain systems can access your S3 buckets.
  - https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-endpoints.html

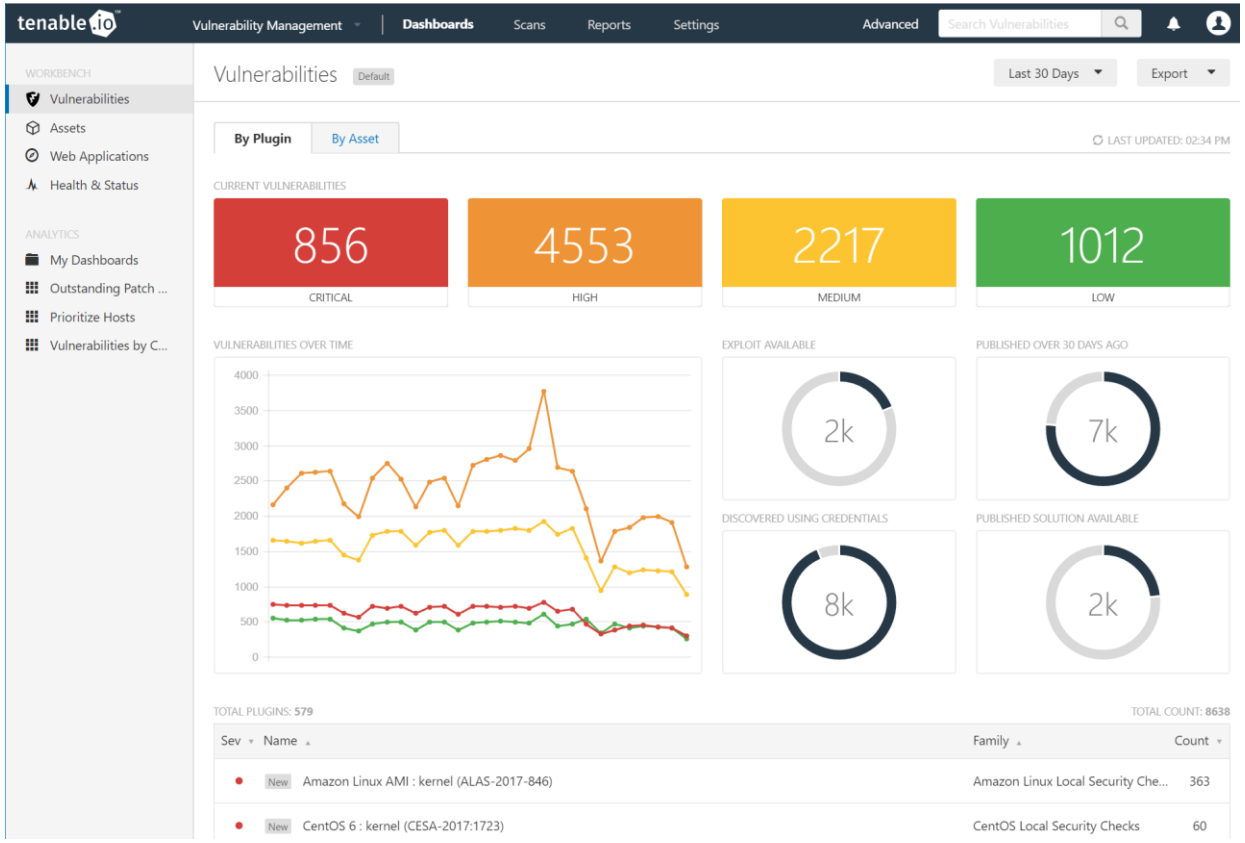
Amazon Machine Instance (AMI)

An image in the cloud

- Who created the AMI?
- Do you trust the creator of the AMI?
- What software is installed on the AMI?
- How is the AMI configured?
- How recently was the AMI patched?
  - Were the patches validated as resolving the vulnerabilities
- Are there any backdoors or access keys installed?
- Roll your own?

Vulnerability management on your instance(s)

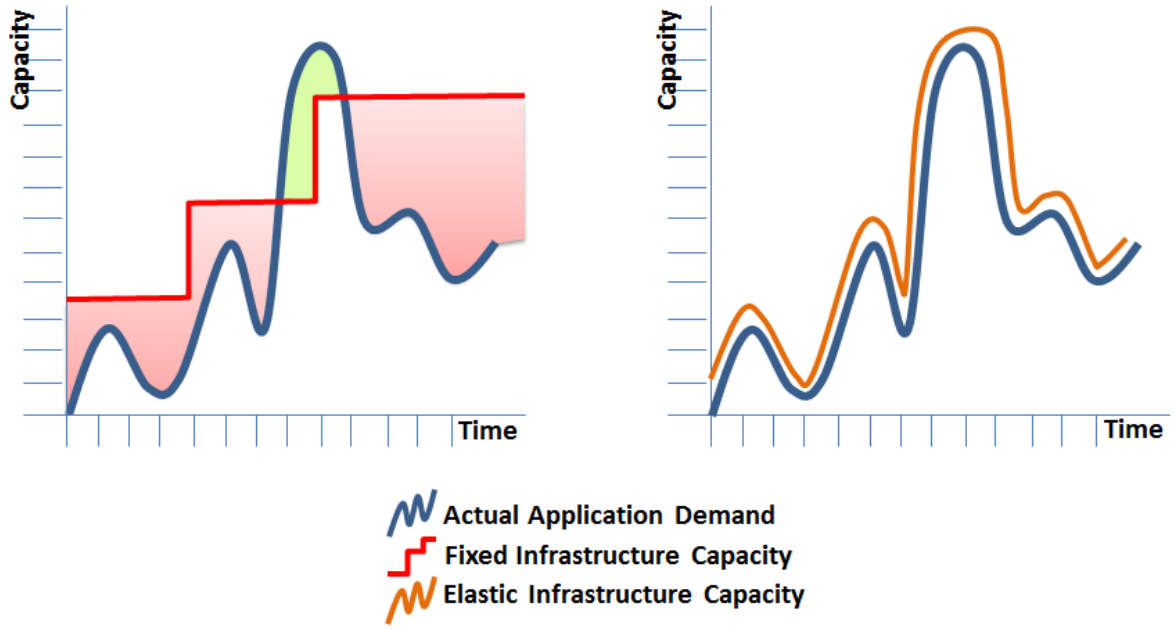
- How do you know what software is installed and how is it configured?
- How do you know what vulnerabilities are present on your systems?
- How do you patch your systems?
- How do you remediate insecure configurations?
- Use a SCAP compliant scanner:
  - https://scap.nist.gov/validation/



Patching

- How often do you patch?
- Do you patch?
- If you don’t patch since you rebuild your systems from scratch, how often do you update your AMI?
- Do you know if your AMI is up to date?
  - What about 3rd Party AMIs? Community, Vendor, Internal
- Is the patched AMI free of vulnerabilities and/or configuration issues?
- Stability of an old AMI that has been patched repeatedly vs. a new AMI (built from scratch)
- Patching with AWS EC2 Systems Manager (if you have Windows systems) https://aws.amazon.com/blogs/mt/windows-ami-patching-and-maintenance-with-amazon-ec2-systems-manager-2/

Auto Scaling



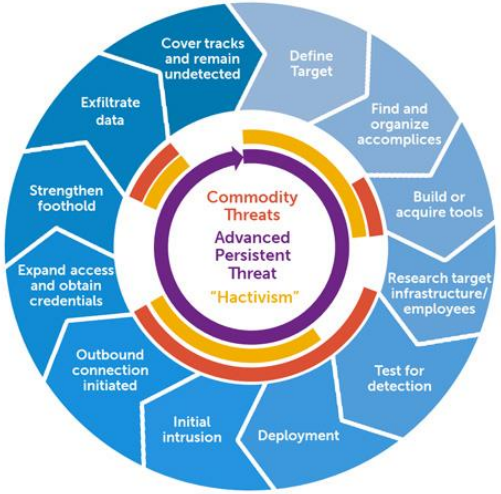
- Where did the AMI come from?
- When was it updated?
- How long will the instance(s) run?
  - Do you have a policy regarding length of instance lifetime in an auto scaling configuration?
- Maximum age of AMI before required replacement
  - Reduces possibility of APT
- Many of these principles apply to Elastic Beanstalk as well

Building your instances

- Start with an absolute minimal OS, barely enough packages to run networking and login
- Use vendor or 3rd party hardening documents / checklists – freely available
- Use Config Management (puppet, chef, SCCM, etc.)
- Layer your software onto the instance, only install what you absolutely need
  - If it is not installed, you don't have to protect it or patch it
  - You are reducing your attack surface and potential vulnerabilities
- Don't run services as root unless privileged port is required and use a proxy service if possible
- Run agents that report on the software vulnerability status (Qualys, Nessus, etc.)
- Patch on a regular basis
- Stop or terminate instances not needed, if it is not running, it cannot be hacked

Configuration Management

- The goal with configuration management is to have a consistent and repeatable environment. While being unique as individuals is a good thing, it becomes unsustainable as far as supporting an infrastructure
- Cloud formation – define your environment such as VPC's, SG's, Instances – makes for a quick and repeatable deployment
- AWS CLI – automate tasks such security group changes, can be scripted and run automatically
- Lamda functions
  - Alert on non-standard implementation, i.e. spinning up an instance with everything "wide open"
  - Automate tasks such as security groups: https://github.com/marekq/aws-lambda-firewall



Instance – Life Cycle Management

- The longer an instance runs, the more likely it is to be compromised
- In order to protect against advanced persistent threats (APTs), you should recycle your instances on a periodic basis (weekly, monthly, quarterly) – the shorter the less likely an APT can exist
- If you are using Auto Scaling or Elastic Bean Stalk, make sure your AMI's are current – Also validate those base images on a periodic basis, just because something is not vulnerable today, it doesn't mean it won't be vulnerable tomorrow

Accessing your instances

- Configure your security groups to allow access to the instance (SSH, RDP, etc) only via trusted IP addresses/network ranges
- Use strong passwords (sufficient length >16 chars) if you CANNOT use keys
- Integrate with your internal directory services (AD, LDAP) using federation if possible (ADFS, SAML, etc)
- If using keys, make sure you delete the keys for personnel that have left your group/company and don’t need to have access to the instance, define this responsibility
- If using a Bastion host, ensure it is HIGHLY secure

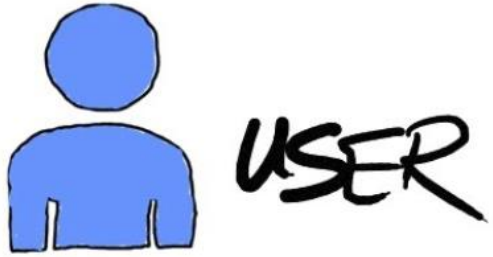
Account Management

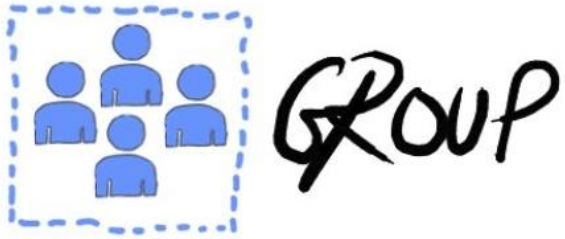
Do you know who is doing what, why and if they should be?

Users, Groups & Roles

- Users, Groups and Roles
  - Users are assigned to Groups, then you assign permissions to the groups
  - Roles are similar to users, but roles can be assumed by users to perform other tasks (if the role has appropriate permissions) that are assigned to those roles
- When to use roles
  - When using federated logins and you don't users to have to provide a second set of credentials
  - When creating applications that access AWS resources
  - For simplified management across multiple AWS accounts

  - https://docs.aws.amazon.com/IAM/latest/UserGuide/id.html



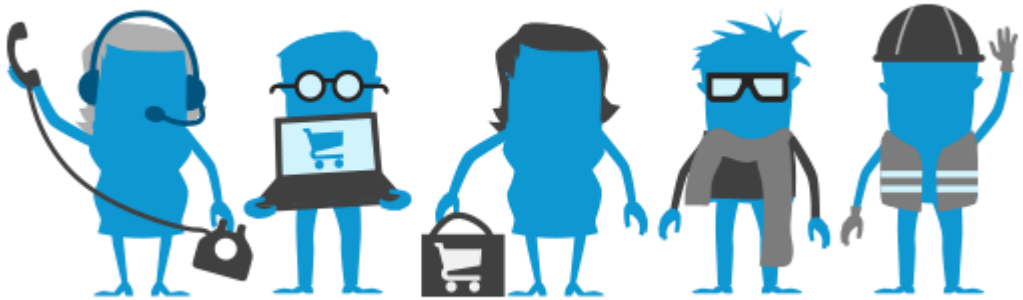


Securing IAM accounts

- Create and apply a default policy that enforces key items such as:
  - Password length and complexity
  - Password duration
  - Other password parameter such as reuse, how frequently they can change, etc.
- Require MFA on all logins via the console
  - Use free or commercial software tokens such as Google Authenticator, RSA, etc.
  - Use hardware based password tokens such as Gemalto token

- https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_passwords_account-policy.html
- https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa.html

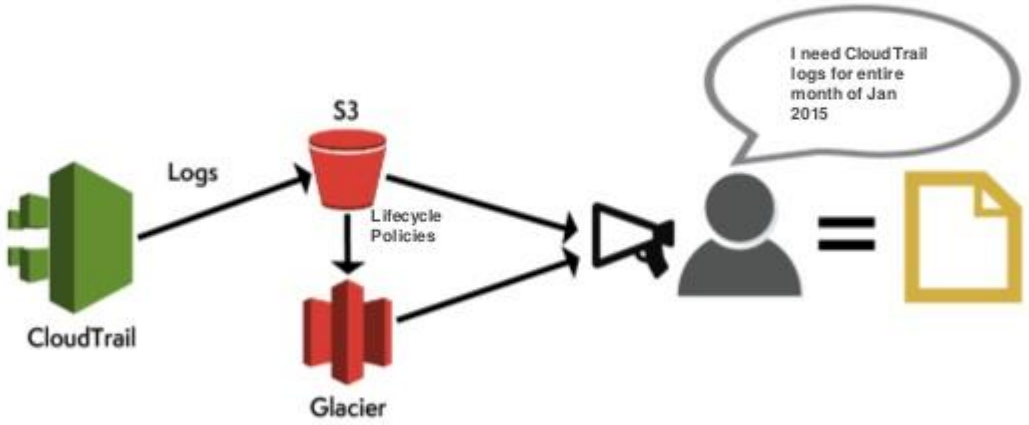
Account Management



- Remove users in a timely fashion if they no longer need access to the account
  - This includes terminated employees or employees that have changed jobs/roles and no longer need to have access to the account
  - If you are using federated logins, this is much easier
- Only add users that have a need to access the console
  - Make sure the access has been approved by management
  - Assign appropriate permissions, don’t give them more rights than they need to accomplish their job
  - If they have rights to create and terminate resources, make sure they have the proper training in processes and procedures and that they follow security rules for configuration

Logging

Needs to be enabled before something happens to be useful



CloudTrail

- Every account gets 1 free CloudTrail
- Use it to log all API access, CLI, GUI, etc. – anything account related
- Send the CloudTrail to an S3 bucket for storage and further analysis
- For additional security, send the trail to an external SIEM (Security information and event management), such as Splunk ES, ArcSight Logger, OSSIM (AlienVault) – The SIEM can alert based on certain events, otherwise it will be investigate after the fact
- Are you actually reviewing these logs on a regular basis??

Centralized logging

- Good practice to send all logs to a centralized log server for correlation (ELK, Splunk, ArcSight Logger, Kiwi, Graylog, syslog-ng)
  - This log server should be in a different VPC, account (preferred), etc. to reduce the likelihood of data removal during a breach
- Instances will have to have syslog (Security logs [minimum] on Windows) forwarding configured or a log collection agent installed on them in order to forward logs to a centralized log server
- Ideally the security (wtmp, utmp, btmp, Windows security access logs, web server access logs, etc.) related logs should be forwarded to your SIEM for detection and alerting purposes
- When in doubt, log it …..(make sure you are not logging sensitive data)
- When you are later investigating a breach or looking for root cause on an issue, you have to have the data. Logging after the fact is too late!!!

Tools

AWS and 3rd party tools to make your life easier

Some of tools we use

- SCAP Scanners (Nessus & Qualys)
- Lambda functions
- Cloud Formation
- Scripts (JSON, AWS CLI, Python, Etc)
- Patching tools (Ivanti)
- Excel
- Security Monkey
- AWS Trusted Advisor
- Custom home grown tools, ie. Zeus

Tool challenges

- There are so many tools, each one handles a specific task, it is impossible to keep track of what tools are out there
- Find tools that work for you and make your processes fit the tools
- Often teams will try to make a tool work with a process – customizing tools can be expensive and prevent you from upgrading
- Don’t chase the latest tool, but rather learn how to use the tools you have more effectively. Are you using all the features?
- This should be an ongoing learning process -- improve and evolve
- Collect user feedback and work with vendors for features and bug fixes

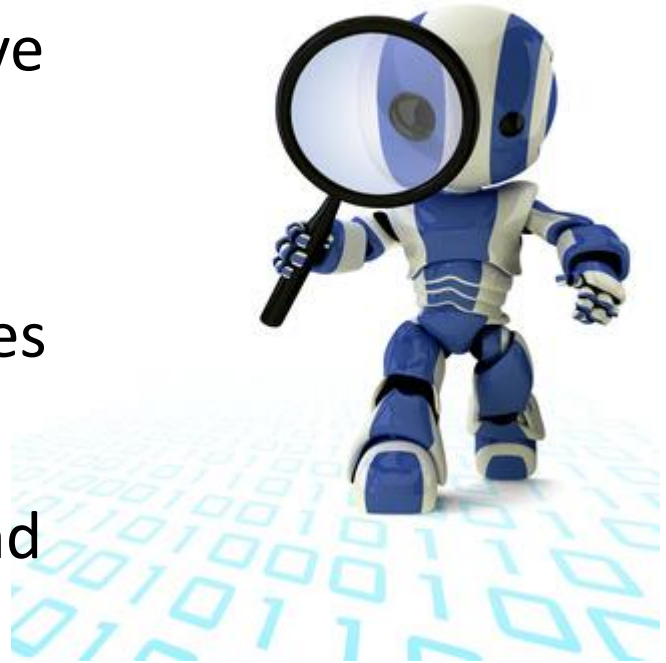
Software

Deploying, maintaining and knowing the security state of your custom software

Static Code Analysis (if you are creating code)

Static Application Security Testing (SAST)

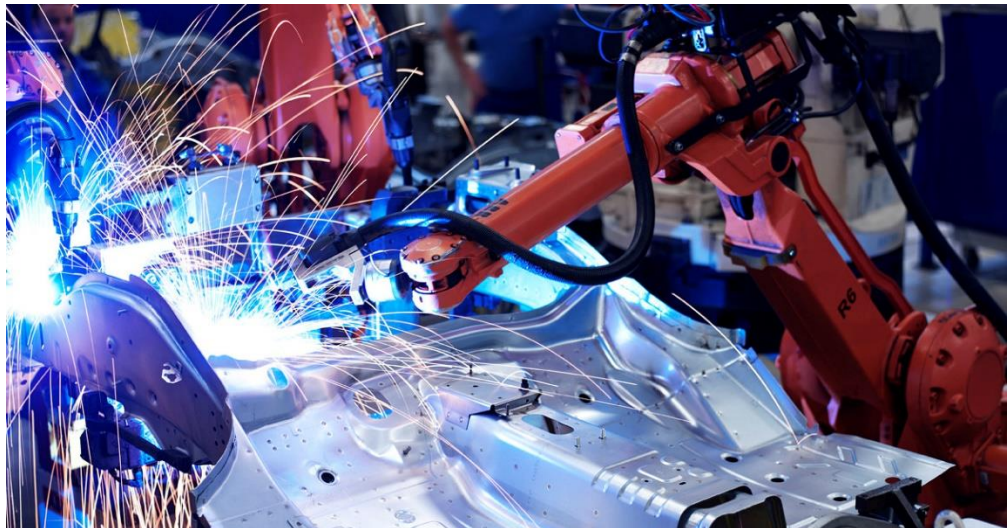
- Scan code upon checking into repo
- If possible, integrate this into your CI/CD pipeline if you have one
- Cheaper to fix issues before they are rolled out to production
- Will need to review the findings and mark any false positives to ensure that your dev's can focus on real issues and not noise
- Work with the development teams to help them understand how to code securely
- If you are using third party including Open Source components in your code, how do you know those don't have security issues?
  - Interesting area, quite a few new vendors venturing into this space



Dynamic testing of code

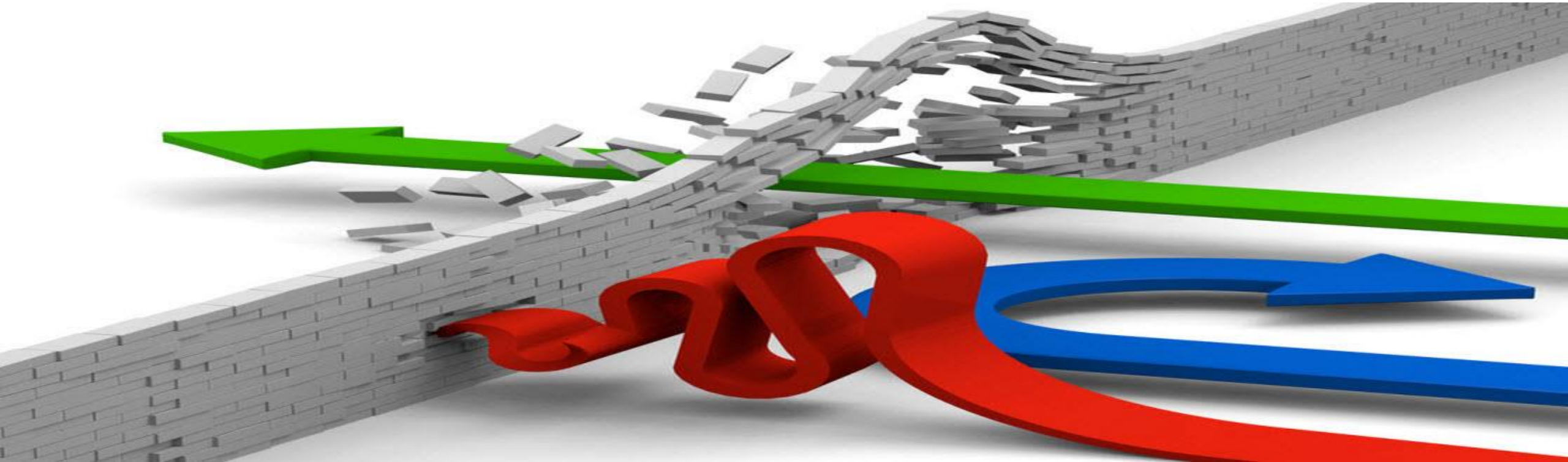
Dynamic Application Security Testing (DAST)

- Test your code as it is running on infrastructure after it has been compiled (if required)
- You will discover issues that you cannot find if just running static code analysis – such as system calls, DB interaction, load issues
- Use tools to continually test your applications, but make sure you are testing non production stacks, otherwise you might cause a DoS on your own site
- Set it up once, let it run continuously
- Review the results of the test
  - Scanning without remediation is a waste of time, it might even prove negligence in a legal proceeding



Vulnerability Scanning / Pen Testing

Automatic testing, manual testing and getting permission



Live Application / System testing (automated)

- Use automated tools to continually test your systems and applications
  - Once scripts are defined, the testing is essentially free
  - This will simulate users and/or attackers utilizing your site
  - Your testing tool will actually login to the application in an attempt to find flaws
  - You might even define tests that utilize users with different privilege levels, this might be especially important if you have additional screens for higher privileged users – remember, many of the attacks are from “insiders”
- Where to test
  - Non-prod environment vs. prod – most testing should be done from non production so you don’t impact site performance, but you might want to periodically test the production site as there are often much larger data sets involved. At the very least, before going live
- Automated tools can find a lot of issues, mostly low hanging fruit, but are unable to find deep logic bugs….

Pen Testing

- Conduct regular pen tests, either by “independent” internal teams or external companies that specialize in this type of work
- Have a clearly defined scope – watch out for scope creep
- If you want to inform your Blue Team that this is coming is entirely up to you – pros/cons
- Get written permission from internal management (executive staff) and Amazon
- Try to perform Pen Tests on non-production environment if possible
- Include social engineering in your pen tests, your security is only as good as the users
- If you discover a show stopper or break something, stop the test and report it immediately to get it rectified
- Provide a comprehensive report
- Share the findings with your blue team (red informs blue)

Data Security

Give access to the data to those who should have it and denying it from those who shouldn’t

Who should have access to data

- Who needs access to the data?
  - Why do they need access?
  - What is the risk if they have access?
- Can we restrict developer access to the data and still function or troubleshoot?
  - Perhaps have a DevOps person with the access and they can screen share with the Dev when needed
- Compliance with regulations, think GDPR, PCI, PHI, PII, SOX, etc…….
  - GDPR will be a game changer for those of you who are or will be doing business in the EU. It goes into effect on May 25, 2018…..not a lot of time but certainly a lot of changes
- Separation of duties
  - Don’t give a single person all the rights, try to separate those rights across multiple people – obviously this won’t work in very small teams

Further reading

- Look into cyber security frameworks/controls
- NIST 800-53
  - http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf
- ISO 27002 (while not a subset of the NIST 800-53, it has a much smaller focus)
  - https://www.iso.org/standard/54533.html
- Critical Security Control (my recommendation) – good real world controls
  - 20 controls that should be implemented (it will take several years) and most of them can be automated.
  - https://www.cisecurity.org/controls/

Questions?

Presentation can be downloaded from here:

https://github.com/derekhillhp/AWS-Security-Class