Protecting Defence Research and Development Organisation from the next "Stuxnet"

An extensive report by
Harsha Aditya,
Microsoft cybersecurity expert

To
National Critical Information Infrastructure Protection Centre (NCIIPC)

Microsoft Engage 2022
Final project
22 June 2022

Index page no.

Chapter -1

Methodology and Assumptions

Introduction to the organisation

Defence Research and Development Organisation (DRDO)

The **Defence Research and Development Organisation** (**DRDO**) is the premier agency under the Department of Defence Research and Development in the Ministry of Defence of the Government of India, charged with the military's research and development, headquartered in Delhi, India. It was formed in 1958 by the merger of the Technical Development Establishment and the Directorate of Technical Development and Production of the Indian Ordnance Factories with the Defence Science Organisation. With a network of 52 laboratories engaged in developing defence technologies, covering various fields, like aeronautics, armaments, electronics, land combat engineering, life sciences, materials, missiles, and naval systems, DRDO is India's largest and most diverse research organisation.

"Balasya Mulam Vigyanam"—the source of strength is science-drives the nation in peace and war. DRDO has a firm determination to make the nation strong and self-reliant in terms of science and technology, especially in the field of military technologies.

Today, DRDO is a network of more than 50 laboratories which are deeply engaged in developing defence technologies covering various disciplines, like aeronautics, armaments, electronics, combat vehicles, engineering systems, instrumentation, missiles, advanced computing and simulation, special materials, naval systems, life sciences, training, information systems and agriculture. Several major projects for the development of missiles, armaments, light combat aircraft, radars, electronic warfare systems etc are on hand and significant achievements have already been made in several such technologies.

Indian forces are using numerous indigenous technologies produced by the DRDO, including Varunastra, Maareech, Ushus, and TAL by the navy; Electronic Warfare

Technologies, radars, composite materials for LCA, AEW&C, Astra, LCA Tejas by the airforce; and ASAT, BrahMos, ASTRA, Nag missile, SAAW, Arjun MBT Mk 1A, 46-metre Modular Bridge, MPR, LLTR Ashwin by the army.

Hence the threat from high-level targeted cyber attacks is immense as the technologies produced here are the most vital to the country. DRDO is a centre of development for some of the world's best nuclear warheads therefore a great target for Stuxnet-like attacks.

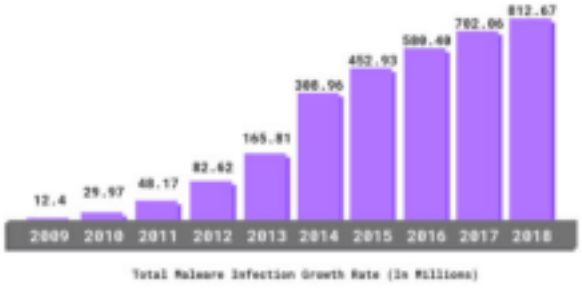
Cybersecurity definition and need

"Prevention of damage to, protection of, and restoration of computers, electronic communications systems, electronic communications services, wire communication, and electronic communication, including information contained therein, to ensure its availability, integrity, authentication, confidentiality, and nonrepudiation."

-NIST, National Institute of Standards of Technology

India faces a vital need to protect critical infrastructures such as banks, satellites, automated power grids, and thermal power plants from cyberattacks. Indeed, the Indian government has admitted that there has been a large spike in cyberattacks against establishments such as the banking and financial services sector. Malicious activity on the Internet in India has ranged from viruses, hacking, identity theft, spamming, email-bombing, web defacement, and cyber defamation, to the denial of service.

For example, even though the country ranks eighty-fifth in net connectivity compared to other countries globally, it holds the seventh spot in terms of cyberattacks (Express News Service 2014). Strikingly, the number of cyberattacks rose from 23 in 2004 to 62,000 by mid2014 (The Economic Times 2014). The year 2013 alone saw a 136 per cent increase in cyber threats and attacks against government organizations as well as a 126 per cent increase in attempts against Indian financial services organizations (Athavale 2014). Approximately 69 per cent of attacks have targeted large enterprises (IANS 2014). Finally, according to a report by security software maker Symantec, four out of ten attacks in 2014 were carried out on nontraditional services industries like business, hospitality, and personal services (Indo-Asian News Service 2014). A clear need, therefore, exists for India to develop an effective cyber crisis management plan, to address these and similar challenges.



Source: PurpleSec

A successful attack can compromise the confidentiality, integrity, and availability of an ICT system and the information it handles. Cyber theft or cyber espionage can result in the exfiltration of financial, proprietary, or personal information from which the attacker

can benefit, often without the knowledge of the victim. Denial-of-service attacks can slow or prevent legitimate users from accessing a system. Botnet malware can give an attacker command of a system for use in cyberattacks on other systems. Attacks on industrial control systems can result in the destruction of the equipment they control, such as generators, pumps, and centrifuges.

source: https://www.academicapress.com/journal/v1-1/Parmar_Cybersecurity-in-India.pdf

Vulnerabilities and assumptions

In cybersecurity, a vulnerability is a weakness that can be exploited by cybercriminals to gain unauthorized access to a computer system. After exploiting a vulnerability, a cyberattack can run malicious code, install malware and even steal sensitive data. vulnerabilities are not introduced to a system; rather they are there from the beginning. There are not many cases involving cybercrime activities that lead to vulnerabilities. They are typically a result of operating system flaws or network misconfigurations. A vulnerability, which has at least one definite attack vector is exploitable. Attackers will, for obvious reasons, want to target weaknesses in the system or network that are exploitable. Of course, vulnerability is not something that anyone will want to have, but what you should be more worried about is it being exploitable.

Below are some of the most common types of cybersecurity vulnerabilities:

1. System Misconfigurations.
2. Out-of-date or Unpatched Software.
3. Missing or Weak Authorization Credentials.
4. Malicious Insider Threats.
5. Missing or Poor Data Encryption.
6. Zero-day Vulnerabilities.

Defenders can often protect against weaknesses, but three are particularly challenging: inadvertent or intentional acts by insiders with access to a system; supply chain vulnerabilities, which can permit the insertion of malicious software or hardware during the acquisition process; and previously unknown, or zero-day, vulnerabilities with no established fix. A great example of this would be Stuxnet as it had exploited all three vulnerabilities mentioned above.

We assume that DRDO being a defence facility has a great amount of cyber security in place already such as Intrusion detection systems (IDS), firewalls and so on which are reliable. Good assumptions should be universally verifiable. From a security perspective, that anyone can verify an assumption implies the lack of insider advantage an assumption is not universally verifiable, then we are forced to make additional assumptions regarding

mechanisms to prevent the ability to exploit insider advantage.

Methodology

We have taken a set assumptions as escribed earlier and shown what type approaches could be taken with description and strategies which would be best for the organisation .Also the outcomes and lessons which could be taken from previous Stuxnet attack . We have also described about legal and treaty assumptions.We have also given some prototype codes on how to detect botnets. We have also defined the benchmarks for success and differen t questions related to it.

For active approaches, some of the components of the security protocol S include a) design and deployment of intrusion detection systems (IDS), and firewalls, b) strategies for verification and certification of the integrity of software, and hardware; c) isolation mechanisms like virtualization, containers, etc. Some of the assumptions behind such approaches include

1. the correctness of IDSes/firewalls;
2. that software and hardware can be verified to be clear
of unintended functionality,
3. entities who verify correctness of IDSes or certify software/hardware components are not malicious / incompetent; and
4. the additional active components introduced to protect
assets do not themselves introduce new vulnerabilities,

etc.

For passive approaches based on cryptographic techniques, the important components of the security protocol are cryptographic primitives like block-ciphers, hash functions, and asymmetric encryption / signature schemes. Cryptographic primitives are essentially non-deterministic algorithms with well-defined inputs and outputs. By suitably chaining outputs of a cryptographic primitive to inputs of other cryptographic primitives, a wide range of useful cryptographic protocols can be constructed.

Chapter 2

The ‘Challenge’

Cyber -deterrence challenges

Much of theresearch on deterrence strategy and cyber warfare is based on an American perspective. It examines the possibility of successfully implementing the strategy of deterrence in order to prevent cyber attacks, or analyzes the way the US can use cyber warfare in order to deter other threats it faces. These studies make it clear that the possibility of successful deterrence against cyber attacks is limited with regard to each of the dimensions required for its success: the existence of capability (weapons), the credibility of the threat, and the ability to convey the threatening message to the potential challenger.
There are different ways in which actors can try to prevent their enemies from taking undesirable action.The first essential condition for successful deterrence by punishment is that the defender be able to exact a price from the challenger.A second condition for successful deterrence is the credibility of the threat. In order for the deterrence threat to be effective, the defender must be ready to use the capabilities at its disposal. The third condition is effective delivery of the messages to the challenger concerning the two previous conditions – capabilities and intentions. Inother words, the challenger must be aware of the defender's capabilities andits willingness to use them. Researchers who have developed psychological approaches to deterrence claim that this condition is the most important
of all.

As mentioned earlier three key vulnerabilities are the most challenging

- supply chain vulnerabilities, which can permit the insertion of malicious software or hardware during the acquisition process; and
- previously unknown, or zero-day, vulnerabilities with no established fix.
- inadvertent or intentional acts by insiders with access to a system

All these three vulnerabilities were used full effect by the creators of the Stuxnet. Supply chain vulnerability can be defined as 'an exposure to serious disturbance, arising from risks within the supply chain as well as risks external to the supply chain'. Such risks can be usually taken advantage by malicious attempts by hackers as such systems are usually highly automated.

Zero day vulnerability is a software vulnerability discovered by attackers before the vendor has become aware of it. Because the vendors are unaware, no patch exists for zero-day vulnerabilities, making attacks likely to succeed. Hence it is very difficult to stop unless noticed the attack in it's earlier stages because finding what the hackers targeted and patching it up is along tedious process as was seen in the case of Stuxnet.

Social engineering played a crucial role of injecting the Stuxnet worm into the systems in Iranian nuclear plant. Intentional attacks by insiders is something very hard to control from organisation point as they are no longer part of the organisation but still have all the information regarding the organisation. Hence very difficult to trace the source. As cyber warfare is quickly evolving even many goverments have not placed a policy framework for addressing cyber attacks and many not yet so developed in order to have cyber deterrence facilities.\

Unpredictable effects of cyberattacks

"*A Harvard graduate began an experiment to see how many computers were connected to the Internet. 24 hours later, 10% of all computers around the world had been taken down and the damages soared into the millions. Robert Tappan Morris had inadvertently created the first ever computer worm.*

Once Morris realized the speed at which his program was replicating, he tried to send instructions to the victims to dismantle the worm and curb the attack. But it was too late. He was indicted one year later and faced fines of over $10,000. "

-1988

"*The DarkSide ransomware group most likely only intended to hit the IT system and corporate business operations of Colonial Pipeline and underestimated the full impact the malware would have. The consequences were disastrous, halting the supply of fuel across the East Coast, leading to gas shortages, hoarding, and spikes in gasoline prices around the world.*"

- Current day scenario

Misjudging the impact and collateral damage of a cyber-attack can lead to a range of unintended ramifications, from a cyber-crime group feeling increased heat from law enforcement to a nation state escalating a conflict greater than they intended. It is for this reason that many ransomware groups historically have tended to keep their affairs under the radar. Over 70% of ransomware attacks target SMBs.

"Last month, ransomware claimed its first life. German authorities reported a ransomware attack caused the failure of IT systems at a major hospital in Duesseldorf,

and a woman who needed urgent admission died after she had to be taken to another city for treatment."

- Cybercrime magazine

Unfortunately, while many cyber-crime groups pledge to avoid larger bodies like hospitals and critical infrastructure, the allure of fast payouts for record-breaking ransoms has led to the healthcare sector, even vaccine efforts, being a heavy target for ransomware actors. Also it leads to access of sophisticated equipment leading to supply chain disruptions and so on.

Source:https://www.darktrace.com/en/blog/unintended-consequences-when-cyber-attacks-go-wild/

Damage due to counter retaliation

"Cybercrime To Cost The World $10.5 Trillion Annually By 2025
A cyberattack could potentially disable the economy of a city, state or our entire country."
-Cybercrime magazine

Their is very little information regarding this in open and their are various technical and legal aspects to it. First of all counter retaliation often involves unpredictable effects as mentioned earlier as usually such attacks are done at a large scale to make the opponent understand the defender's potential. Also such attacks often lead to question about ethicality and legal domain. Such attacks on Cyber Physical Systems (CPS) can lead to loss of life and property. In the case of Stuxnet hundreds of centrifuges worth millions were lost.

Chapter 3

Legal and treaty assumptions

Due to the magnitude of cyber threats, there is an urgent need for countries to make headway on laws and treaties addressing cyber warfare.
Jurisdiction is a huge problem when it comes to cyber attacks because usually such attacks are global in nature and generally nation states are involved in Stuxnet like attacks. Wildly placing allegations could have significant geo-political effects and create war like situation
Currently we assume Ministry of Electronics and Information Technology, The Government of India law frame work .
. The Personal Data Protection Bill, 2019 (PDPB) has a lot of landmark provisions for data protection and
privacy of individuals.
Some laws and judgements that enforce the safety of consumer's data across telecommunication networks are as following:
• **Transparency in the processing of personal data**: Clause 23 attempts to ensure transparency in the processing of personal data by requiring the data controller to be advised by the fiduciary and to make accessible information. This provision introduces a new term, 'consent manager,' which is described as a data fiduciary from which a data principal may, through an open platform, offer, withdraw, review and manage his/her consent.
• **Hacking and Data Theft**: Sections 43 and 66 of the IT Act prohibit a range of activities, ranging from hacking into a computer network, data theft, the introduction and dissemination of viruses via computer networks, the malicious
use of computers or computer networks or computer programmes, the destruction of any computer or computer system or network, the denial of access to a
computer or computer by an authorised individual.
• **Personal identity theft**: Section 66C of the IT Act provides for punishment for identity theft and provides that any individual who fraudulently or dishonestly makes use of any other person's electronic signature, password or any other unique identification feature shall be punished by imprisonment of either description for a period of up to three (three) years and fine upto Rs one lakh.
India currently not a signatory of any international treaty related to cyber crime

The Convention on Cybercrime, also known as the Budapest Convention on Cybercrime or the Budapest Convention, is the first international treaty seeking to address Internet and computer crime (cybercrime) by harmonizing national laws, improving investigative techniques, and increasing cooperation among nations. It was drawn up by the Council of Europe in Strasbourg, France, with the active participation of the Council of Europe's observer states Canada, Japan, Philippines, South Africa and the United States.

The Convention and its Explanatory Report was adopted by the Committee of Ministers of the Council of Europe at its 109th Session on 8 November 2001. It was opened for

signature in Budapest, on 23 November 2001 and it entered into force on 1 July 2004.As of December 2020, 65 states have ratified the convention, while a further four states had signed the convention but not ratified it.

Since it entered into force, important countries like Brazil and India have declined to adopt the Convention on the grounds that they did not participate in its drafting. Russia opposes the Convention, stating that adoption would violate Russian sovereignty, and has usually refused to cooperate in law enforcement investigations relating to cybercrime. It is the first multilateral legally binding instrument to regulate cybercrime. Their are two more international treaties and some UN treaties which are of not much relevance to us for handling Stuxnet like attack.

Chapter 4

The Solution

Solution Architecture

Sometimes offense is not the best defense and the answer for defense lies in the word itself . Throughout our earlier discussions we have seen the problem with cyber offensives the biggest one being collateral damage and then the legal issues such as treaties , jurisdictions and the socio-ethical issues that come along with it. Hence here we go into detail on how to stop such attacks from happening and we will see some prototypes in the next part of this chapter.

First of all lets demystify the term solution architect

Cybersecurity architect roles and responsibilities include:

Gaining a total understanding of the organization's technology and information systems

- Planning, researching, and designing reliable, powerful, and flexible security architectures for all IT projects
- Performing vulnerability testing on the completed infrastructure, including risk analyses and security assessments
- Researching the latest security standards, new security systems, and updated authentication protocols
- Defining, creating, implementing, and maintaining all needed corporate security policies and procedures, making sure that all employees abide by them ● Developing requirements for all IT assets including routers, firewalls, local area networks (LANs), wide-area networks (WANs), virtual private networks (VPNs), and any other related network devices
- Reviewing and approving the installation of all firewalls, VPN, routers, servers, and IDS scanning technologies
- Preparing cost estimates for all cybersecurity measures and identifying any potential integration issues
- Designing critical public infrastructures (PKIs), including digital signatures and certification authorities (CA)
- Testing the organization's final security structures to make sure they function as planned
- Providing technical guidance and supervision for security teams
- Taking charge of any security awareness programs and educational efforts to better prepare non-IT personnel
- Responding immediately to any security-related incidents (e.g., data breaches, viruses, phishing scams) and providing a complete post-event analysis once there is a resolution
- Updating and upgrading the organization's security systems as needed

Let's breaks the situation down into two distinct phases: Prevention and Reaction. The first set of countermeasures should be preventative in nature, and designed to minimize the likelihood that a control system could be infected by such an attack. The second, and equally important, set of countermeasures should be reactive in nature, and designed to minimize any negative consequences to the control system should the system be compromised. Each of these sets of countermeasures should also possess both passive and active components that utilize direct and indirect methods in responding to the event. These countermeasures are then implemented in real-time based on the impact of the attack and the duration of the attack (which correlates into the likelihood of greater damage or negative consequences).

1. Active Prevention:

- Good and effective cybersecurity policies and framework which are updated at regular intervals
- Policies should be created that address specific host-to-host and zone-to-zone communication requirements, including protocols, ports, etc. This information is vital and will be used in subsequent countermeasures to identify suspect traffic, and is a basic requirement in complying with ISA-99 standards.(The ISA99 standards development committee brings together industrial cyber security experts from across the globe to develop ISA standards on industrial automation and control systems security)
- USBs should be disabled in secure zones wthin the organisation. ● Awareness regarding Cyber security especially regarding social engineering attacks should be created
- Implementation of Software Restriction Policies (SRP) that prevent the execution of code on remote and removable media such as USB, CDs,DVDs.
- Adhere to the GoI guidelines as much as possible
- Active vulnerability scanners on these systems to evaluate and document the configuration against known vulnerabilities and predetermine compliance guidelines. The fact that Stuxnet is using MS08-067 shows that generally vendors may not even be aware of the power of exploiting this vulnerability, Hence regular penetration testing becomes a very important part of organisation these days.
- Implement a comprehensive Patch Management Program that regularly updates operating systems and installed applications in accordance with vendor guidelines and approval of hotfixes, patches, and relevant security updates. Systems should be audited to ensure that updates are installed on a regular basis.

Active prevention:

- Firewalls and Incident response sysytems must be made mandatory to all systems within the organisation with regular updation of firewall iptables.Firewall rules should be implemented that "deny by default" all outbound traffic from the control system networks and zones.

"*Some tests have shown that certain activities of Stuxnet would have triggered HIDS alerts, including DLL injection and rootkit installation attempts.*" ● Authentication methods must be made secure by using physical evidences and multiple staged.

- Utilize code signing of all critical systems (in addition to whitelisting). Updates and changes should go through a unique traceable process at which code should be compared to an out-of-band provided signature from the vendor. Unless verified, code should not be allowed to the system.

Passive reactive:

- Security Information and Event Management (SIEM) systems should be installed to automatically analyze and correlate the data that is generated and stored in system logs and event journals throughout the control system network.
- Implement "Extrusion Detection" which when configured and deployed properly, will generate alerts and or alarms for potential client side attacks.
- Rules needs to be create that look at "What is LEAVING the system: Who, What, When, Where and How."
- Implement a SCADA honeypot or identical system on the same network which can be periodically scanned and tested using more "aggressive" methods, and will not impact normal operation or production.(A honeypot sensor are a "fake" system, designed to let unsuspecting hackers trying to breach the information security, while recording everything to research the tactics, techniques, and procedures (TTP) a attacker would use to take control over critical infrastructure.)
- Set up test and validation systems that mimic the production systems (at least for all the critical components), and implement a recurring comparison process between the production and test systems. Financial institutions have used this approach for several years, and should be considered for critical infrastructure protection as well.

Reactive active:

- Once the attack has been confirmed, all non-essential communication conduits should be filtered and closely monitoring to contain the attack while not negatively impacting plant operation.
- When an attack is identified, operational staff or automated procedures should use the knowledge of the attack to possibly isolate affected hosts, segments or networks until they can be trusted and placed back into service.
- Switches, routers, firewalls and the hosts themselves can be used to isolate an attack in progress.

Prototype

Botnet detection plays a huge role in averting a cyber attack
One is shown below

```
import os
import threading
import concurrent.futures
```



```
import magic
import pandas as pd
from collections import Counter
from scipy import stats
from math import log2
import pyshark
import nest_asyncio
from tqdm import tqdm
import sys
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.metrics import accuracy_score
import numpy as np
from tqdm import tqdm
import pickle



FLOW = {}

def make_csv(filepath, savepath):
  """
  Extracts the featues and generates a csv by invoking tshark
  Args:
    filepath: File path of pcap data to filter
    savepath: Path of directory where to save file
  Returns:
    None: generates a filename.csv
  """
  command = "tshark -r {0} -T fields -e ip.src -e ip.dst -e _ws.col.Protocol -e ip.len -e frame.time_relative -e
frame.time_delta -e tcp.srcport -e tcp.dstport -e udp.srcport -e udp.dstport -E separator=, -E header=y > {1}"
base_name = os.path.basename(filepath)
  save_file_name = os.path.splitext(base_name)[0]+'.csv'
  save_file = os.path.join( savepath, save_file_name )
  os.system(command.format(filepath, save_file))
  print("EXTRACTED features from {}".format(filepath))

def get_pcaps(base_path):
  """
  returns a list of paths to pcap files from the base file

  Args:
    base_path: path to be searched for pcap files

  Returns:
    pcap_list: list of relative paths from the base_path to pcap files
  """
  pcap_list = []
  for path, dir, files in os.walk(base_path):
    for file in files:
```

```
    file_name = os.path.join(path, file)
    magic_mime = magic.from_file(file_name, mime=True)
    if magic_mime == 'application/vnd.tcpdump.pcap' or magic_mime == 'application/octet-stream':
      # vnd.tcpdump.pcap for pcap and octet-stream for pcapng
```



```
      pcap_list.append(file_name)
  return pcap_list


def byte_entropy(labels):
  ent = stats.entropy(list(Counter(labels).values()), base=2)
  if len(labels)<256:
    return ent*8/log2(len(labels))
  else:
    return ent


class HostInfo:
  """
  Class to contain the desired features of a particular host
  """
  def __init__(self, ipv4_address):
    """
    Initialize the host with its IPv4
    """
    self.ip = ipv4_address # string (can be converted to 32 bit int if needed)
    self.src_ports = set() #list of host ports i.e the ports where it serves as source
    self.dst_ports = set() # list of dest ports i.e the ports where it serves as destination
    self.ip_recieved_from = set()
    self.ip_sent_to = set()
    self.protocols = set() # protocols used by the host
    self.total_data_sent = 0
    self.total_data_recv = 0
    self.total_payload = 0 # payload only in data
    self.num_udp_packets_recv = 0
    self.num_udp_packets_sent = 0
    self.num_tcp_packets_recv = 0
    self.num_tcp_packets_sent = 0
    self.total_packets = 0

  def ipv4_to_int(ip_addr):
    """
    takes an ipv4 address and converts it to 32 bit int
    """
    vals = map(int, ip_addr.split('.'))
    return sum(x*256**y for x,y in zip(vals,(3,2,1,0)))

  def __repr__(self):
    """
    string representation, consisting of ipv4 address
    """
    return self.ip

  def __hash__(self):
```

```
  """
  Making the class hashable to store in dictionary
  """
  return hash(repr(self))

 def process_packet(self, packet):
```



```
  packet_type = packet.transport_layer
  layer_names = list(map(lambda x: x.layer_name,
  packet.layers)) if packet_type:
   if packet.ip.src == self.ip:
    self.protocols.add(packet_type)
    self.ip_sent_to.add(packet.ip.dst)
    self.total_data_sent += packet.length
    self.total_packets += 1
    if packet_type == 'TCP':
     self.src_ports.add(packet.tcp.srcport)
     self.num_tcp_packets_sent += 1
     if 'data' in layer_names:
      self.total_payload += len(packet.tcp.payload)
    elif packet_type == 'UDP':
     self.src_ports.add(packet.udp.srcport)
     self.num_udp_packets_sent += 1
     if 'data' in layer_names:
      self.total_payload += len(packet.data.data)
   elif packet.ip.dst == self.ip:
    self.protocols.add(packet_type)
    self.ip_received_from.add(packet.ip.src)
    self.total_data_recv += packet.length
    self.total_packets += 1
    if packet_type == 'TCP':
     self.dst_ports.add(packet.tcp.dstport)
     self.num_tcp_packets_recv += 1
     if 'data' in layer_names:
      self.total_payload += len(packet.tcp.payload)
    elif packet_type == 'UDP':
     self.dst_ports.add(packet.udp.dstport)
     self.num_udp_packets_recv += 1
     if 'data' in layer_names:
      self.total_payload += len(packet.data.data)
  elif 'arp' in layer_names:
   if packet.arp.src_proto_ipv4 == self.ip:
    self.protocols.add('ARP')
    self.ip_sent_to.add(packet.arp.dst_proto_ipv4)
    self.total_data_sent += packet.length
    self.total_packets += 1
   elif packet.arp.dst_proto_ipv4 == self.ip:
    self.protocols.add('ARP')
    self.ip_received_from.add(packet.arp.src_proto_ipv4
    ) self.total_data_recv += packet.length
    self.total_packets += 1
  elif 'icmp' in layer_names:
   if packet.ip.src == self.ip:
    self.protocols.add('ICMP')
```

```
        self.ip_sent_to.add(packet.ip.dst)
        self.total_data_sent += packet.length
        self.src_ports.add(packet.udp.srcport)
        self.total_packets += 1
      elif packet.ip.dst == self.ip:
        self.protocols.add('ICMP')
        self.ip_received_from.add(packet.ip.src)
        self.total_data_recv += packet.length
```



```
        self.dst_ports.add(packet.tcp.dstport)
        self.total_packets += 1


def min_none(a,b):
  """
  Min(a,b), returns the other element if either of `a` or `b` is None
  Args:
     a: int or float value
     b: int or float value
  Returns:
     minimum of a or b
  """
  if not a:
    return b
  if not b:
    return a
  return min(a,b)

def max_none(a,b):
  """
  Min(a,b), returns the other element if either of `a` or `b` is None
  Args:
     a: int or float value
     b: int or float value
  Returns:
     minimum of a or b
  """

  if not a:
    return b
  if not b:
    return a
  return max(a,b)

# key src ip, src port, dst ip, dst port, protocol
class Flow:
  """
  Class to represent flow information (5-tuple)
  """
  def __init__(self, src_ip, src_port, dst_ip, dst_port,
    protocol): """
    initialize the source, destination host, ports and protocol
    """
    self.src_ip = src_ip #ip of source of flow
```

```
    self.src_port = src_port # port used by source of flow
    self.dst_ip = dst_ip
    self.dst_port = dst_port
    self.protocol = protocol

    self.total_data = 0
    self.sent_packets = 0
    self.recv_packets = 0
    self.sent_data = 0
    self.recv_data = 0
```



```
    self.num_small_packets = 0

    self.total_sent_payload = 0
    self.total_recv_payload = 0
    self.max_payload_size = 0
    self.max_payload_entropy = 0
    self.min_payload_size = 0
    self.min_payload_entropy = 0
    self.highest_protocols = set()
    self.last_timestamp_sent = None
    self.start_timestamp_sent = None
    self.last_timestamp_recv = None
    self.start_timestamp_recv = None
    #post processing data
    self.total_time = None
    self.all_payload = b''
    self.net_entropy = 0
    self.average_payload_size = 0
    self.average_packet_size_per_sec = 0
    self.average_packet_per_sec = 0
    self.average_packet_length = 0
    self.incoming_outgoing_ratio = 0
    self.label = 0


  def __repr__(self):
    return "{0},{1},{2},{3},{4}".format(self.src_ip,self.src_port,self.dst_ip,self.dst_port,self.protocol)

  def __hash__(self):
    return hash(repr(self))

  def post_processing(self):
    self.total_time = max_none(self.last_timestamp_recv, self.last_timestamp_sent) -
min_none(self.start_timestamp_recv, self.start_timestamp_sent)
    if self.all_payload:
      self.net_entropy = byte_entropy(self.all_payload)
    if self.total_time:
      self.average_packet_size_per_sec = self.sent_data/self.total_time
      self.average_packet_per_sec = self.sent_packets/self.total_time
    if self.sent_packets:
      self.average_payload_size = self.total_sent_payload/(self.sent_packets)
    self.average_packet_length = self.total_data/(self.sent_packets+self.recv_packets)
    if self.sent_data !=0:
      self.incoming_outgoing_ratio = self.recv_data/self.sent_data
```

```
    else:
      self.incoming_outgoing_ratio = self.recv_data


  def to_csv(self):
    return
"{0},{1},{2},{3},{4},{5},{6},{7},{8},{9},{10},{11},{12},{13},{14},{15},{16},{17},{18},{19},{20},{21},{22},{23},{24},{2
5}\n".format(
      self.src_ip,
      self.src_port,
      self.dst_ip,
```



```
      self.dst_port,
      self.protocol,
      self.total_data,
      self.sent_packets,
      self.recv_packets,
      self.sent_data,
      self.recv_data,
      self.total_sent_payload,
      self.total_recv_payload,
      self.max_payload_size,
      self.max_payload_entropy,
      self.min_payload_size,
      self.min_payload_entropy,
      self.net_entropy,
      self.average_payload_size,
      self.average_packet_length,
      self.average_packet_per_sec,
      self.average_packet_size_per_sec,
      len(self.highest_protocols),
      self.total_time,
      self.incoming_outgoing_ratio,
      self.num_small_packets,
      self.label
      )

def process_payload(data, is_hex=True):
  """
  returns size and normalized entropy for the hex data
  """
  if is_hex:
    payload_bytes =
  bytes.fromhex("".join(data.split(":"))) else:
    payload_bytes = data.encode()
  payload_size = len(payload_bytes)
  payload_entropy = byte_entropy(payload_bytes)
  return payload_size, payload_entropy

def process_packet(packet):
  """
  Collects info and fills in the respective class from the
  packet Args:
    packet: pyshark packet
```

```
  """
  highest_layer = packet.highest_layer
  packet_type = packet.transport_layer
  layer_names = list(map(lambda x: x.layer_name, packet.layers))
  src_ip = None
  dst_ip = None
  src_port = -1
  dst_port = -1
  payload_size = 0
  payload_entropy = 0
  timestamp = float(packet.sniff_timestamp)
  packet_size = int(packet.length)
  small_packet = int(packet_size < 100)
```



```
  if packet_type: ##contains an IP layer
    src_ip = packet.ip.src
    dst_ip = packet.ip.dst
    if packet_type == 'TCP':
      src_port = packet.tcp.srcport
      dst_port = packet.tcp.dstport

      if 'data' in layer_names:
        payload_size, payload_entropy = process_payload(packet.tcp.payload)

    elif packet_type == 'UDP':
      src_port = packet.udp.srcport
      dst_port = packet.udp.dstport
      if 'data' in layer_names:
        payload_size, payload_entropy = process_payload(packet.data.data)
  elif 'icmp' in layer_names:
    try:
      payload_size, payload_entropy = process_payload(packet.icmp.data)
    except AttributeError:
      payload_size, payload_entropy = 0,0
    packet_type = 'ICMP'
  elif 'arp' in layer_names:
    dst_ip = packet.arp.dst_proto_ipv4
    src_ip = packet.arp.src_proto_ipv4
    packet_type = 'ARP'
  if 'dns' in layer_names:
    payload_size, payload_entropy = process_payload(packet.dns.qry_name,False)
  return src_ip, src_port, dst_ip, dst_port, packet_type, timestamp, packet_size ,highest_layer,
payload_entropy, payload_size, small_packet

def fill_flow(packet,label):
  src_ip, src_port, dst_ip, dst_port, packet_type, timestamp, packet_size ,highest_layer,
payload_entropy, payload_size, small_packet = process_packet(packet)
  flow_key = (src_ip, src_port, dst_ip, dst_port, packet_type)
  flow_key_rev = (dst_ip, dst_port, src_ip, src_port, packet_type)

  flow = FLOW.get(flow_key, Flow(*flow_key))
  flow.total_data += packet_size
  flow.sent_data += packet_size
  flow.max_payload_size = max(payload_size, flow.max_payload_size)
```

```
  flow.max_payload_entropy = max(payload_entropy, flow.max_payload_entropy)
  flow.min_payload_size = min(payload_size, flow.max_payload_size)
  flow.min_payload_entropy = min(payload_entropy, flow.max_payload_entropy)
  flow.total_sent_payload += payload_size
  flow.sent_packets += 1
  flow.num_small_packets+=small_packet
  flow.highest_protocols.add(highest_layer)
  flow.label = label
  if not flow.start_timestamp_sent:
    flow.start_timestamp_sent = timestamp
  flow.last_timestamp_sent = timestamp
  FLOW[flow_key] = flow

  flow_rev = FLOW.get(flow_key_rev, Flow(*flow_key_rev))
  flow_rev.total_data += packet_size
```



```
  flow_rev.recv_data += packet_size
  flow_rev.total_recv_payload += payload_size
  flow_rev.recv_packets += 1
  flow_rev.highest_protocols.add(highest_layer)
  if not flow_rev.start_timestamp_recv:
    flow_rev.start_timestamp_recv = timestamp
  flow_rev.last_timestamp_recv = timestamp
  flow_rev.label = label
  FLOW[flow_key_rev] = flow_rev


def get_num_packets(path):
  command = "tshark -r {} | wc -l"
  data = os.popen(command.format(path)).read()
  return int(data.strip())

def packet_types(path):
  nest_asyncio.apply()
  capture_dump = pyshark.FileCapture(path)
  capture_dump.keep_packets = False ##very memory consuming, very
  important packet_types = {}
  packet_list = []
  count = 0
  while True:
    try:
      packet = capture_dump.next()
      if packet.highest_layer not in packet_types:
        packet_list.append(packet)
      packet_types[packet.highest_layer] = packet_types.get(packet.highest_layer,0) + 1
      count +=1
      if count == 30000:
        break
    except StopIteration:
      break
  for packet in packet_list:
    print(packet.layers, packet_types[packet.highest_layer])
  return packet_list
```

```
def get_ips(path):
  nest_asyncio.apply()
  capture_dump = pyshark.FileCapture(path)
  capture_dump.keep_packets = False ##very memory consuming, very
  important ips = {}
  count = 0
  while True:
    try:
      packet = capture_dump.next()
      if packet.transport_layer:
        src_ip = packet.ip.src
        dst_ip = packet.ip.dst
        ips[src_ip] = ips.get(src_ip,0)+1
      count += 1
      if count == 50000:
        break
    except StopIteration:
```



```
      break
  sorted_dict = sorted(ips.items(), key = lambda x: x[1], reverse=True)
  for a,b in sorted_dict:
    print(a,b)


def filter_data(pcap_path, ip_list, csv_path, label=0):
  num_packets = get_num_packets(pcap_path)
  nest_asyncio.apply()
  capture_dump = pyshark.FileCapture(pcap_path)
  print("Number of packets found: {}".format(num_packets))
  capture_dump.keep_packets = False ##very memory consuming, very important
  for i in tqdm(range(num_packets), desc = "processing pcap {}".format(pcap_path),
    ascii=False): try:
      packet = capture_dump.next()
      fill_flow(packet,label)
    except Exception as e:
      print(e)
  with open(csv_path,'w') as output_csv:
    header = "src_ip,src_port,dst_ip,dst_port,protocol,total_data,
sent_packets,recv_packets,sent_data,recv_data,total_sent_payload,total_recv_payload,max_payload_size,max_
payload_entropy,min_payload_size,min_payload_entropy,net_entropy,average_payload_size,average_packet_l
ength,average_packet_per_sec,average_packet_size_per_sec,num_protocols,total_time,incoming_outgoing_rati
o,num_small_packets,label\n"
    output_csv.write(header)
    for key in tqdm(FLOW.keys(), desc = "saving to {}".format(csv_path), ascii=False
      ): m = FLOW[key]
      m.post_processing()
      output_csv.write(m.to_csv())

def clean_dataset(df):
    assert isinstance(df, pd.DataFrame), "df needs to be a pd.DataFrame"
    df.fillna(0,inplace=True)
    indices_to_keep = ~df.isin([np.nan, np.inf, -np.inf]).any(1)
    return df[indices_to_keep].astype(np.float64)
```

```
def format_flow(flow):
  return "{}:{} -> {}:{} ; {}\n".format(*flow)

def clean(df, pred,indices, output_path):
  h1 = Counter([i[0] for i in pred])
  h2 = Counter([i[2] for i in pred])
  n = len(pred)
  N = len(df)
  output_flows = []
  print(n,N,n/N)
  thresh = n/200
  output_file = open(output_path, "w")
  host = h1.most_common(1)[0][0]
  botnets = set( i for i in h1 if h1[i] > thresh ).intersection( i for i in h2 if h2[i] > thresh )
  if botnets:
    botnets.remove(host)
  for predic in pred:
```



```
    if predic[4]== 'UDP' or predic[4]=='TCP':
      if predic[0] in botnets or predic[2] in botnets:
        output_flows.append(predic)

  lb = max(len(botnets),1)
  if n/N < 0.2 or len(output_flows)/lb < 10:
    output_file.write("No Botnets detected\n")
  else:
    output_file.write("----------Detected Botnet Hosts----------\n")
    for botnet in botnets:
      output_file.write(botnet+"\n")
    output_file.write(host)
    output_file.write("----------Malicious Flows----------\n")
    for flow in output_flows:
      output_file.write(format_flow(flow))
  output_file.close()
  print("output written to {}".format(output_path))


def detection(pcap_path, csv_path):
  filter_data(pcap_path,[],csv_path)
  df = pd.read_csv(csv_path)
  features = clean_dataset(df[df.columns[5:-1]])
  model_name = "trained_model.pickle"
  with open(model_name,'rb') as model_file:
    model = pickle.load(model_file)
  flows = df[df.columns[0:5]]
  predictions = model.predict(features)
  indices = []
  botnet_flows = []
  for i in range(len(predictions)):
    if predictions[i] == 1:
      indices.append(i)
      botnet_flows.append(list(flows.iloc[i]))
```

```
  return df, botnet_flows, indices


def main(pcap_path, csv_path,output_path):
  a,b,c = detection(pcap_path, csv_path)
  clean(a,b,c,output_path)




def train(model_name):
  p2pbox1_ip = ["192.168.1.2"]
  p2pbox2_ip = ["192.168.2.2"]
  torrent_ip = ["172.27.28.106"]
  storm_ip =
["66.154.80.101","66.154.80.105","66.154.80.111","66.154.80.125","66.154.83.107","66.154.83.113","66.154.8
3 .138","66.154.83.80","66.154.87.39","66.154.87.41","66.154.87.57","66.154.87.58","66.154.87.61"]
vinchua_ip = ["172.27.22.206"]
  zeus_ip = ["10.0.2.15"]
  p2pbox1_pcaps = get_pcaps("Botnet_Detection_Dataset/Benign/p2pbox1")
  p2pbox2_pcaps = get_pcaps("Botnet_Detection_Dataset/Benign/p2pbox2")
```



```
  torrent_pcaps = get_pcaps("Botnet_Detection_Dataset/Benign/torrent")
  storm_pcaps = get_pcaps("Botnet_Detection_Dataset/Botnet/storm")
  vinchua_pcaps = get_pcaps("Botnet_Detection_Dataset/Botnet/vinchuca")
  zeus_pcaps = get_pcaps("Botnet_Detection_Dataset/Botnet/zeus")

  files_benign = p2pbox1_pcaps+p2pbox2_pcaps+torrent_pcaps
  files_botnet = storm_pcaps+vinchua_pcaps+zeus_pcaps


  if not os.path.exists("filtered_data"):
    os.mkdir("filtered_data")

  for file in files_botnet:
    base_name = os.path.basename(file)
    filter_data(file, [], os.path.join("filtered_data", base_name+".csv"),
    label=1) FLOW.clear()

  for file in files_benign:
    base_name = os.path.basename(file)
    filter_data(file, [], os.path.join("filtered_data", base_name+".csv"),
    label=0) FLOW.clear()

  df_all = None
  for file in tqdm(os.listdir("filtered_data"), ascii=False):
    df = pd.read_csv(os.path.join("filtered_data",file))
    if 'label' not in df.columns:
      print(file)
    if type(df_all) == type(None):
      df_all = df
    else:
      df_all = df_all.append(df, ignore_index = True)
```

```
  with open('training.csv','w') as out_csv:
    out_csv.write(df_all.to_csv(index = False))


  features = clean_dataset(df1[df1.columns[5:-1]])
  flows = df1[df1.columns[0:5]]
  y = df1['label']
  X_train, X_test, y_train, y_test = train_test_split(features, y, test_size=0.2)
  dtc = DecisionTreeClassifier()
  bag=BaggingClassifier(base_estimator=dtc, n_estimators=100, bootstrap=True)
  bag.fit(X_train, y_train) # Fit the model using train data
  print(bag.score(X_test,y_test)) # Get the accuracy of test data
  print(precision_recall_fscore_support(bag.predict(X_test),y_test))
  with open(model_name,'wb') as model_file:
    pickle.dump(bag, model_file)


if __name__ == "__main__":
  USAGE_INFO="""
  detection: usage python3 botnetdetect.py <path to pcap>
  training: usage python3 train <name of model to save> (NOTE: running directory must contain
Botnet_Detection_Dataset )
  output stored in "output.txt"
```



```
  """

  if len(sys.argv)==2:
    if os.path.exists(sys.argv[1]):
      csv_path = "extracted_features.csv"
      output_path = "output.txt"
      main(sys.argv[1],csv_path,output_path)

    else:
      print("file not found")
      print(USAGE_INFO)
      exit(1)

  elif len(sys.argv)==3:
    if sys.argv[1]=="train":
      model_name = sys.argv[2]
      train(model_name)
    else:
      print(USAGE_INFO)
      exit(1)

  else:
    print(USAGE_INFO)
    exit(1)
```

Chapter 5

SUCCESS!

- **What does success look like when trying to prevent a cyber warfare attack on a critical infrastructure?**

  When malicious attacks are getting flagged regularly by the incident response system and firewall are not allowing suspicious addresses to go through. All this factors can be checked through regular penetration testing and moving in accordance with latest cyber laws.

- **What does success look like when under a cyber warfare attack?**

  Again when from previous some sort of patching is helping and their is some sususpicious behaviour you are able to catch out from various flagging system. Moreover the critical objects and files are safe from such attempts.

- **What does success look like when asked to launch an offensive?**

  Offensive is successful if you are able to strike your target efficiently and not getting in the radar .Offensive have many positive and negative impacts as mentioned earlier.

Bibliography

- https://acehacker.com/microsoft/cybersecurity/resources/The-History-of-Stuxnet.pdf ● https://acehacker.com/microsoft/cybersecurity/resources/Protecting_Critical_Infrastructure_Against_the_Next_Stuxnet.pdf
- https://www.cisa.gov/uscert/ics/content/overview-cyber-vulnerabilities
- https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/
- https://en.wikipedia.org/wiki/Stuxnet
- https://www.darktrace.com/en/blog/unintended-consequences-when-cyber-attacks-go-wild/ ● https://www.inss.org.il/wp-content/uploads/2017/02/FILE1333533336-1.pdf ● https://www.justitsministeriet.dk/sites/default/files/media/Arbejdsomraader/Forskning/Forskningspuljen/Legal_Aspects_of_Cybersecurity.pdf
- https://guides.ll.georgetown.edu/c.php?g=363530&p=4821478
- https://www.simplilearn.com/why-become-a-cyber-security-architect-article
- https://scadahacker.com/resources/stuxnet-mitigation.html
- https://www.honeypot.dk/