University of Missouri – Columbia

Capture the Flag Training Box

Team CyberSec

Team Members:
Team Leader: Andy Schuster (absdm2@mail.missouri.edu)
Steven Coll (src4gb@mail.missouri.edu)
Nadia Guidry (nig39k@mail.missouri.edu)
Michael Schmitt (mjs7wf@mail.missouri.edu)

Team Mentor:
Ronny Bazan Antequera

10/22/2020

Abstract:

The Capture the Flag (CTF) Training Box is a cybersecurity learning environment made with Docker containers deployed on Amazon Web Services (AWS). Docker containers are units of software that contain light-weight, virtual images of operating systems. We plan to create the CTF learning environment with a perfect difficulty increase as users pass through the training levels. Our key methods will be assigning each of our team members to a specific area of work, such as scripts for the dockerfiles, vulnerable web servers, vulnerable databases, and vulnerable operating systems. Assigning tasks in this way will help overcome problems of people working on the same thing accidentally or someone working on something that they are not efficient at. Near the end of the final deadline, we will run the system to catch any possible errors that may have gotten missed throughout the process and test them on multiple devices and virtual machines. We will be utilizing red team (offensive security) exercises for this virtual environment. These types of skills are crucial for testing your own network to find any vulnerabilities and to patch them before someone else finds them.

Problem Definition:

Our team's goals for the CTF Training Box are to provide a learning environment for users in order to teach them Red Team based concepts and educate them on cyber vulnerabilities. Our learning environment will be hosted on a website with Red Team based challenges for users to execute. The motivation behind our project is to help individuals gain skills and knowledge about cybersecurity concepts through our challenges. By doing this, users can use the skills and knowledge they learned from our website for a real cyber security career. While our team wants to teach users about cyber security concepts, we also want to teach them how to protect themselves. By utilizing our website, users will learn what not to do in a situation where hackers are trying to exploit them. Instead, they will know how to combat potential attacks to keep their systems more safe. This is important because with a lack of cybersecurity professionals, unethical hackers can do a lot of damage to our society (see table 1.1 and 1.2). Users will use our website to learn about offensive security skills. This will allow them to learn how to keep their systems and personal information more secure against hackers who target victims everyday.

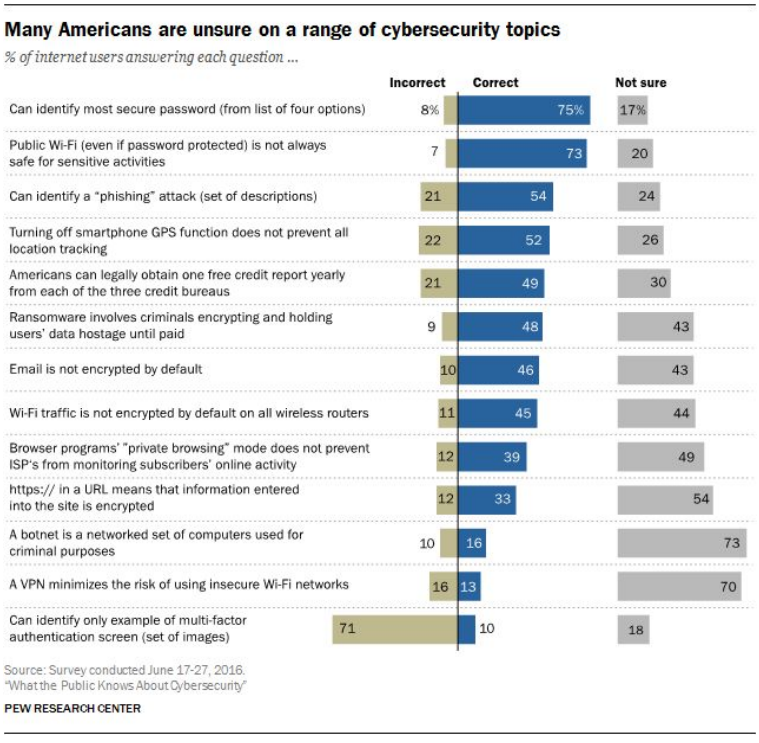
There are similar boxes that have been made, but the ones we have come across are either not through docker containers, are not made with progressive learning in mind, or are not through the cloud based on what we have come across on the internet. Combining these two things will be a fresh take on having people learn and have fun all while using minimal computer resources. One main problem some people have is a lack of specs on their computer to run fully fleshed virtual machines while also being able to do their other personal things because of the amount of memory and CPU used by virtual machines. With Docker and AWS technology, we plan to make this problem of lack of hardware extinct. Capture the Flag cybersecurity events are similar to what we are doing, except they assume you to already have the knowledge to compete. This is another problem - it can be intimidating to people and discourage their efforts in learning cyber security. This is why we plan to give similar challenges as Capture the Flag events but without it being a stressful competition so we can welcome all skill levels.

Graphs, Tables, and Images:

Table 1.1

Table 1.1 shows how costly human mistakes are. You can notice that these numbers are recent, from 2019, and is still an issue in today's work environment. With multiple companies moving into the teleworking format, we are certain that we will see these 2019 stats eventually rise. COVID19 has made many companies realize that employee presence at a physical location is not necessary to accomplish the same job. This presents more opportunities for hackers to get into systems, it also presents an opportunity for employers to educate their employees. We hope our project can be a tool for them to assist in this education.

Table 1.2



The survey on Table 1.2 shows that many cyber security concepts are not understood well by the general population. Based on this data and the importance of knowing how to stay secure, we aim to tackle these issues and make a friendly and easy to use platform for users to learn. Once broken down these concepts can be easily

understood, but many people have never had it broken down and/or would not know where to go to have this done for them.

Table 1.3

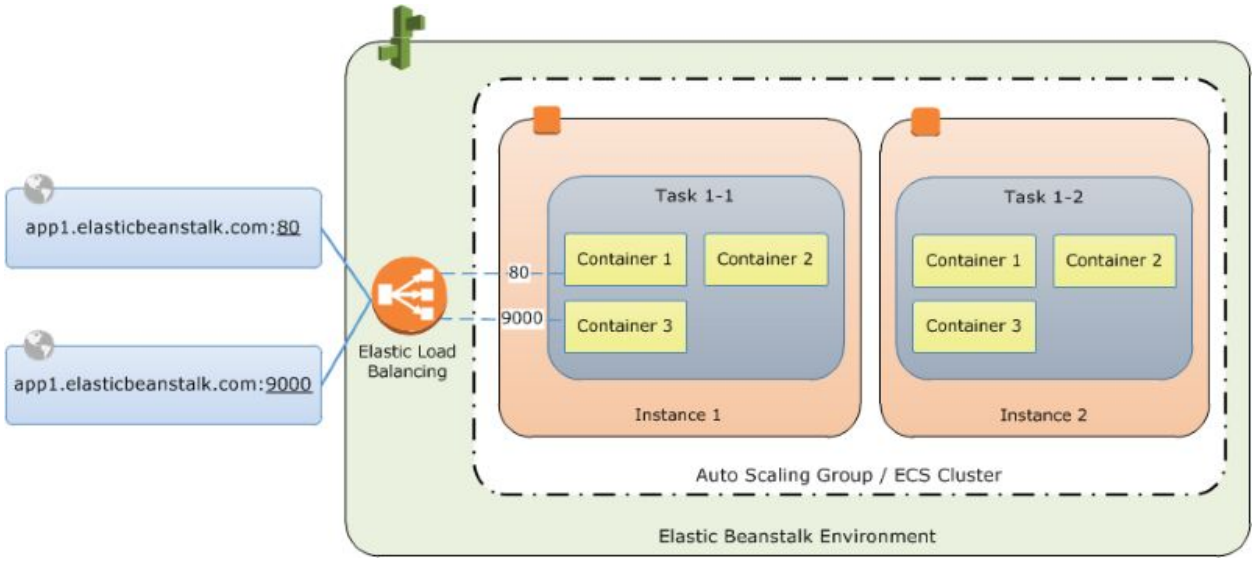


Table 1.3 shows the basic structure that we will follow when building our challenges. In a standard docker platform you would only be able to have one docker image running on one instance at a time. We will use the elastic beanstalk environment to run multiple containers side by side. This will help us maximize the docker capabilities while allowing multiple challenges on one level of difficulty.

Table 1.4

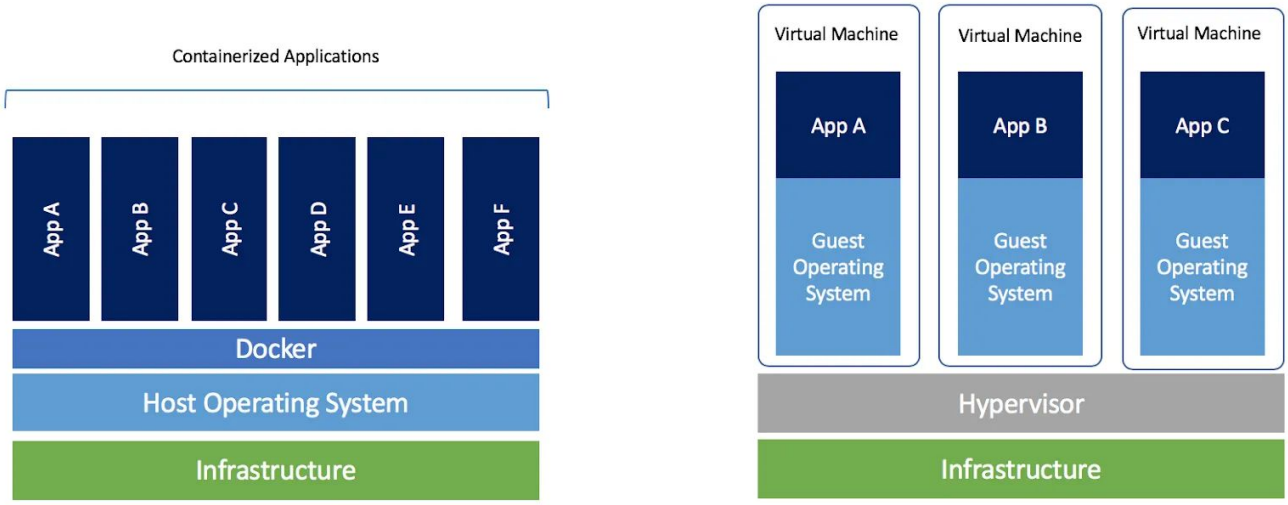


Table 1.4 shows the difference between docker containers(as seen on the left) and virtual machines on a hosted hypervisor(as seen on the right). Docker containers make it possible to create portable, lightweight environments. Docker containers are essential for this project as the purpose is to have users be able to load up the applications quickly. This is because it maximizes the amount of applications running on a minimal amount of servers.

Frontend Design:

Our front end design will consist of HTML, CSS, and Javascript. HTML will be used to create and structure our website. This includes headers, paragraphs, links, etc. For our CSS, we will be using Bootstrap 4. Bootstrap's grid system is built with flexbox and allows up to 12 columns across the page. This is ideal because it will format to mobile devices. It is also a well known library to use to make our project more efficient. Along with Bootstrap, we will use CSS. CSS will be used for the styling of the project which includes fonts, colors, layout, etc. Lastly, Javascript will be used for the homepage animations and effects to captivate the user and engage them to enter our website.

Backend Design:

users	
id	int
email	varchar(100)
username	varchar(100)
password	varchar(100)
score	varchar(100)

This is an example of the SQL table holding the registered users information.

Our backend design will consist of docker images holding different challenges. Along with the challenges, we will also have a docker image holding a database (see table 1.3). This database will be used to store the user's account information (username, password, and email) upon sign up. Not only will the database store the user's account information, it will also store their points for the challenges. This is valuable because the user will not lose their points each time they login. The SQL users table uses the columns id, username, password, and score. Our team will utilize PHP and an SQL docker image to store the user's information and display the page accordingly depending on if they are logged in or not.

Proposed Solutions:

We will create our CTF Training box using website development, database management, dockerfile scripting, AWS web servers and operating system docker images (see table 1.4). We

will also modify the docker images to be installed with our chosen tools for solving the challenges.

For hosting our project we have chosen Amazon Web Services. As a group we have all used AWS in previous courses, so choosing AWS as hosting our server was almost a no brainer. Our mentor was kind enough to give us credit with AWS so we will have enough credit to house our instances for the duration of this class. AWS instances are simple to set up and are straightforward to use. We plan on setting up our instances on one host account. The team members will then each create our own key pair and share the public key in the AWS instance. For an extra layer of security, we may also decide to change the ssh port to something more secure, rather than having everyone know our SSH port is 22.

Not only do we plan on hosting our instances containing our dockerfiles on AWS, we also plan on hosting our web application on AWS as well. A web based application will be a lot easier to navigate and more user friendly. We plan on incorporating our challenges into three categories; Beginner, Proficient, and Advanced. These three categories will hold different difficulty challenges. With this format, the user can practice each level rather than being stuck doing challenges below their level.

In cyber security, the red team represents offensive security professionals who are experts in attacking systems and breaking into defenses. The blue team is used for defensive security responsible for maintaining internal network defenses against all cyber attacks and threats. With our attacks being red team based, we will instruct our users how to crack a MD5 hash password, find hidden text/flag in imagefile, cross Site Scripting, gain root access in another linux distro, find text in hidden directory, SQL injection, Brute Force another linux distro password, Cryptography, Steganography and so much more.

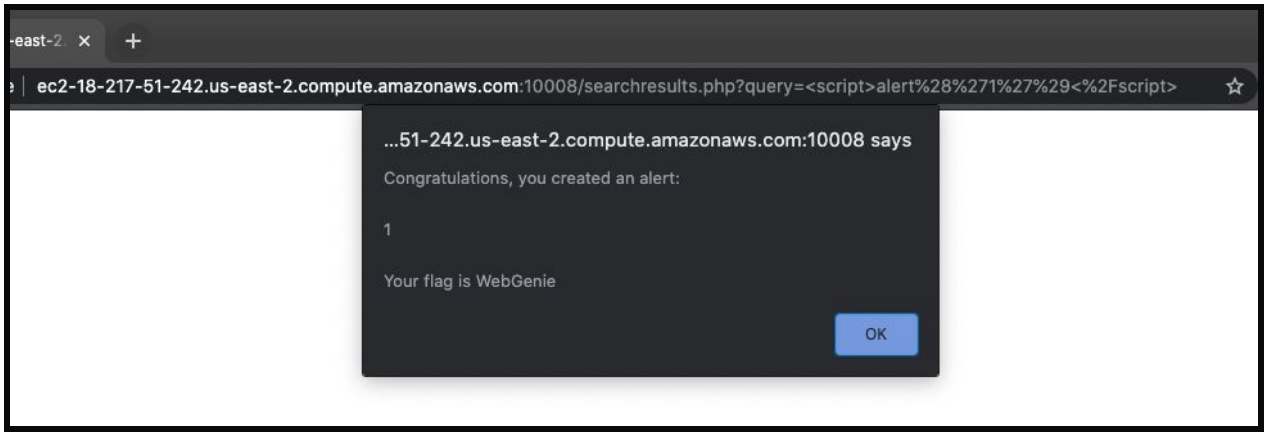


Image above: The image above is an example of what the challenge flag will look like. After the user successfully completes the challenge, they will be greeted with a pop up that tells them so. They will also be presented with a flag, in this case, the flag is "WebGenie". With that flag, the user enters the flag to proceed and to gain points on the leaderboard.

Once our instances are created, we will use those to host docker images and containers. This will be a pretty simple task, as AWS has a lot of documentation surrounding dockers and

instances. To accomplish this, we will map docker images and containers to different ports which will allow us to run multiple docker created websites.

Development Timeline/Schedule:

Week	Docker Scripting/Network setup	Website coding/Database configuration
Sept. 20, 2020	- Plan where containers will be downloaded from - Create test Dockerfiles	- Setup AWS server - Setup AWS database
Sept. 27, 2020	- Begin development of scripting vulnerabilities	- Begin coding vulnerable sites with easy difficulty
Oct. 4, 2020	- Setup network between Docker Containers	- Put vulnerable sites on the AWS server that are accessed through specific port numbers
Oct. 11, 2020	- Create instructions and hints on how to do the challenges	- Create instructions and hints on how to do the challenges
Oct. 18, 2020	- Begin developing steganographic files	- Start configuring website
Oct. 25, 2020	- User permissions	- User permissions
Nov. 1 2020	- Additional research	- Additional research
Nov. 8 2020	- Scoring system	- Scoring system
Nov. 15 2020	- Scoring system	- Scoring system
Nov. 22 2020	- Add additional challenges if need be	- Create main web page to access website challenges
Nov. 29 2020	- Configure web page with challenges	- Have webpage tested
Dec. 6 2020	- Testing CTF box	- Testing CTF box
Dec. 13 2020	- Bug fixes	- Bug fixes

Work Delegation:

Although we have mapped out our work delegation to a table, this is ultimately a group project. This cannot and will not be done by one team member as we will all pitch in wherever is needed. If something needs to be done, we can certainly all help each other. The table below is strictly an outline on what team member is responsible for what part of the project. It does not indicate a team member being "stuck" in the position that they picked.

Team Member	Responsibility
Andy	Team Leader, Organizes group meetings and turns in all group assignments, Website design, working with color palettes to fit project aesthetic, Navbar configuration and compatibility, CSS bootstrap for mobile users, dockerfile scripting, Testing, Debugging,
Steven	Challenge creation, internal website dockerfile scripting, Database creation for users scoreboard, login database that will allow users to register for an account and track score, acquire website domain and https certificate, Testing, Debugging, Web Page Design
Michael	Creation of internal docker files for challenges, Internal dockerfile scripting for challenges, Creation of web page and email addresses for web challenges, Creation of video walkthrough/tips to help users complete challenges, Testing, Troubleshooting
Nadia	Overall website design, layout, and creation, creation of the hacker figure that will guide users through the website and provide tips throughout challenges, Internal dockerfile scripting (see above timeline), Testing, Debugging

Possible Troubles:

A problem we may face as a group is communication. Due to COVID-19, the course has been moved online. Because of this, there will be no face to face interactions. As a solution to this potential issue, we have dedicated one to two days a week to communicate through scheduled Zoom calls. Although it is not face to face, we are still making an effort to come together during these difficult times remotely. We have also formed a group message. This is a great way to communicate if specific group members are unable to attend the Zoom meetings. By having this second form of communication, we can give updates or share any information they may have missed during a meeting. The group message also allows group members to voice their concerns or share any information in relation to the project.

Another potential issue is setting up user permissions incorrectly on the box. It is possible for an individual connecting to the box to have more privileges than we intended. This could leave our box very vulnerable and dangerous to operate because it leaves an open door for

viruses and malware exposure. While permission issues relating to the box may be a problem, we also want to ensure that users have enough privileges. By having certain privileges, users will be able to perform tasks without running into unintended barriers. These barriers can happen due to being too low-level for a user within the permission rules. We could do this by creating a group in the sudoers config file and restrict permissions from there, making it so only the root has full access and users are limited.

Another potential issue is working simultaneously. Like mentioned above, this is not possible to do in person due to COVID-19. This was an issue that we had to review with our professor, group members, and mentor. We wanted a system that would allow each group member to add to the project at the same time. By using this method, we remove the issue of waiting for a group member to finish their work before we can start ours. After discussing it with our mentor, I believe we have found a possible solution. Our solution would be one account hosting the servers, then allowing access to ssh for each team member. This will work securely if we create separate key pairs for each member to be able to ssh into the instance.

Testing Plan:

Programming anything comes with a decent amount of bugs along the way. We know that we will not have a flawless system right off the bat. Therefore we have dedicated a lot of time to working the known bugs we see when setting up our training box.

Testing Type	Audience
Team Testing	This testing will include solely the team. With team testing, this will include working through bugs as we go.
Mentor Testing	This testing will include our mentor. Once we are a comfortable place and do not notice regular bugs, we will share with our mentor. With his knowledge and his testing, we will know if any errors were missed from team testing. If errors are found, we will fix them and retest the error to make sure it was a permanent fix.
Beta Testing	This type of testing will be by our peers as well as users that want to get involved. This testing will include bugs found by users and reported to us for fixes. We will then fix these bugs and have the user retry the given bug to see if it is fixed. If not, the cycle repeats until there is a fix. This testing will only be conducted by beta testers, meaning only the beta testers will be allowed access. Any other person will have to wait until the next phase of testing is active, meaning the product is live.
Active Testing	This type of testing includes errors or bugs found when the program is operational and in full production. Similar to the beta testing above, this testing will cycle

	until the error is fixed. This testing will remain until the product is no longer active.

Maintenance Plan:

Our current maintenance plan is to continue to add to the project and update it frequently. Our team will use the credit given for this course in order to keep the website live and accessible. We want users to continue to learn through our website and allow team members to use it for future career opportunities. By consistently updating the website, users will have access to new features and challenges after updates are made. Having the project accessible will allow team members to be able to present the project during an interview for a job or internship. In addition to this, all team members will have access to the original code. This allows team members to make changes that they think are beneficial for the project to be more efficient.

In the event that the website is passed onto future students, all of the project code is located in the instance at /var/www/html/. In this directory, you will find all of the contents of the website that include our project's HTML, CSS, JavaScript, and PHP code. To make things simple, we used FileZilla to SSH into the instance where we placed all of the website's content. This makes it easy for future students to be able to take over the project if needed.

Resources:

Null Byte Youtube Channel: An informative Youtube channel that shares tutorials and videos for ethical hackers, computer scientists/engineers, and individuals interested in cyber security. This channel provides a wide spectrum of topics to choose from for inspiration regarding our dockers, challenges, and project design as a whole.

Steven Coll (team member) Github: Steven's Github acts as a great way for our group members to see the practice dockers he created as an example for our project. This allows the opportunity to give feedback on his methods regarding his personal execution of a docker.

SniperOJ Dockerfiles: A Github account that runs SniperOJ (an open source CTF (Capture The Flag) Platform) with dockerfiles that hold multiple challenges. This Github is strictly used as an example for inspiration and ideas of how we want to use specific techniques in our capstone project.

OverTheWire: A website that provides over 30 different levels of challenges that include cyber security related topics. Each level includes hints/directions of how to execute each challenge which may be used as a template later on for our project.

Hack The Box: A website that provides great ideas of CTF challenges that may act as inspiration or a guide for our project.

Try Hack Me: A website that presents a structured pathway of learning in order to test one's skills in order to tackle real world cyber security challenges. These challenges include hacking for instance, which is important to our Red Team concept that will be applied to our challenge ideas.

Docker Builds: A website that provides information on how to build multi-staged dockers. This website acts as a great resource to educate ourselves on how to build dockers that are specifically multi-staged to possibly use in our project.

Docker Hub: A website that includes hundreds of thousands of dockers that are already created and ready to use. This website includes an SQL docker that can be applied to our project database to save us time in order to be more efficient.

Challenge Inspiration ideas taken from Null Byte:

Beginner Level Challenges:

- https://www.youtube.com/watch?v=M0eEwqUpKDc&t=115s (Hack a computer using SSH)
- https://www.youtube.com/watch?v=Y_zjwIJTkBA (How Hackers Can Take Sudo Passwords from Linux & MacOS Computers)
- https://www.youtube.com/watch?v=z8_qz938wFU (Google Search Like a Hacker)
- https://www.youtube.com/watch?v=K78YOmbuT48 (Scan for Vulnerabilities on Any Website Using Nikto)

Proficient Level Challenges:

- https://www.youtube.com/watch?v=8a1yTN2kFNw (Conduct a Penetration Test Like a Pro in 6 Phases)
- https://www.youtube.com/watch?v=u_gOnwWEXiA(Find Vulnerable Services & Hidden Info Using Google Dorks)
- https://www.youtube.com/watch?v=3U1pJ-eJrAU (Find Network Vulnerabilities with Nmap Scripts)

Advanced Level Challenges:

- https://www.youtube.com/watch?v=1-ykWq6BEsQ&t=55s (Create Brute-Force Wordlists from Leaked Password Databases)
- https://www.youtube.com/watch?v=yhC5Kh5Z_4o (Crack SSH Private Key Passwords with John the Ripper)
- https://www.youtube.com/watch?v=PlP6i3qkaJk (How Hackers Can Brute-Force Website Logins)
- https://www.youtube.com/watch?v=aIW-BssqS3s (How a Hacker Could Create a Trojan PDF for Macs Using AppleScript, Part 1)

The challenges above will be made with our base knowledge and additional research. There are tons of tools and packages we can use to create these challenges, for example - Steghide (http://steghide.sourceforge.net/documentation.php) for embedding secret text in an image file, John the Ripper (https://www.openwall.com/john/) for brute forcing passwords, renaming linux directories with certain characters to hide them, and much more.

Table References:

Table 1.1:
https://www.ekransystem.com/en/blog/how-prevent-human-error-top-5-employee-cyber-security-mistakes

Table 1.2:
https://www.pewresearch.org/internet/2017/03/22/what-the-public-knows-about-cybersecurity/pi_2017-03-22_cybersecurity-quiz_0-01/

Table 1.3:
https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/create_deploy_docker_ecs.html

Table 1.4:
https://www.docker.com/blog/containers-replacing-virtual-machines/