KCDC-Alpha Report

Sagindyk Urazayev

University of Kansas
KU Information Security Club

March 20, 2019

KCDC-Alpha Report

Sagindyk Urazayev

University of Kansas
KU Information Security Club

1. Before the competition

In comparison with CANSec 2018, KCDC had a preparation period of ˜10 days. We were given a VPN access to the test network so we can work on it and by work, I mean taking a look. The scenario said that we are a hired contractee to inspect the company's systems and networks, implying that we are not allowed to make changes. We were given account credentials with sudo privileges to audit the system and write a comprehensive report about the recommendations we can give to harden their machines and network. What was really interesting, is the fact that this whole ballet of services consisted of 2 networks, totaling over 28 active machines. I will try to describe my experience in an objective manner. Subjective or biased thoughts are still possible. Read with caution.

There were two big networks:

- Production Network
- Development Network

1.1. Production Network

Traditionally, production network is where the actual services are deployed and/or are accessible via internet. The list below included the following machines/services.

The format is: **Name of service/machine** *(Operating System)* - Description.

- **Production Firewall/Router** *(Vyos)* - router to the outside world. SSH is available. About this later.
- **Metasploit** *(Ubuntu 16.04)* - collection of different metasploits and tools to test vulnerabilities. Metasploit by Rapid7 and msfconsole were the tools they expected us to keep up and running. SSH is available.
- **VPN** *(Vyos 1.1.8)* - provides VPN access to the production network. SSH is available.
- **Mail** *(Server 2016)* - pretty standard Windows-based mail server that listen and serves IMAP, SMTP, and POP3 ports. Very difficult to configure. Like any other mail system. Only RDP is available. The system provided full system GUI with pretty windows and stuff.
- **Kali** *(Kali)* - Linux distribution for penetration testers. On this networks I believe it was used as a workstation for "white hat" employees to work on and test their code/software. SSH is available.
- **Website** *(Server core 2016)* - provided website on port 80 (http) on the domain "alpha.red" if I remember it correctly. RDP and console-only access is available. Windows console to do some website hosting magic. Powershell was the way to go.
- **GitLab** *(Ubuntu 16.04)* - hosted git repos of company's metasploits, grok, software, and other useful internal stuff. SSH is available.
- **File Share** *(Ubuntu 16.04)* - I did not work with this machine. Probably just an FTP server, thus the name. SSH is available.
- **ELK** *(Ubuntu 16.04)* - this is the machine that I worked most on. This machine is responsible for collecting access logs from all other machines on the network. Like SSH logins and that interesting info. It was fundamentally broken and this is the **only** machine on the whole preparation stage that we were allowed to modify and actually work with. I had a privilege of working with it. Basically, it has

a lot of dependencies, including: logstash, elasticsearch, kibana, grok, nginx, and ELK. All machines on the network were sending their logs to ELK's port 9600 that logstash was collecting. And then elasticsearch with logstash were in sync to store the logs and categorize them. Elasticsearch was using the grok language to store all of it and serve RESTful API requests on port 5601. Where later on, Kibana was deployed on port 80 and would serve a GUI/nice webpage representation of all the logs collected via elasticsearh's API. This was a whole wagon of dependencies and other pre-requisites to get this system working. Initially, nothing worked. After several hours of aggressive DuckDuckGoing, I was able to copy paste commands from organizers' .bash_history that they left and online guides to get it working. Apparently, this was good news. SSH is available.

- **DNS** *(Ubuntu 16.04)* - I worked with this machine directly. It just served DNS requests when they were coming. Nothing malicious found. Though it was super slow. Every request would take up to a second. SSH is available.
- **Empire** *(Ubuntu 16.04)* - I believe this was a development/general purpose workstation that was serving on port 80. With a program called "Empire". Never heard of it but it had some really nice ASCII pictures of storm-troopers from Star Wars. SSH is available.
- **SSH** *(Ubuntu 16.04)* - just an SSH machine to configure SSH access. SSH is available.
- **rdp** *(Server 2016)* - just an rdp access machine. RDP only access.
- **production-dc** *(Server core 2016)* - Control machine on the network. All other machines were directly connected to it. All user credentials changes to the control machine would be immediately reflected on the whole network (Very useful). Console only RDP access.
- **Firewall** *(Vyos?)* - Firewall between the networks. Just a set of iptable rules. SSH is available.

1.2. Development network

Internal network where employees would perform their development tasks. It was connected to the production network via the firewall in the middle. Most of the machines are just windows machines with dynamic IP addresses. Probably just assigned by DHCP and I will not include them in the list.

- **Development Firewall/Router** *(Vyos?)* - don't know.
- **dev-dc** *(Server core 2016)* - one of the control machines on the network. It was connect to all of the Windows machines and the **Empire.**
- **trial-dc** *(Server 2019)* - why do they need two control towers on the network? That was the question that we had. This one is connected to everything that **dev-dc** was not connected to. My theory is that they are just testing/migrating to new system, as it is a more recent Windows Server release.
- **Metasploit** *(Ubuntu 16.04 LTS)* - another Metasploit machine.
- **VPN** *(Vyos 1.1.8)* - provides VPN access to the development network. SSH is available. What is weird about this VPN is the fact that the only way to connect to this machine is through the production network and the Firewall. It is **not** connected to the development router.
- **Kali** *(Kali)* - beefy penetration testing workstation. This is the machine that was used to break into the production machines and fiddle with them during the attack that came after. SSH is available.
- **Empire** *(Ubuntu 16.04)* - just a workstation? SSH is available.
- **Windows *** *(Windows)* - countless number of windows machines. They were off during the preparation stage.

1.3. Audit Report

We did our full network analysis of each box. Final report boiled down to 25 pages. It only accounts to 5% of final score.

2. The Thursday Attack

Our access got revoked on Thursday noon. I believe that later same day we got an email about an incident happened on the network. They fired a Red Team style employee but did not revoke his access to the

system until the end of the day. He cooperated with some other employees and did something bad to the machines on both of the networks. In the report that we got from the company, it was reported that **the only** good thing happened was that we fixed their ELK system so they could do a trace-back of the attack. I was wondering if this was one of my fixes that I introduced.

From here on, the competition started to get really *messy.*

3. The on-site setup

During most of the competitions, we have VPN access or any other type of remote access to the competition network and we can work on it. Optionally, you can be *physically* present on the site. They decided not to make the rules standard and required **everyone** to come to Manhattan. Firstly, we did not about this as they did not explicitly notify us about mandatory physical presence and we were not as agile to make people come there. We would be very exhausted traveling to other city. K-State's home team has no trouble coming on-site and working there. We sent our complaint form and request for a remote access to the boxes. The organizers reported that they did not have an appropriate infrastructure to make a competition network VPN. This was very strange. They told us they will **try** to provide us remote access but they **will** give us a 10% penalty of the final score in any case. We thought that is a bit harsh and takes out a part of fun when you are being penalized so much. We took the penalty, so we can have some level of confidence about our machines when actually going to the competition.

Members of our team allocated their own time on Saturday to work on the machines as they promised to provide us with stable VPN access from 8am till 8pm. What happened after is going down in history. VPN just simply **did not** work. It was up for about an hour between 8am and 6pm. No adequate access was provided to us, so we did not have a chance to actually work on the machines. I was a bit sad and happy to learn that we were not the only ones. Edwards team "Jayhackers" did not have access either, but they drove all the way from Overland Park to the setup. Physically present teams could not work on the machines. Late evening, around after 9pm, everyone was granted VPN access when it actually started working. I thought they would remove the penalty because clearly promised remote access was not working and everyone got it. I was wrong.

They told us we can work on the network all night long. Very funny joke they made. No one is going to spend their whole night debugging a pile of machines when the competition is at 8am. Attack Phase starts at 9am.

4. The competition

This was glorious.

So much happened that I will divide the whole section into small chapters. Everything that happened here, should be remembered by next generations and new teams.

4.1. The grace period

We had around an hour before the attack phase. Some of our machines died. We thought that someone ours accidentally killed it. Kid you not, it was the White Team. They did not setup some of the boxes correctly and the boxes just died. They tried to reset it, live-booting was on, use vcenter to have somewhat close to a "physical" access to the boxes. One of our team members, Yousif, spent around 2 hours working with the White Team in the Data Center, trying to bring our machines back live. And he was succeeding. Some of them were back. One by one. Not a lot was done there. We tried to squeeze in our last-minute changes before Red Team attacks.

4.2. Rocketchat

For the competition, they used slack-kind of a competition messenger. It wasn't bad and all but was very slow, we had to manually add a DNS server to our /etc/resolv.conf, and use Firefox, because it did not like any other browsers.

4.3. Scoreboard

This is where the trouble and confusion started. We all needed to manually register in Rocketchat and then we would have our accounts migrated to Scoreboard but with a default password of "password". It only worked for Noah? He sent us his credentials, because our attempts to login with our username and "password" did not work. We notified the organizers immediately.

Scoreboard was used to track teams' scores, services, anomalies, and the usual ranks. However, I think that they were working on it while we were competing. The scoreboard would die every couple of minutes. All the scores would reset to 500pts just because they were debugging it. The scores would come back. Eventually.

When we just started, for some reason, our team "Louisiana Hot Snakes" were the last ones with 2800pts. K-State's team had around 3200pts, and Edwards held the 3400pts mark. I did not know why. Noah Brabec told me it is probably due to the 10% penalty for the VPN. Seems a bit big of a penalty. Maybe we can still beat them?

4.4. Anomalies and New Penalty

Submitting anomalies was whole another story. Anomalies simply **did not** work when we had grace period. There were 4 anomalies to solve before the Attack but the buttons were inactive and we just could not submit any of them. I reported this immediately, asking whether we need to do them or what is going to happen. I am not sure I got a follow up on that. And what is more interesting, I got a direct message from one of the organizers, reaching out to me. Telling me that our team did not register rocketchat accounts yesterday(?) and they will make a 15% late submission penalty on all of our anomalies. What.

When it was time to submit anomaly. There was a problem to write a PowerShell script to add a user to Windows machines. What a horrible language. Very powerful bit incredible complicated, bloated, and restricted. There is no shell romance with pipes. To add password to a user, you cannot just pass a plain-text password like in UNIX `passwd`. You have to create a

```
$password = ConvertTo-SecureString 'password' -AsPlainText -Force
```

object and then pass it to the user creation to assign a password

```
New-LocalUser -Name $username -Password $password.
```

4.5. Internal Errors

So, I made this script and tried to send it and oh my. `500 Internal Server Error.`

Something bad happened on their side. Reported immediately and had one of the lovely organizers to help me with it. I saw how the whole scoreboard was made. It was just a python script with some Flask, gunicorn, sqlalchemy working together to make all of it. The error I was getting was divided into 2 parts. First, the script only accepted .txt extensions. Second, they relations in their database were not configured properly, so the database actively rejected my request and would just die violently right in front of me.

She did some SQL magic, database machinations, appending extensions to a python list on the server to make it work. The whole scoreboard was just one file called `server.py` script. It would basically just run and do everything. She would kill the script and restart it to make changes. Not saying it was bad or anything. Quite interesting, actually. My file was in .ps extension for PowerShell, though I have no idea if it were the correct extension. I just used it because I assumed. Asked them how long did it take them to build the script. I was told about 2-3 months. That seems true.

4.6. Add new domain

One of the anomalies was to add a new `Office` domain to the production network of Windows machines. Noah and Tiger were working on it. Apparently, something finnicky was about the current network that did not support scaling up? Our White Team member, Noah, and Tiger were working hard to fix the network and add a new domain. I believe it was never fixed because the domain was never there. And some anomalies depending on the new domain were not completed for this reason.

4.7. Just hash it yourself

This is a fantastic story of hord Linux boys and constant penalties. We lost access to the Production Router as the root password was gone and we couldn't restore it. We asked the White Team if they can help us with it via directly accessing the vcenter. The guy at the Data Center told us that he accidentally left his session on our router open and he can configure it back but he demanded that we get a penalty for it. It was a ransom - "You give me points and I give you the password". We do not make deals with terrorists, so we politely refused the generous offer and moved on. If we didn't have access, no one had it. And the router was in perfectly fine condition. Don't touch it if it works.

Before that, Ross was trying to change password to the `vyos` user with `passwd vyos` as root but it just did not work. Root got rejected by the system? I just told him - "Just make the hash of the password and put it in `/etc/shadow` file." I meant it as a half joke but the madlad did it. Ross started working on manually configuring the UNIX password in the shadow file. Noah complained that maybe Linux adds salt to it. Eventually, Ross could not complete the mission but what a time to remember.

4.8. DOS the VPN

The organizers did not provide their own WiFi, so we had to connect to KSUGUEST and use our VPN profiles to get into the networks. Every time the VPN was working again, I would joke with a hacker phrase - "I'm in." It was so unstable so that one time it just turned off. And this phrase was the catchphrase of the day - "I'm out." Read it with the same hacker voice whenever the system or the VPN kicks you out of the network. Complete polar opposite of "I'm in." Also, do not forget about "Omae mou shinderu", which roughly translates from Japanese to "You are already dead." Use it when services are dying. In response, just yell "NANI?", meaning "WHAT?".

White Team reported that Red Team is overloading the VPN servers with DOS attack. This is something new. I would never suspect that Red Team would do such a thing. I am still not sure if it were the Red Team DOSing our VPN or White Team breaking it on accident. I like to believe in both stories. Noah went to the Red Team with some Mountain Dews as a bribe to stop pwning us. However, I would not confirm nor deny this.

4.9. Dvorak incident

This is a very good one made by me by accident. As you may know, by default I use dvorak keyboard layout. Took me about 3 months to migrate from QWERTY (worth it). And when I used remmina to RDP into the Windows console machines (**prod-dc, http**), somehow, RDP read my keyboard layout of my system, sent them to the windows server and configured it there. Permanently. Poor Yousif trying to log in to the **prod-dc** was caught off how every key he typed was different. He thought we were compromised (very dangerous on control towers) and when I heard, I immediately had this idea but refused to believe in it. Started typing on Yousif's keyboard, it was dvorak. Welp, there is no way to change layout in console-only Windows and I was solely responsible for it. It was fine, we updated passwords and stuff. Security through obscurity, what can I say. I'm in awe.

4.10. The Services

One of the big things as usual were the services. We had to keep them up and running. I believe we had around 10 of them.

1 **prod VPN** - production VPN (died when "DOS" happened)

2 **Empire** - production Empire

3 **http** - The website is up and running (hacked by Red Team)

4 **Metasploit** - Running Metasploit server by Rapid7

5 **IMAP** - Mail

6 **SMTP** - Mail

7 **POP3** - Mail

8 **SSH** - SSH

9 **rdp** - rdp

10 **File Share** - file share

I have to say that we were the **only** team that was keeping the most number of services up and running. We had about 7-8 running, while other teams were struggling to keep 2-3 alive. There was a moment I remember when Edwards team were completely off and could not do a thing. All of their services but one (rdp, lol) were compromised and were completely down. However, during our stand, we did not record any major breaches into the system. Absolutely no breaches except the Website (but no one figured out how to work with it and we just dumped it a bit). We were the team to keep services alive.

4.11. Deleting Binaries

This was the absolute peak of our performance. I had to bring the Metasploit service back up. The Metasploit by Rapid7 is a Hugh mungus collection of metasploits and tools written in ruby. Simple command `find / -type f | grep '.rb$' | wc -l` yields around 27 thousand files, where one file has to be executed in order to bring the service back up. After learning where to look for an executable, I was completely shocked by what happened next.

`The file specified the interpreter '/bin/sh', which is not an executable command.`

Wat.

Default `sh` not found?! My first thought was that Red Team deleted our binaries but then I thought why would they do that? It is too barbaric. And then it clicked. It was us. During the preparation stage on Saturday and Morning Sunday, some of my teammates were cleaning up unnecessary binaries, some of them included: `sh, screen, tmux, ksu, curl, wget, and etc.` I was in pure horror. No one ever deletes fundamental *NIX binaries from the system. Just and`chmod 600 bin`them on `sh` as an interpreter with `!#/bin/sh`. No wonder it was complaining. And here I tried to `sudo apt install sh screen`and something w orse happened. Aptitude thought the packages are installed. When I tried to delete them, it was yelling at me if I were the madman. The whole aptitude's dependency tree was broken. You never delete binaries like sh from the system. Delete `bash, zsh` but please not something like `sh`.

And then a miracle occurred. Ross and I tried to move his sh binary from his Mac OS to the Kali machine. Surely, it did not work because CPU architecture were different and other stuff. What Ross did after that, was amazing. He booted up his 64-bit Kali Virtual Machine, moved the binary from his VB to his machine and then `scp` the file to the machine. IT WORKED! I gave the whole thing a 7% chance of working, we gambled and we won. We also had to port screen to keep the session alive even if I log off. One of the hackiest experiences of my life but it worked. Metasploit was brought back to life and it was up till the end.

4.12. Golang, GO!

Undeniably, this is my proudest moment during the whole competition.

Also, the scoreboard was not very descriptive. I think they were just checking for status headers instead of contents. Because it was written in python, I assume the code looks something like this:

```
try:
    foo()
except Exception as e:
    print(e)
```

The scoreboard would just show a python exception error like `must be str, not int` and when you see this on your Metasploit, you have no idea what happened there. No connection refused, no error thrown, nothing. I think their python checker expects a return like 200 for success. Instead, it returns everything but status code.

It was almost the end, frustrated a bit but having so much fun, I decided to pull the greatest trick of all. I wrote a Go web-server that would just always return a `HTTP STATUS OK 200` on every request making it look like we are doing fine. I did it, please find the code below

```
dummy.go
--------

package main

import (
       "log"
       "net/http"
)

func serve(w http.ResponseWriter, r *http.Request) {
     w.WriteHeader(http.StatusOK)
}

func main() {
     http.HandleFunc("/", serve)
     log.Fatal(http.ListenAndServe(":80", nil))
}
```

I deployed it. Waited for the services update.

The scoreboard was reporting the service is sending 200 status code, we are good. They really did not check the contents of the return. Go is cross-compatible. Just install golang, change the port number and you are all set. It works for IMAP, SMTP, POP3, FTP. When I tried to `curl` the ports, they all said OK. Proudest moment that we pulled this off. I was in such an awe that it just... worked.

4.13. Penalty Strikes Back!

The competition finished. Earlier. Because we made an ultimatum that we will leave. It was getting stale, and learning opportunities were going down. They finally announced the results and the winner was... Edwards!

Happy for them, for sure. However, I still talked for a while with the organizers about the actual scores. What you are about to hear are the exact words that I was told and I do not have a fact to back it up, here it is.

```
Edwards team got roughly 2500pts and your team has 2000pts.
Don’t forget that you were penalized for 1000pts.
```

W H A T.

The took off a third of our score for VPN penalty? That is not 10% that we agreed on. Even if they took off 10%, we still would have gotten the first place. I am not salty about the results, it was fun and all. But we got penalized for every single action that we took. Remote accessing? 10% off. You want to do anomalies? 15% off. You want your password? Several hundred points.

We started with more than 500pts behind other teams for whatever reason and a thousand taken off because we asked for VPN that did not work and everybody had it. It all just felt a bit unfair and bashing our points like that was not fun. K-State team was not even allowed to qualify or take places in the competition.

5. Conclusion

All and all. I liked the competition. I learned that Powershell language is a thing. I learned some new resources for Metasploit tools. Vyos OS. The team was fantastic and I want to thank every single member for being a part of it, contributing to the team, learning, and just having a good time.

```
Dear Noah, Ellis, Ross, Karen, Yousif, and Tiger,

Thank you for such a fantastic time.
I hope you enjoyed it as well and even more.
```

```
Sincerely,
Sandy
```

I use Arch Linux and I found some like-minded people on the competition (mostly organizers) and we had a lovely chat about life and stuff.

btw, i use arch