IN618 - Security Assignment 1: Security Audit Report for the Death Star Server

Prepared by Matthew Hall

Executive Summary

This document summarises the findings from an independent security audit commissioned by the Rebel Alliance to investigate the Death Star server. A total of three security vulnerabilities were discovered and exploited successfully along with numerous unexploited vulnerabilities. All three of these exploits can be used to great effect to gain unauthorized access to the target server. One of these granted a remote shell as the root user. All ten requested pieces of intelligence were recovered. The system's password file was extracted and all user passwords, in one way or another, were recovered.

Recommendations to improve security include ensuring the patching of out-of-date software, implementing a firewall, investigating for rogue employees, implementing a strict password policy, redesigning a web app hosted on the server and the encryption of sensitive information.

Contents

1 Network Information

1.1 Discovering hosts with nmap

I used the following commands during the host discovery process:

```
# identifies hosts with IP addresses in the network space of 192.168.100.0/24
nmap -sn 192.168.100.0/24
# attempts to identify the operating system of the host with address 192.168.100.1
nmap -O 192.168.100.1
# attempts to identify the operating system of the host with address 192.168.100.115
nmap -O 192.168.100.115
# attempts to identify the operating system of the host with address 192.168.100.253
nmap -O 192.168.100.253
```

1.2 Network Diagram

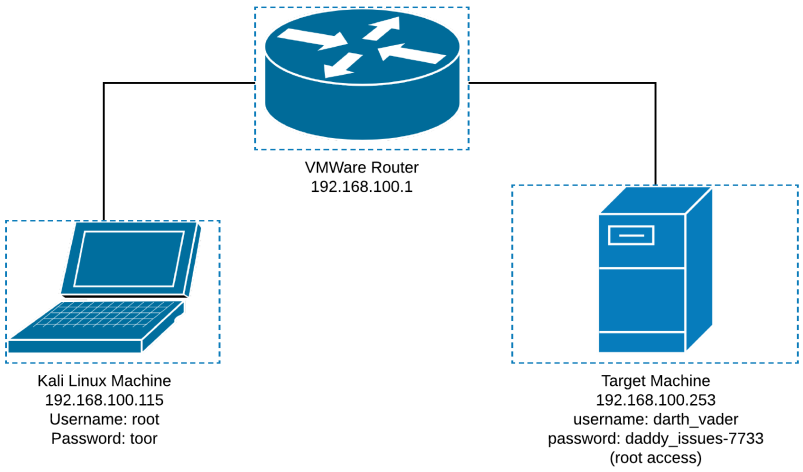


Figure 1: Network diagram of address space `192.168.100.0/24`

1.3 Target Machine Information

- **Target hostname**: `death-star`
- **Target IP address**: `192.168.100.253`
- **Target MAC address**: `00:50:56:86:F2:31`
- **Target MAC vendor**: `VMWare`

- **Operating System version**: `Ubuntu 14.04.01 LTS "Trusty Tahr"`
- **Linux Kernel version**: `3.13`

2 Open Ports and Running Services

By use of the nmap command `nmap -sV 192.168.100.253`, I was able to retrieve the following information about what services on the target machine are running on what ports:

Port Number	State	Service	Version
21/tcp	open	ftp	ProFTPD 1.3.5
22/tcp	open	ssh	OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.10 (Ubuntu Linux; protocol 2.0)
80/tcp	open	http	Apache httpd 2.4.7
111/tcp	open	rpcbind	2-4 (RPC # 100000)
139/tcp	open	netbios-ssn	Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp	open	netbios-ssn	Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
3306/tcp	open	mysql	MySQL (unauthorized)
6667/tcp	open	irc	UnrealIRCd
6697/tcp	open	irc	UnrealIRCd
8067/tcp	open	irc	UnrealIRCd
8080/tcp	open	http	Jetty 8.1.7.v20120910
8181/tcp	open	http	WEBrick httpd 1.3.1 (Ruby 2.3.6 (2017-12-14))
49307/tcp	open	status	1 (RPC # 100024)

3 Vulnerability Identification

The following services on the target machine had exploitable vulnerabilities:

3.1 ProFTPD 1.3.5 (port 21/tcp)

Service Summary:

- Protocol: File Transfer Protocol (FTP)
- Port Number: 21/tcp
- Software Name: ProFTPD
- Software Version: 1.3.5

Exploit Summary:

- Exploit Platform: Metasploit Framework
- Exploit Name: `exploit/unix/ftp/proftpd_modcopy_exec`
- URL: `https://www.rapid7.com/db/modules/exploit/unix/ftp/proftpd_modcopy_exec`
- Exploit Result: Successful (remote FTP session as `www-data` user)
- Reference Number: CVE-2015-3306
- Exploit Severity: High

Version 1.3.5 of ProFTPD has vulnerability *CVE-2015-3306*, which allows attackers to read and write to arbitrary files via the `site cpfr` and `site cpto` commands (*CVE-2015-3306.* 2015). By using the Metasploit Framework module given above and specifying the `SITEPATH` option to be `/var/www/html`, I was able to remotely log in to the machine as the `www-data` user. While logged in, I was able to download files in the `/var/www/html` directory. Among those files was the administrator account details of the phpMyAdmin service and MySQL database management system (see section 5.6). I was also able to traverse the rest of the filesystem after starting an interactive shell.

3.2 UnrealIRCd 3.2.8.1 (port 6667/tcp)

Service Summary:

- Protocol: Internet Relay Chat (IRC)
- Port Number: 6667/tcp
- Software Name: UnrealIRCd

- Software Version: 3.2.8.1

Exploit Summary:

- Exploit Platform: Metasploit Framework
- Exploit Name: `exploit/unix/irc/unreal_irc_3281_backdoor`
- URL: `https://www.rapid7.com/db/modules/exploit/unix/irc/unreal_irc_3281_backdoor`
- Exploit Result: Successful (remote ssh session as `boba_fett` user)
- Reference Number: CVE-2010-2075
- Exploit Severity: *Critical*

During the period of time from November 2009 to June 2010, version 3.2.8.1 of UnrealIRCd contained *CVE-2010-2075*: a backdoor allowing remote attackers to login to the system and execute arbitrary commands. This vulnerability was only present in copies of the software downloaded from particular mirror sites (*CVE-2010-2075.* 2010). This backdoor, if present, can be exploited to open a remote terminal on the target machine. Using exploit given above, I was able to remotely log in to the machine as the `boba_fett` user[1] and use the machine as if I were in a normal ssh session, all without having to supply a password.

3.3 Apache Continuum 1.4.2 (port 8080/tcp)

Service Summary:

- Protocol: Hypertext Transfer Protocol (HTTP)
- Port Number: 8080/tcp
- Software Name: Apache Continuum
- Software Version: 1.4.2

Exploit Summary:

- Exploit Platform: Metasploit Framework
- Exploit Name: `exploit/linux/http/apache_continuum_cmd_exec`
- URL: `https://www.rapid7.com/db/modules/exploit/linux/http/apache_continuum_cmd_exec`
- Exploit Result: Successful (remote ssh session as `root` user)
- Reference Number: EDB-39886
- Exploit Severity: *Critical*

[1] This user had a text file in their home directory containing the login information for the `darth_vader` user, see section 5.5.

The server was hosting the Jetty web server (version 8.1.7v20120910) on port 8080/tcp, which was running the discontinued piece of software Apache Continuum 1.4.2. This version of Apache Continuum has vulnerability *EDB-39886*, which allows attackers to perform arbitrary code injection via a remote shell (*Apache Continuum 1.4.2 - Multiple Vulnerabilities.* 2016). I was able to exploit this using the Metasploit Framework module mentioned above. Using this exploit, I was granted root access to the machine and was able to read and download root-only files, including the `/etc/shadow` file as well as the server's private ssh keys. I was able to do all of this without supplying a password.

4 Information Extraction

As requested, I have located and extracted both the login details of every user account on the system and all information regarding the plans of the Galactic Empire.

4.1 Login Details

Below are the login details for every user account on the Death Star server, along with password hashes and salts represented as base64 encoded strings.

Username	Password	Password Hash (md5crypt)	Password Salt
storm_trooper_1	starwarsrocks	HmlEMwzAUCz0NEitMjx9d1	lnwk829Q
storm_trooper_2	stormtrooper1	7a/Tj3TjRzZYR6mhbZksq0	9AJdbBeI
storm_trooper_3	Lego starwars	N7hkGMUGsyeBgNPwIaF/40	WdB.ds.7
storm_trooper_4	starwarsbatman	5F/sLNVjUPMFtLUdS.hog.	.jX4bdHx
storm_trooper_5	Password1234567	w6VyqglDCJot.Xeb9slLI0	0HHFKzl.
imperial_guards	starwars4life	ebSf18qOk7tgu.iMqf.bi/	v9GI28ar
captain_needa	*darksidegod@hotmail.com*	FO9c2OF4Qf1onEyYkq.gK/	VtXabEV0
admiral_piett	darksideofthemoon	cTG0isRNogwyCeQwCZJXF.	D06DmZeK
admiral_ozzel	darksiderules	eFPtaxBv7sX5IDp8Bc19h.	lfbtu2co
general_veers	darkside3000@hotmail.com	AJlGTM7XYFaY3Ezr7Av/u/	.wG8JtvN
emperor_palpatine	912Deathstars	hy9v3MmcpwRq/G3Dhtu2U1	Sr5iUN.o
darth_sidious	7ujMko0admin	Mp7O4bzX8bmWsGGV8ZrVY0	TyPfW4pp
boba_fett	*bountyhunter1976*	GfHV875pepnKEg.JC.zYY/	eOFOT0eZ
death_star_admin	<3DeathStars<3	erKQWB6ZTfw2efmZMPDME.	HnIyNzWr
darth_vader	*daddy_issues-7733*	0TkhYTZFnI1srHEzG1TrO/	AnAm41bc

The plaintext passwords that are written in italics were found in plaintext form (see section 5.5). The others were found in their salted md5crypt form and were recovered with hashcat using the rockyou dictionary and the `default_pass_for_services_unhash` dictionary included in Metasploit. Fourteen out of fifteen passwords were able to be recovered with the use of hashcat.

4.2 Plans of the Empire

In order to locate the requested intelligence files, I used the following command:

```
root@deathstar:/# locate -i --regex '(death|rebel)[_-]?(star|alliance)'
```

Below are the names and locations of the extracted intelligence files. The contents of the files can be found in the Intelligence Appendix (Appendix A).

In the directory `/home/death_star_admin/death-star_plans`:

- `deathstar-crafts.png`, Figure 3
- `deathstar-cross-section.png`, Figure 4
- `deathstar-operations.png`, Figure 5
- `deathstar-summary.png`, Figure 6
- `deathstar-technical-specs-diagram.png`, Figure 7

In the directory `/home/general_veers/rebel-information`:

- `rebel-alliance-fleet-1.jpg`, Figure 8
- `rebel-alliance-fleet-2.jpg`, Figure 9

In the directory `/home/darth_sidious`:

- `death-star-weakness.png`, Figure 10

In the directory `/home/darth_vader`:

- `i-love-my-death-star.jpg`, Figure 11

In the directory `/opt/proftpd/share/locale`:

- `deathstarinfographic.pNg`, Figure 12

5 Security Recommendations

Over the course of this audit, I have discovered multiple vulnerabilities in key services running on the Death Star server and exploited them. I have also found many other security issues during my investigation of the server. Here I have compiled the details of all of the security flaws I could find with this server, as well as various recommendations to fix these problems for the future.

5.1 Upgrade Operating System

The operating system on the Death Star server was Ubuntu 14.04 LTS, which is due to stop receiving security updates in April 2019 (Canonical Group Ltd 2019). Because of this, it is imperative that the operating system is upgraded to a new version.

Security recommendations regarding unsupported operating systems:

1. Upgrade to Ubuntu 18.04 LTS, which will receive security updates until 2023. Additionally, installations of Ubuntu can be upgraded to later releases easily via the command line by use of the `do-release-upgrade` command.

5.2 Upgrade Software

Many of the services provided by the Death Star server were running out-dated software that contained vulnerabilities. Some pieces of software have even stopped receiving updates altogether (see section 3.3). Out-of-date software is much more likely to contain vulnerabilities than up-to-date versions.

Security recommendations regarding out-of-date software:

1. Update your software. *Now.*

5.3 Firewall

The Death Star server did not have any kind of firewall. Because of this, an attacker could easily scan every port on the system and send malicious data packets to the server.

Security recommendations regarding firewalls:

1. I would recommend the installation of a physical firewall, ideally running an OS designed for the purpose of being a firewall/router such as pfSense.
2. If the budget does not allow this, some kind of packet-filtering process should be implemented on the server.

I personally can vouch for the use of iptables, a software firewall that is available on virtually every Linux distribution.

5.4 Inside Threats

During my investigation, I found a questionable Ruby script in the `/home/darth_vader/poc/payroll_app` folder, implying that they were created by this user. Upon analysing the contents of the file, it appears to perform an SQL injection attack on the Death Star's payroll app (see section 5.6). All of this implies that there is a rogue employee of the Empire using the Death Star server.

Security recommendations regarding the possibility of a rogue employee:

1. *This cannot be ignored and must be investigated as soon as possible.*

5.5 Insecure Password Storage and Bad User Passwords

During my attempts to recover passwords, I was able to recover passwords in three different ways.

The first method I was able to use was simply reading a user's password that was written down in a file. Incidentally, this was the password of the `darth_vader` user. I was able to log in to the account with the password I found and created another account for myself in the process. I was able to do this because the `darth_vader` user had superuser access.

The second method I was able to use to recover passwords was by exploiting the payroll app. The payroll app on the Death Star server has numerous security issues (see section 5.6) and one of those issues was the fact that the user passwords were stored in plaintext. I went through each password I found in the app and attempted to log in to the corresponding user account. By use of this method I was able to log in as the `captain_needa` and `boba_fett` users. Incidentally, the `boba_fett` user was the user that had the password for `darth_vader` in their home directory.

The third method I used[2] was password cracking. By using the `hashcat` tool I was able to crack the remaining user's passwords with two dictionary attacks (see section 4.1). Using a relatively powerful GPU[3] I successfully recovered the remaining passwords with roughly 20 minutes of computing time. The only password I was *not* able to crack belonged to the `darth_vader` user. The other passwords on the server were able to be cracked because of two problems. The first problem is that the user's passwords were relatively common passwords, whether the users were aware of this or not. The second problem is that the passwords were hashed+salted using md5crypt. This hashing algorithm is too fast to be used for storing passwords and should not have been used on the server. This is a problem because if a hashing algorithm is too fast, it becomes easy for attackers to guess what a hashed password might be.

> Security recommendations regarding passwords and password storage:
>
> 1. *ALL EMPLOYEES MUST CHANGE ALL OF THEIR PASSWORDS IMMEDIATELY*
> 2. All employees must be educated about
>   (a) The dangers of using easy-to-guess passwords
>   (b) The dangers of storing/writing down passwords in plaintext
>   (c) The dangers of reusing passwords
>   (d) How to think of good passwords (see below for example method)
> 3. Future passwords should be stored by salting and hashing using a secure, trusted cryptographic hashing algorithm (see below)
> 4. The owner of the `boba_fett` user should be sentenced to death by sarlacc

A reasonable technique for making a password is as follows (Riley 2016):

1. Choose three random, 8-10 letter words
2. Choose another reasonably long, more obscure word (or even a made-up word)
3. Pick a symbol and put it between two of the words

As for hashing, I would recommend using the SHA-512 algorithm instead of md5crypt, since SHA-512 is slower and therefore harder to guess a hashed password. In addition, it is practically impossible[4] to generate a hash collision with this algorithm.

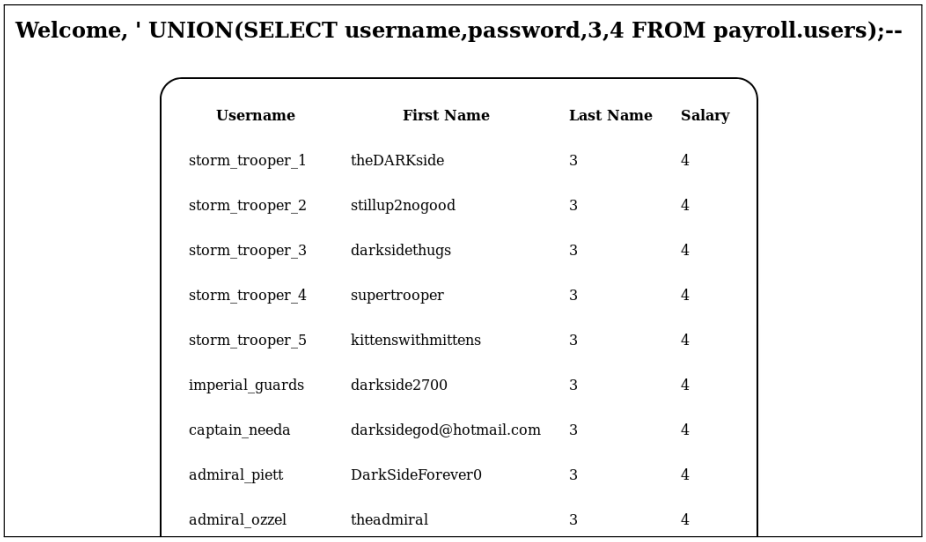
[2] And expected to use
[3] A GTX 1050ti
[4] At the time of writing

5.6 Poorly Designed Payroll App

The payroll app on the server is incredibly insecure and needs to be replaced as soon as possible. Firstly, the source code contains the password for the `root` account on the MySQL server. This means that anyone on the server could view the source code, read the password and log in as the root MySQL user. This can be remedied by changing the file permissions so that only the root user can view the contents of the file.

Secondly, the app is vulnerable to SQL injection to the extent that I was able to print arbitrary information from the database to the screen via the username input box. The input that I used to extract the following data was:

```
' UNION(SELECT username,password,3,4 FROM payroll.users);--
```



Welcome, ' UNION(SELECT username,password,3,4 FROM payroll.users);--

Username	First Name	Last Name	Salary
storm_trooper_1	theDARKside	3	4
storm_trooper_2	stillup2nogood	3	4
storm_trooper_3	darksidethugs	3	4
storm_trooper_4	supertrooper	3	4
storm_trooper_5	kittenswithmittens	3	4
imperial_guards	darkside2700	3	4
captain_needa	darksidegod@hotmail.com	3	4
admiral_piett	DarkSideForever0	3	4
admiral_ozzel	theadmiral	3	4

Figure 2: Results from SQL injection

This can be fixed by changing the app's code to use prepared statements, which are currently considered the safe way to interact with a database via a web form.

Finally, as you may have deduced from the screenshot above (figure 2), the app stores the user's passwords in plaintext. This is incredibly bad practice for a number of reasons. For one, users tend to reuse the same password for multiple services[5], which means an attacker can find passwords on insecure websites and login on other websites using the same username and password combination. Also, a rogue insider[6] with access to the database can gain access

[5] And in this case, they did (see section 5.5)
[6] Which there may be in this case (see section 5.4)

much more easily than an outside attacker, meaning they can steal the passwords of everyone using the app. Additionally, if there was the ability for a user to click a 'I forgot my password' button and send them their password via email, the email could be intercepted and the password stolen that way.

Security recommendations regarding the payroll app:

1. *ALL EMPLOYEES MUST CHANGE ALL OF THEIR PASSWORDS IMMEDIATELY*
2. The payroll app must be re-programmed to use prepared statements
3. The app should use a separate authentication service, such as OAuth to handle logging in
4. If the above is not possible, future passwords must be stored by salting and hashing with a modern, trusted cryptographic hashing algorithm (see recommendations in section 5.5)

5.7 Sensitive Information

I found it relatively easy to locate the pieces of intelligence I was tasked with finding. Unfortunately for the Empire, I also found the intelligence easy to read. What I mean by this is that these files has no special protections in place to prevent them from being read by those who are not meant to read them.

Security recommendations regarding the storing of sensitive information:

1. Encrypt the files with an encryption tool such as PGP.

6 Conclusions

This document summarised the findings from an independent security audit commissioned by the Rebel Alliance to investigate the Death Star server. A total of three security vulnerabilities were discovered and exploited successfully and root access was acquired. This is an incredibly insecure server and drastic measures would need to be taken to mitigate the security risks present and secure the sensitive information stored on the system. The security recommendations given in this report should be applied to the machines of the Rebel Alliance and then another security audit should be performed to re-asses the security of their servers.

References

Apache Continuum 1.4.2 - Multiple Vulnerabilities. (June 2016). Available from Exploit Database, EDB-ID 39886. Accessed 6 April 2019. URL: `https://www.exploit-db.com/exploits/39886`.

Canonical Group Ltd (Apr. 2019). *Releases*. Wiki page. Accessed 6 April 2019. URL: `https://wiki.ubuntu.com/Releases`.

CVE-2010-2075. (May 2010). Available from MITRE, CVE-ID CVE-2010-2075. Accessed 6 April 2019. URL: `https://cve.mitre.org/cgi-bin/cvename?name=CVE-2010-2075`.

CVE-2015-3306. (Apr. 2015). Available from MITRE, CVE-ID CVE-2015-3306. Accessed 6 April 2016. URL: `https://cve.mitre.org/cgi-bin/cvename?name=CVE-2015-3306`.

Riley, Sean (July 2016). *How to Choose a Password.* Video uploaded to YouTube. Accessed 6 April 2019. URL: `https://www.youtube.com/watch?v=3NjQ9b3pglg&t=7m5s`.

Appendices

A Intelligence Appendix

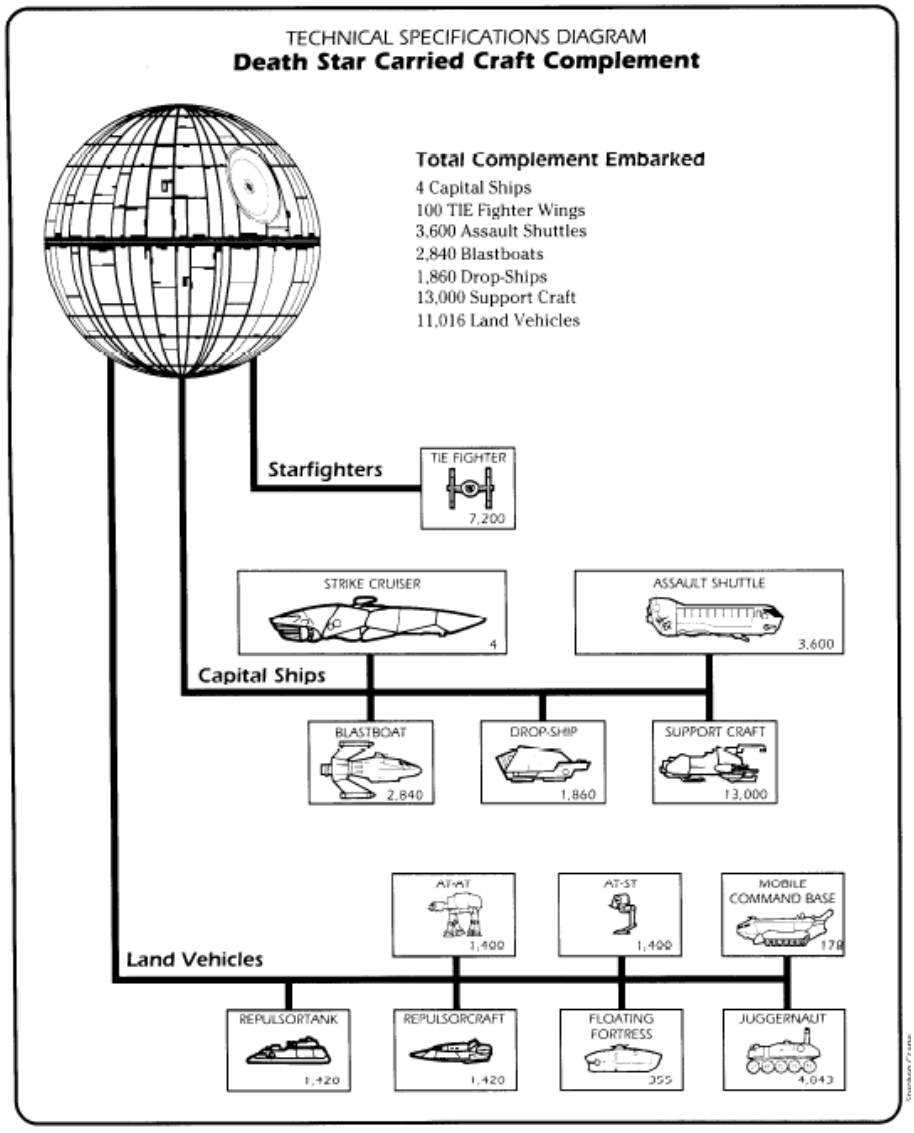


Figure 3: `deathstar-crafts.png`

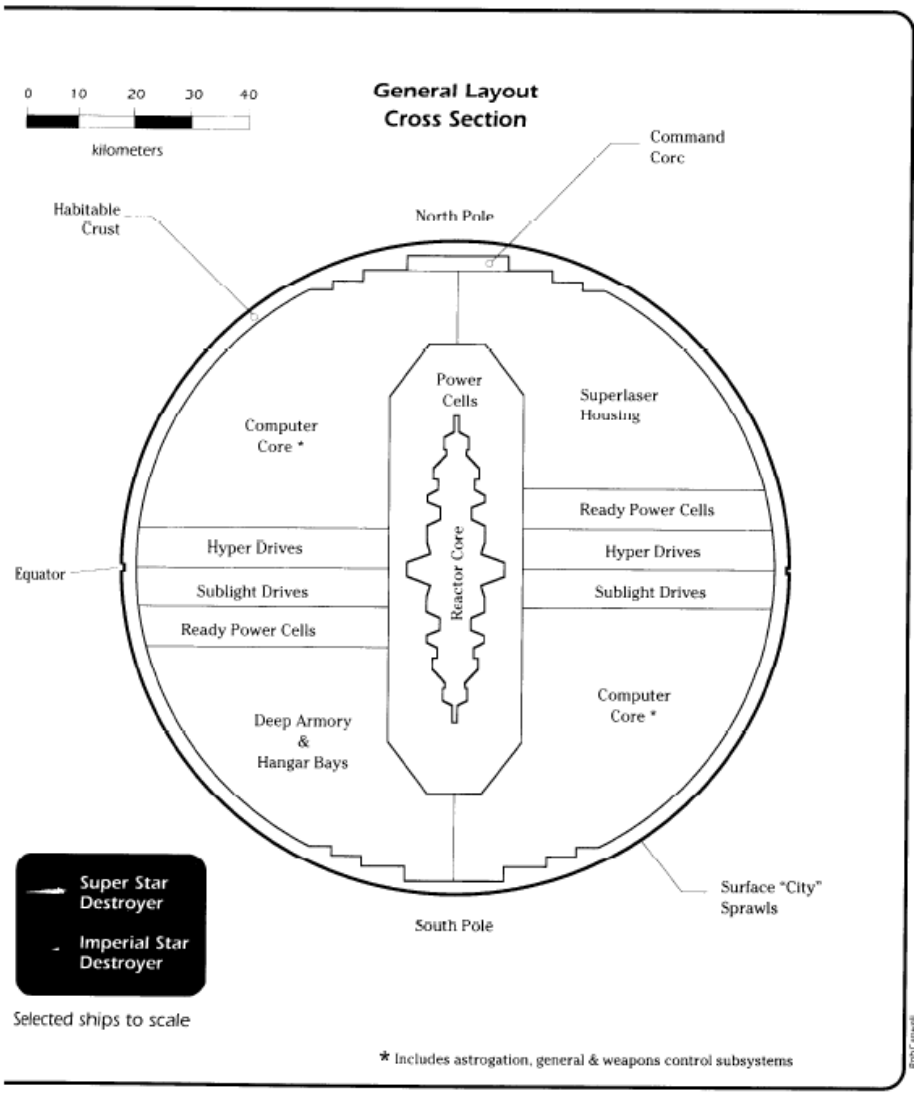


Figure 4: `deathstar-cross-section.png`

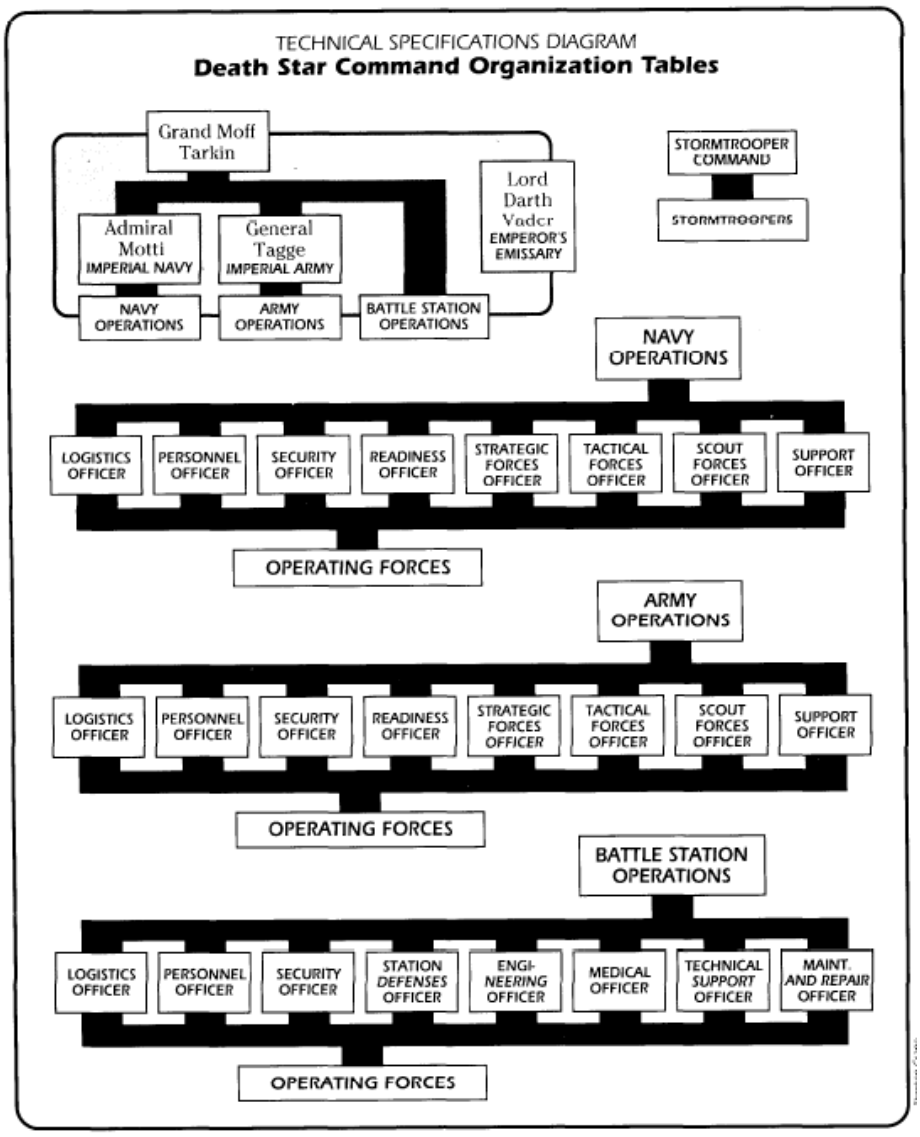
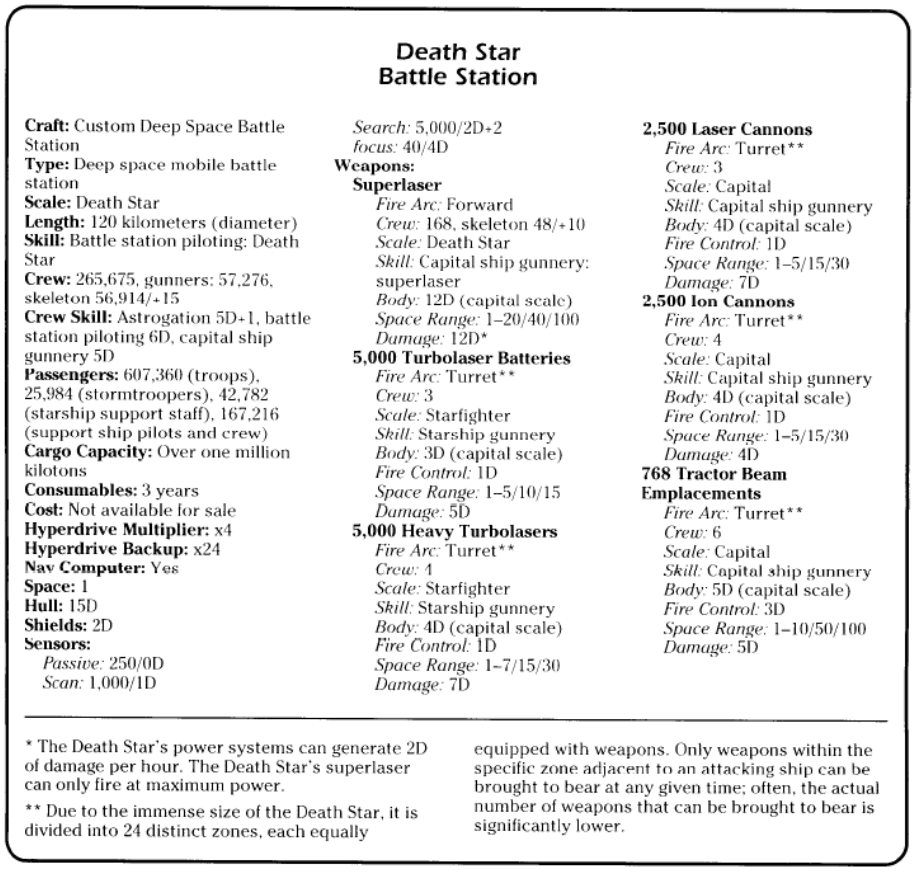


Figure 5: `deathstar-operations.png`



Death Star
Battle Station

Craft: Custom Deep Space Battle Station
Type: Deep space mobile battle station
Scale: Death Star
Length: 120 kilometers (diameter)
Skill: Battle station piloting: Death Star
Crew: 265,675, gunners: 57,276, skeleton 56,914/+15
Crew Skill: Astrogation 5D+1, battle station piloting 6D, capital ship gunnery 5D
Passengers: 607,360 (troops), 25,984 (stormtroopers), 42,782 (starship support staff), 167,216 (support ship pilots and crew)
Cargo Capacity: Over one million kilotons
Consumables: 3 years
Cost: Not available for sale
Hyperdrive Multiplier: x4
Hyperdrive Backup: x24
Nav Computer: Yes
Space: 1
Hull: 15D
Shields: 2D
Sensors:
Passive: 250/0D
Scan: 1,000/1D
Search: 5,000/2D+2
focus: 40/4D
Weapons:
Superlaser
Fire Arc: Forward
Crew: 168, skeleton 48/+10
Scale: Death Star
Skill: Capital ship gunnery: superlaser
Body: 12D (capital scale)
Space Range: 1–20/40/100
Damage: 12D*
5,000 Turbolaser Batteries
Fire Arc: Turret**
Crew: 3
Scale: Starfighter
Skill: Starship gunnery
Body: 3D (capital scale)
Fire Control: 1D
Space Range: 1–5/10/15
Damage: 5D
5,000 Heavy Turbolasers
Fire Arc: Turret**
Crew: 4
Scale: Starfighter
Skill: Starship gunnery
Body: 4D (capital scale)
Fire Control: 1D
Space Range: 1–7/15/30
Damage: 7D
2,500 Laser Cannons
Fire Arc: Turret**
Crew: 3
Scale: Capital
Skill: Capital ship gunnery
Body: 4D (capital scale)
Fire Control: 1D
Space Range: 1–5/15/30
Damage: 7D
2,500 Ion Cannons
Fire Arc: Turret**
Crew: 4
Scale: Capital
Skill: Capital ship gunnery
Body: 4D (capital scale)
Fire Control: 1D
Space Range: 1–5/15/30
Damage: 4D
768 Tractor Beam Emplacements
Fire Arc: Turret**
Crew: 6
Scale: Capital
Skill: Capital ship gunnery
Body: 5D (capital scale)
Fire Control: 3D
Space Range: 1–10/50/100
Damage: 5D

* The Death Star's power systems can generate 2D of damage per hour. The Death Star's superlaser can only fire at maximum power.

** Due to the immense size of the Death Star, it is divided into 24 distinct zones, each equally equipped with weapons. Only weapons within the specific zone adjacent to an attacking ship can be brought to bear at any given time; often, the actual number of weapons that can be brought to bear is significantly lower.

Figure 6: `deathstar-summary.png`

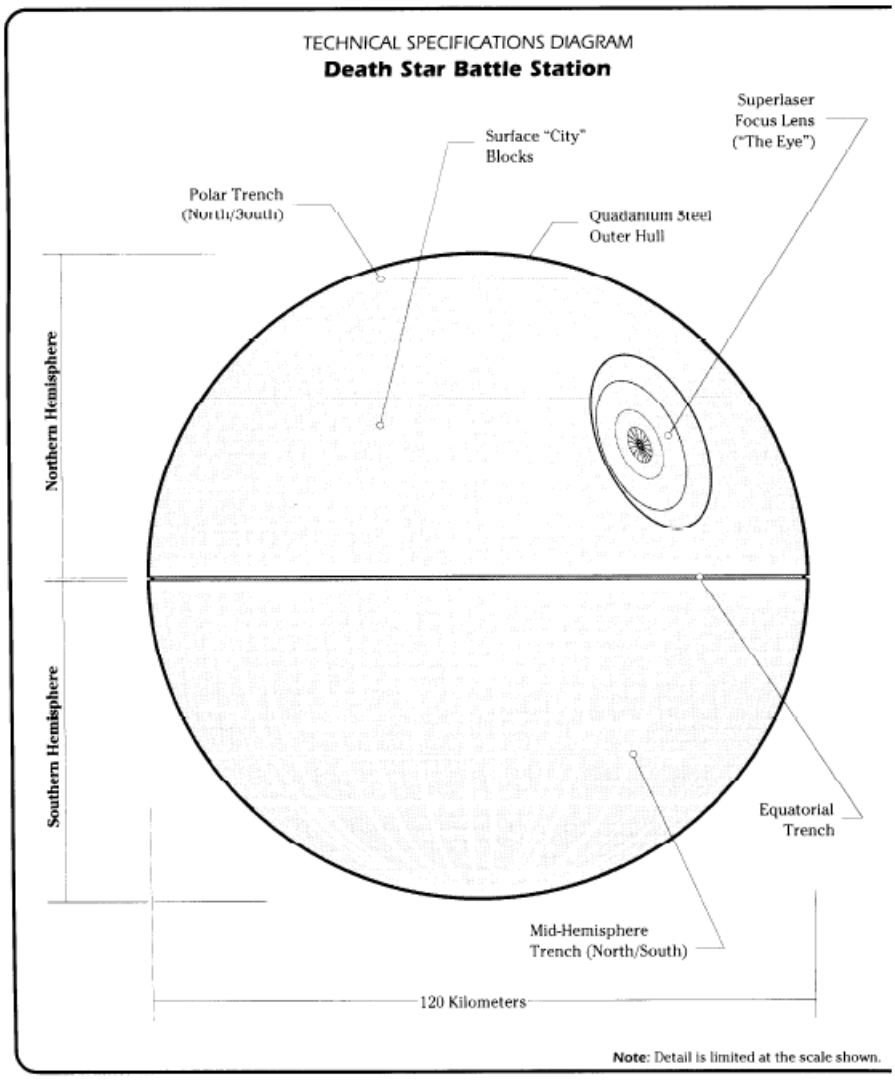


Figure 7: `deathstar-technical-specs-diagram.png`

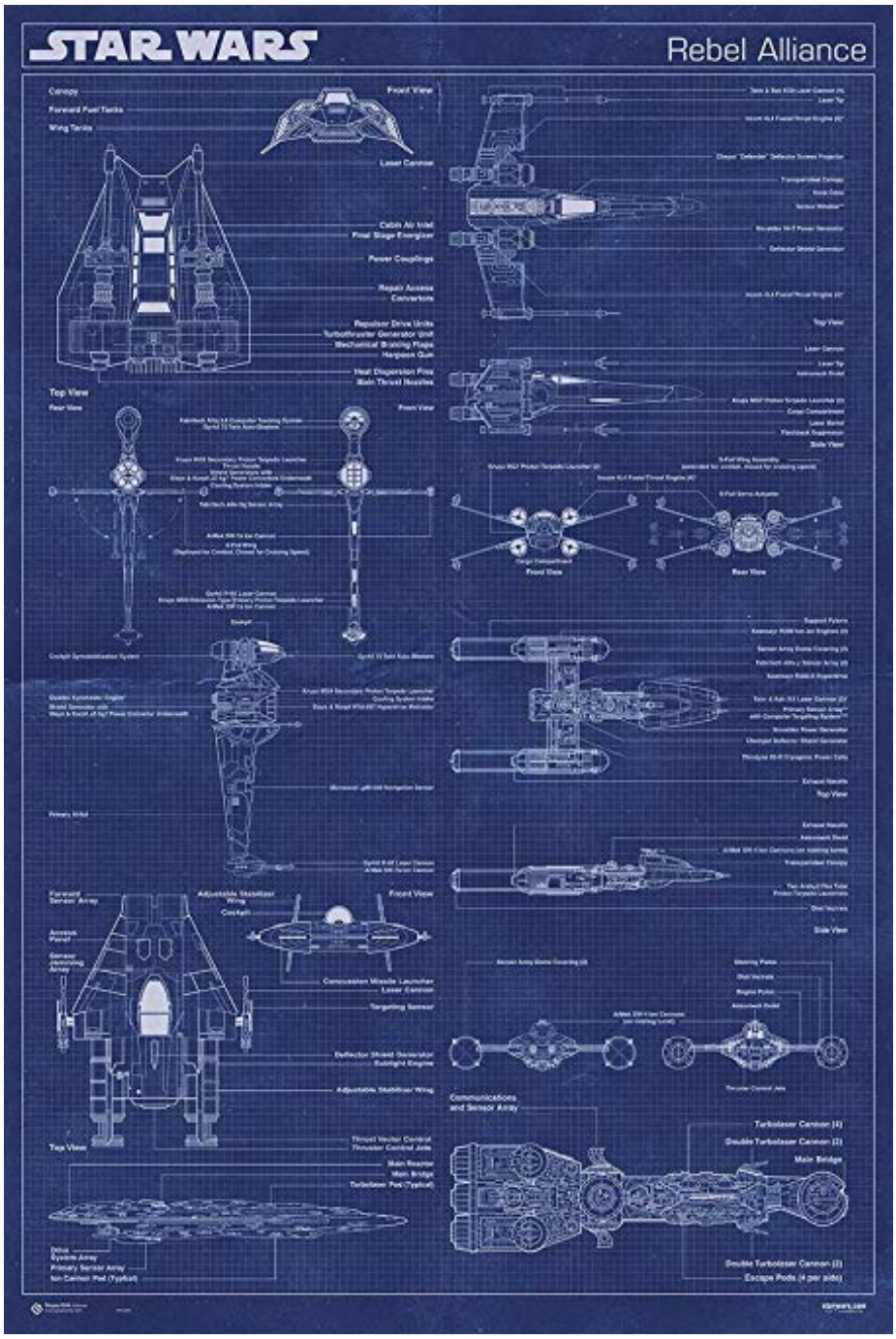


Figure 8: `rebel-alliance-fleet-1.jpg`

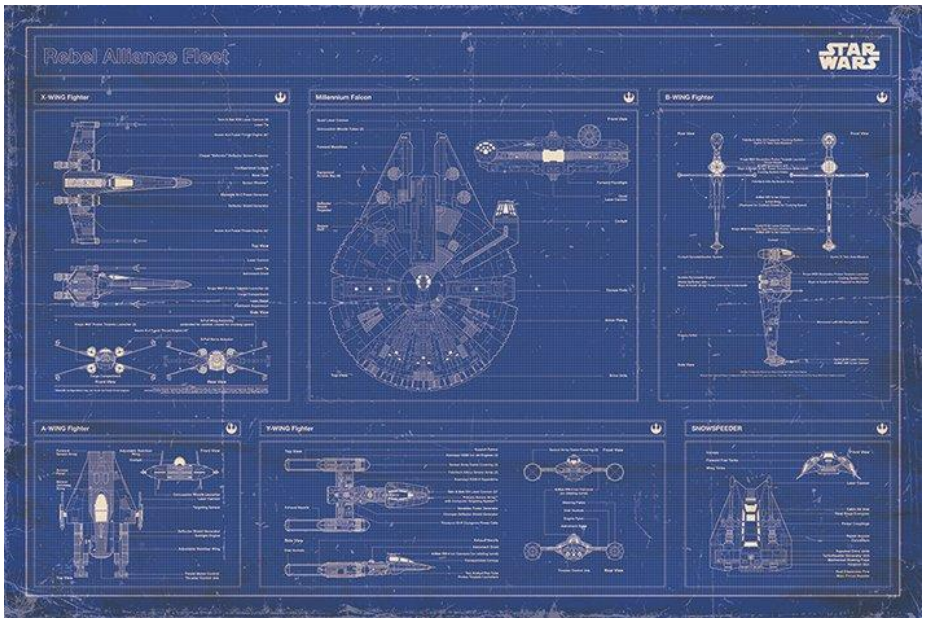


Figure 9: `rebel-alliance-fleet-2.jpg`

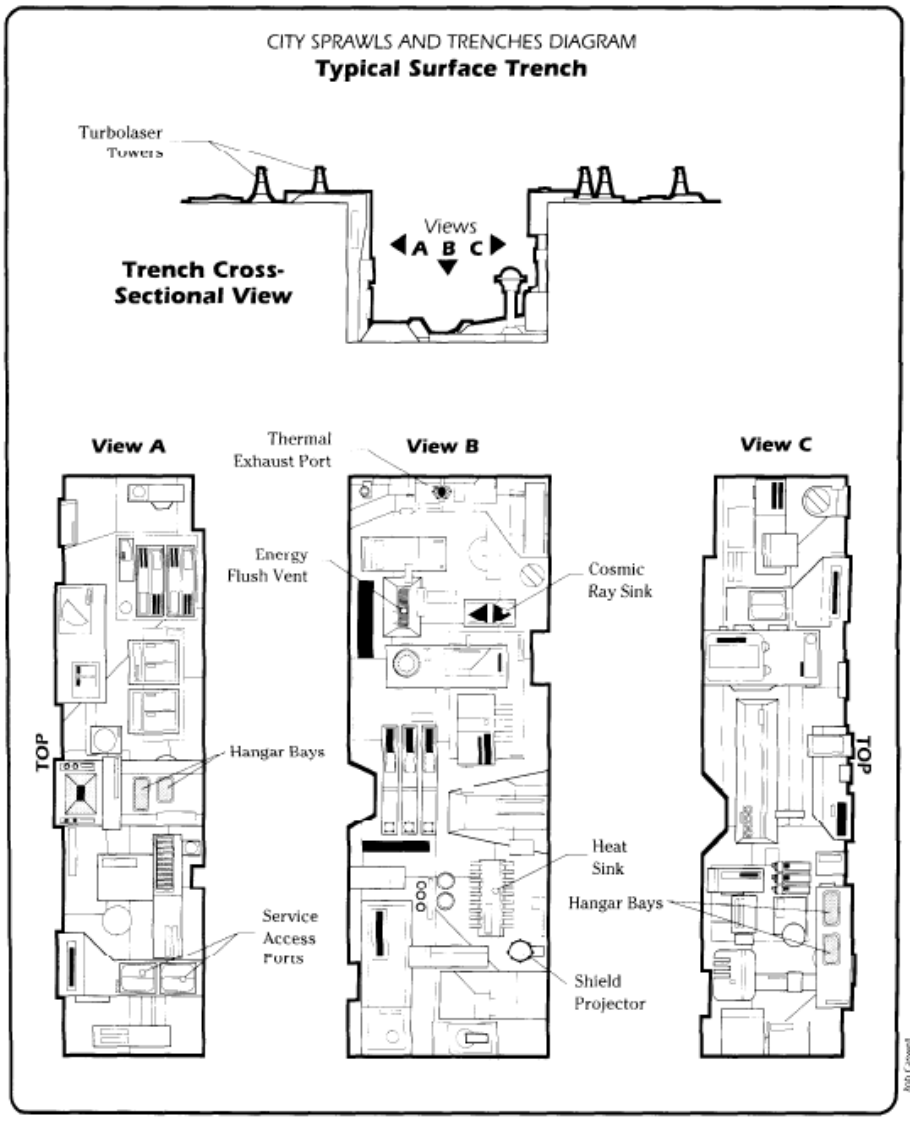


Figure 10: `death-star-weakness.png`

Figure 11: `i-love-my-death-star.jpg`

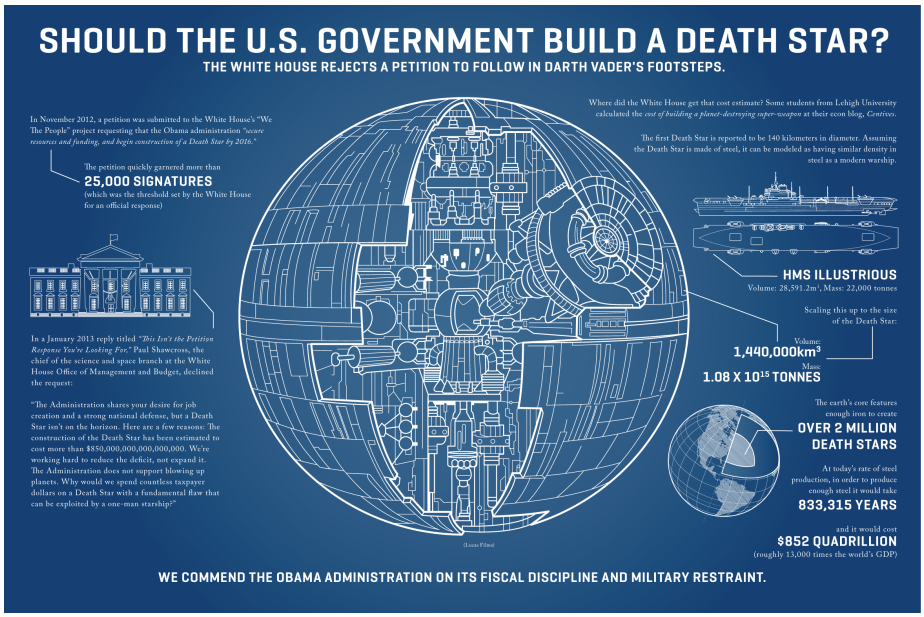


Figure 12: `deathstarinfographic.pNg`