tripwire
CONFIDENCE: SECURED

Twisted Haystack: Protecting Industrial Systems with Dynamic Deception

Lane Thames, PhD
10/3/2018

Who Am I?

Lane Thames

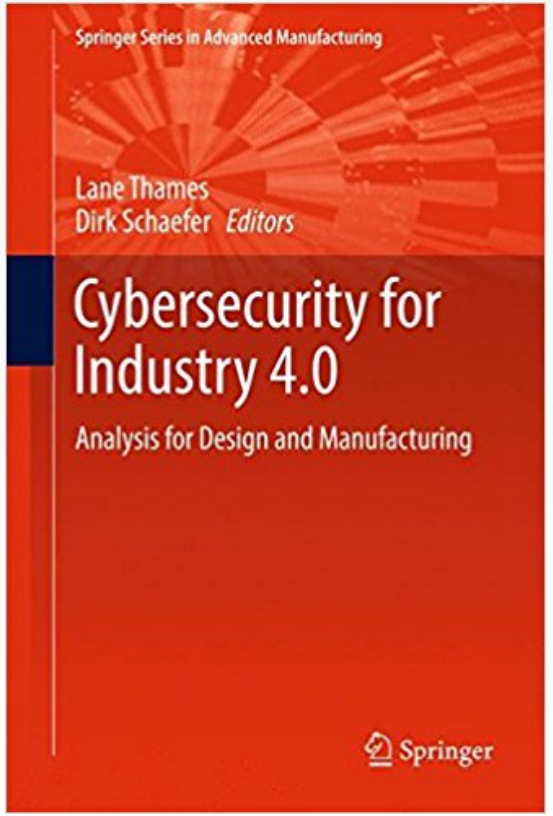
TRIPWIRE® VERT

Vulnerability and Exposure Research Team (VERT)

BELDEN
SENDING ALL THE RIGHT SIGNALS

Background

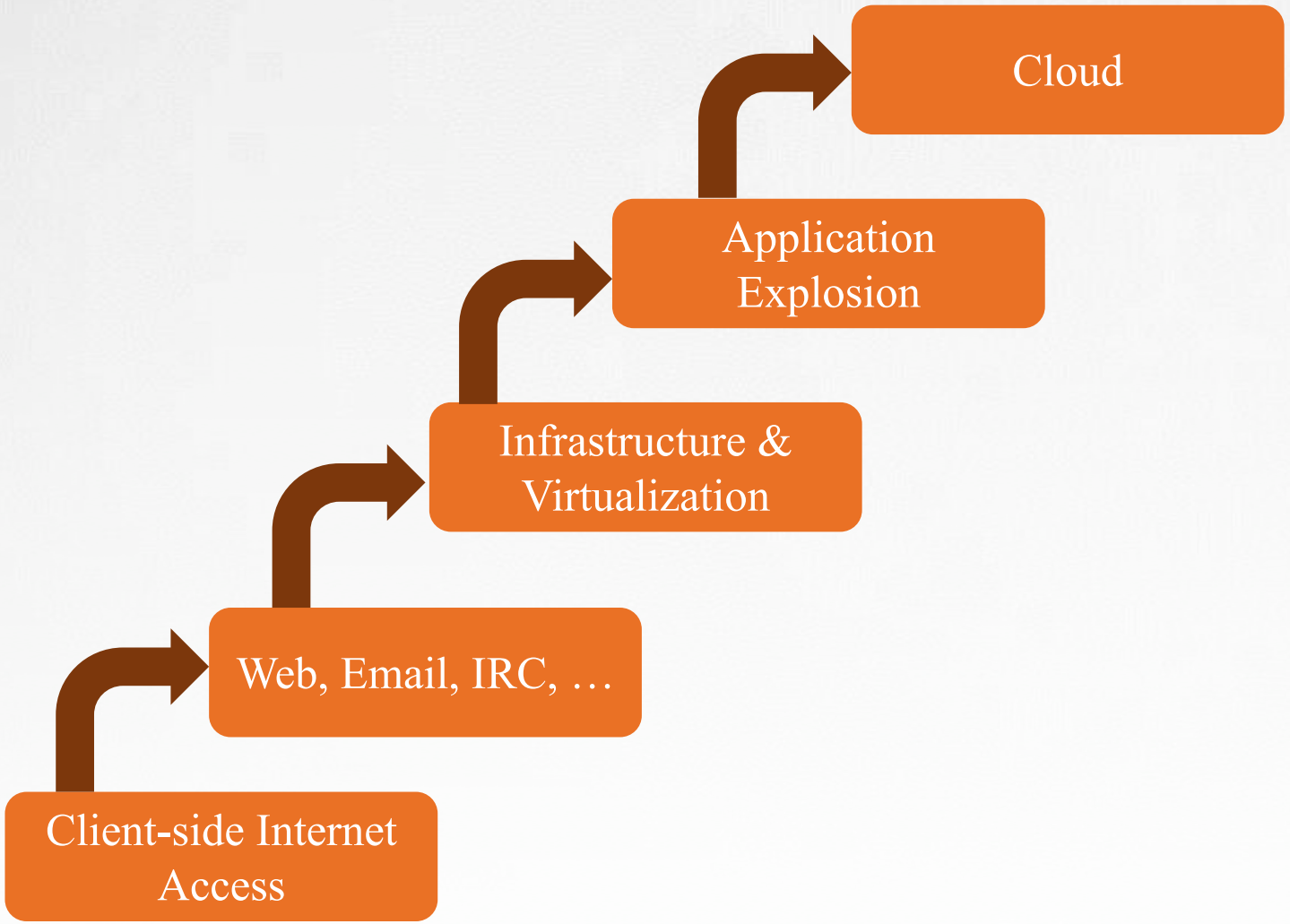
- PhD, Electrical and Computer Engineering (Georgia Tech)
- 16 years of experience
  - Computer Engineering
  - Software Engineering
  - IT
  - Cybersecurity



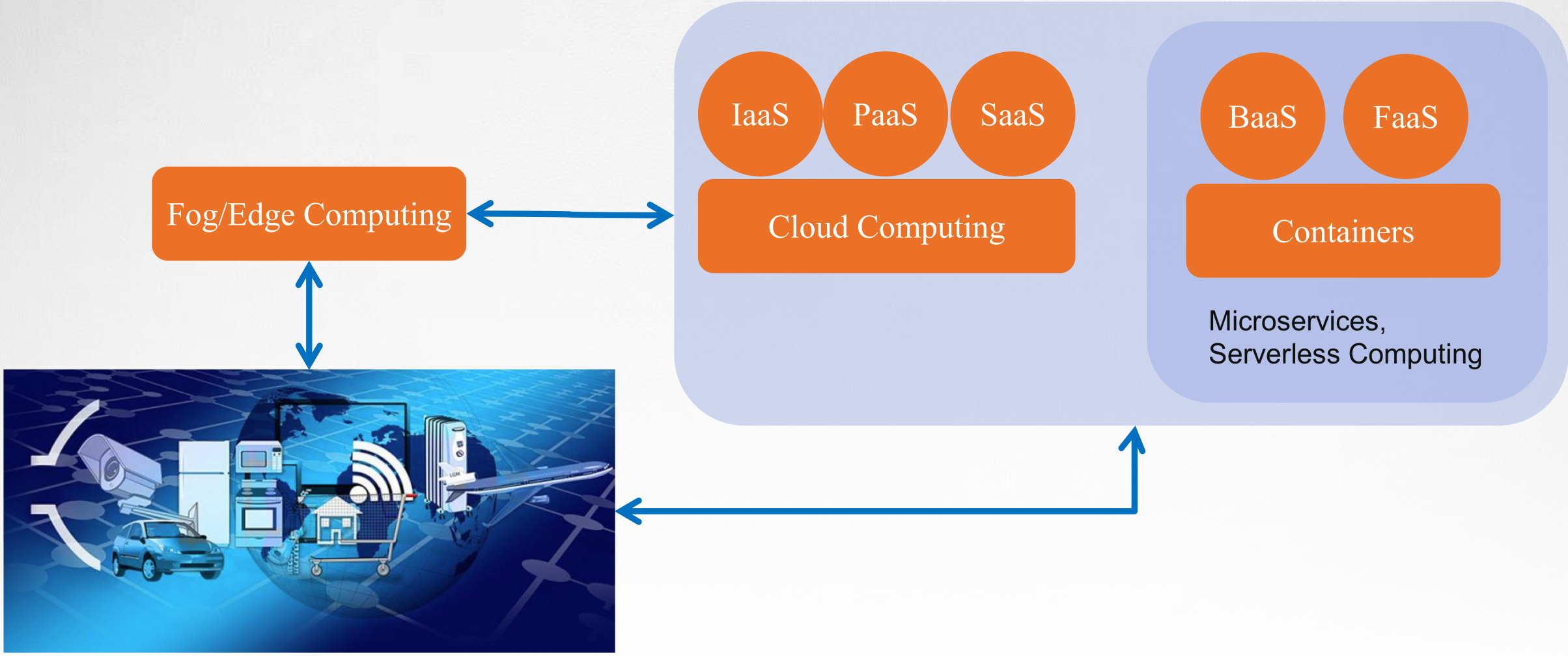
Overview

- Motivating Factors
  - Cloud evolution and the Internet of Things
- Problems &Opportunities with our computing future
  - New solution opportunities
- Deception & Dynamic Deception
- Implementing Dynamic Deception with Python and Twisted
- Scalable Dynamic Deception with Containers
- Conclusions and Future Work

Cloud Evolution



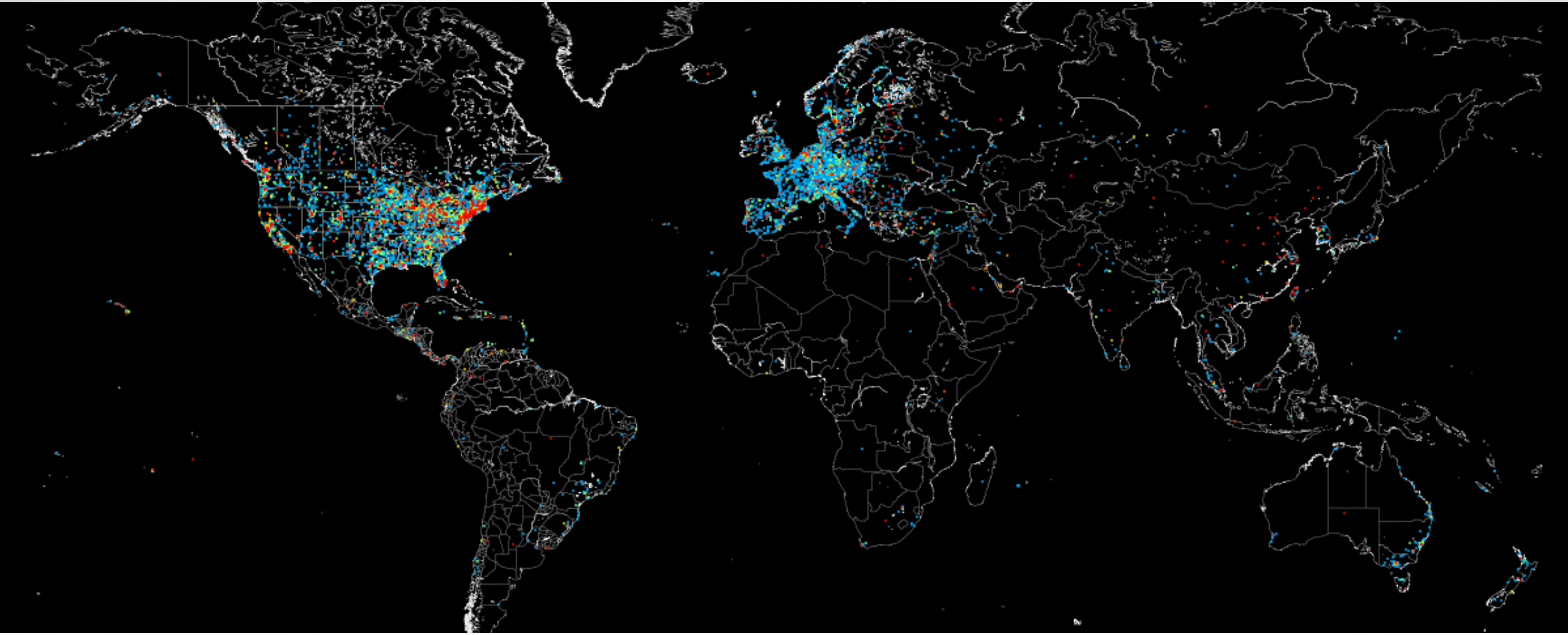
From Cloud Evolution to the Internet of Things



Industrial IoT

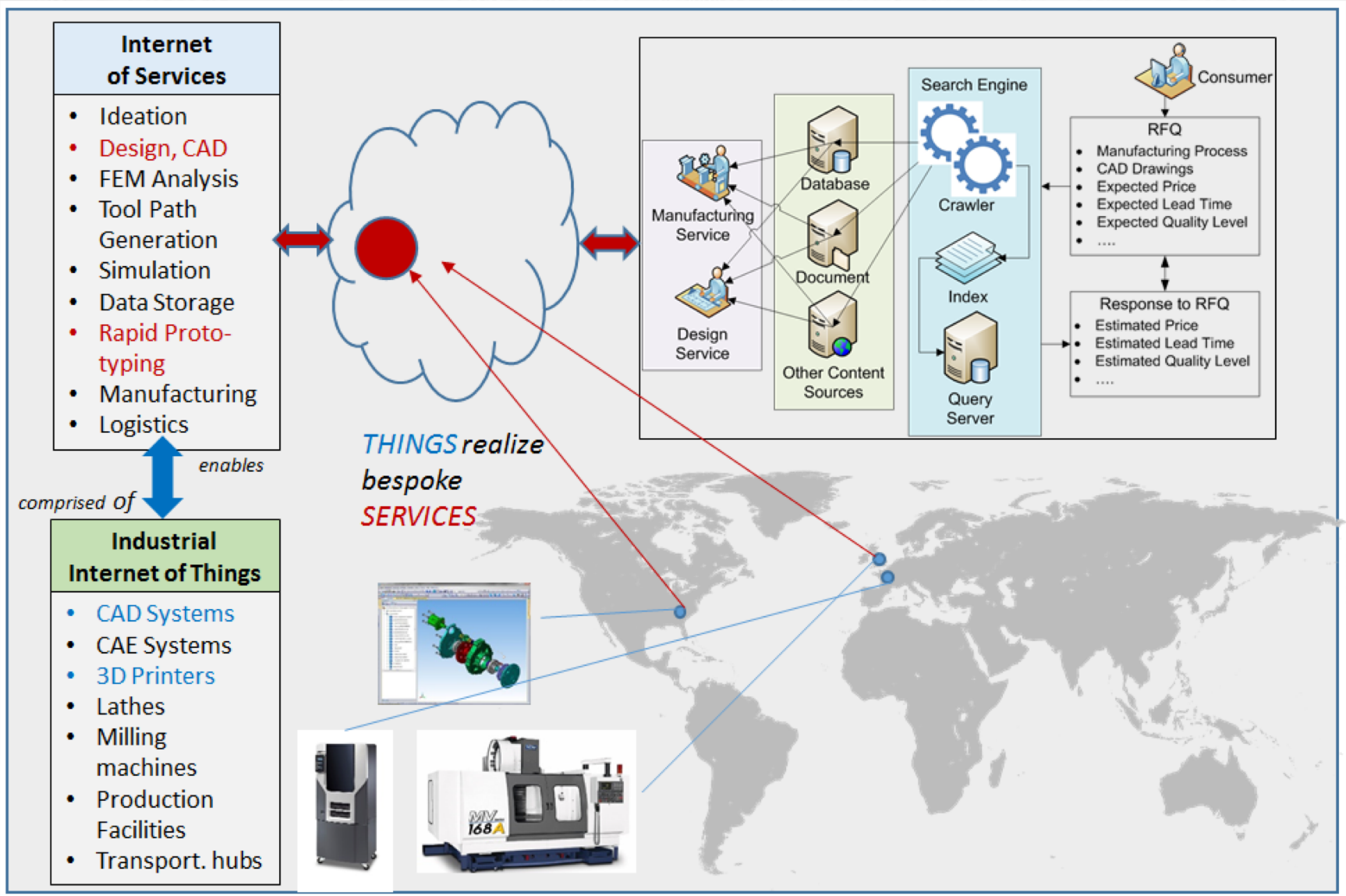
Internet

Internet Connectivity: An Inevitability



Data Source: https://icsmap.shodan.io/

Cloud-based Design & Manufacturing (CBDM)



The Future is Bright

Smart Power Grids

Smart Logistics

Smart Inventory

Smart Machine Diagnostics

Smart . . .

Self-monitoring, Group-monitoring

Self-configuration, Group-configuration

Self-healing, Group-healing

Provides:

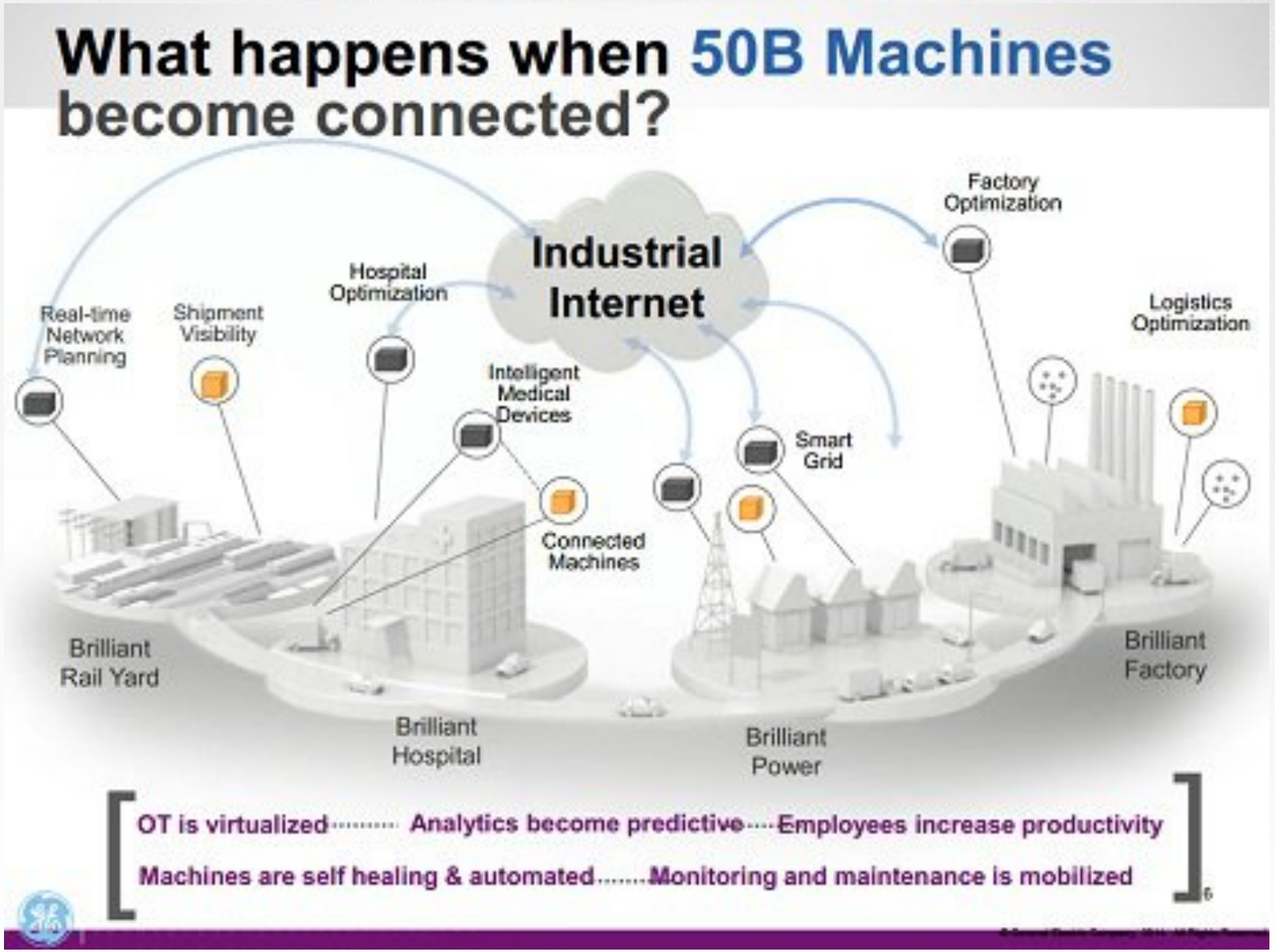
Operational Efficiencies

Outcome-driven Processes

Machine-to-Human Collaboration

Symbiotic Product Realization

Countless Value Creation Opportunities



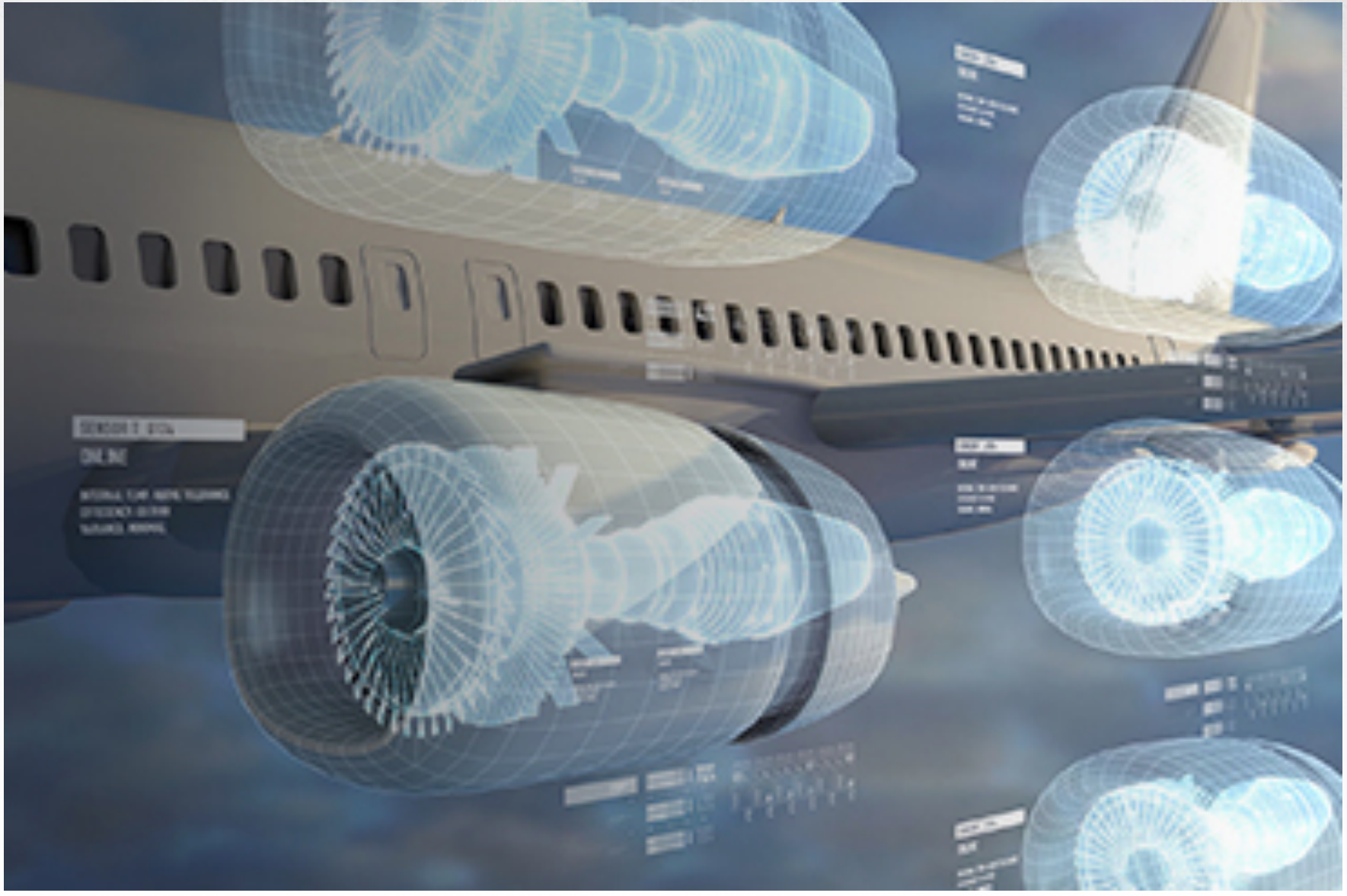
The Future is Bright

Digital Twin

Wikipedia:

Digital twin refers to a digital replica of physical assets, processes and systems that can be used for various purposes. The digital representation provides both the elements and the dynamics of how an Internet of Things device operates and lives throughout its life cycle.

Digital Twins integrate artificial intelligence, machine learning and software analytics with data to create living digital simulation models that update and change as their physical counterparts change.



The Future is Bright

It's an amazing time to be an engineer!

*Richard Feynman – The Character of Physical Law Lecture Series

What will prevent us from achieving the full potential?

Cybersecurity is a fundamental “limit” on our technological advancements!

Our Approximate Solution

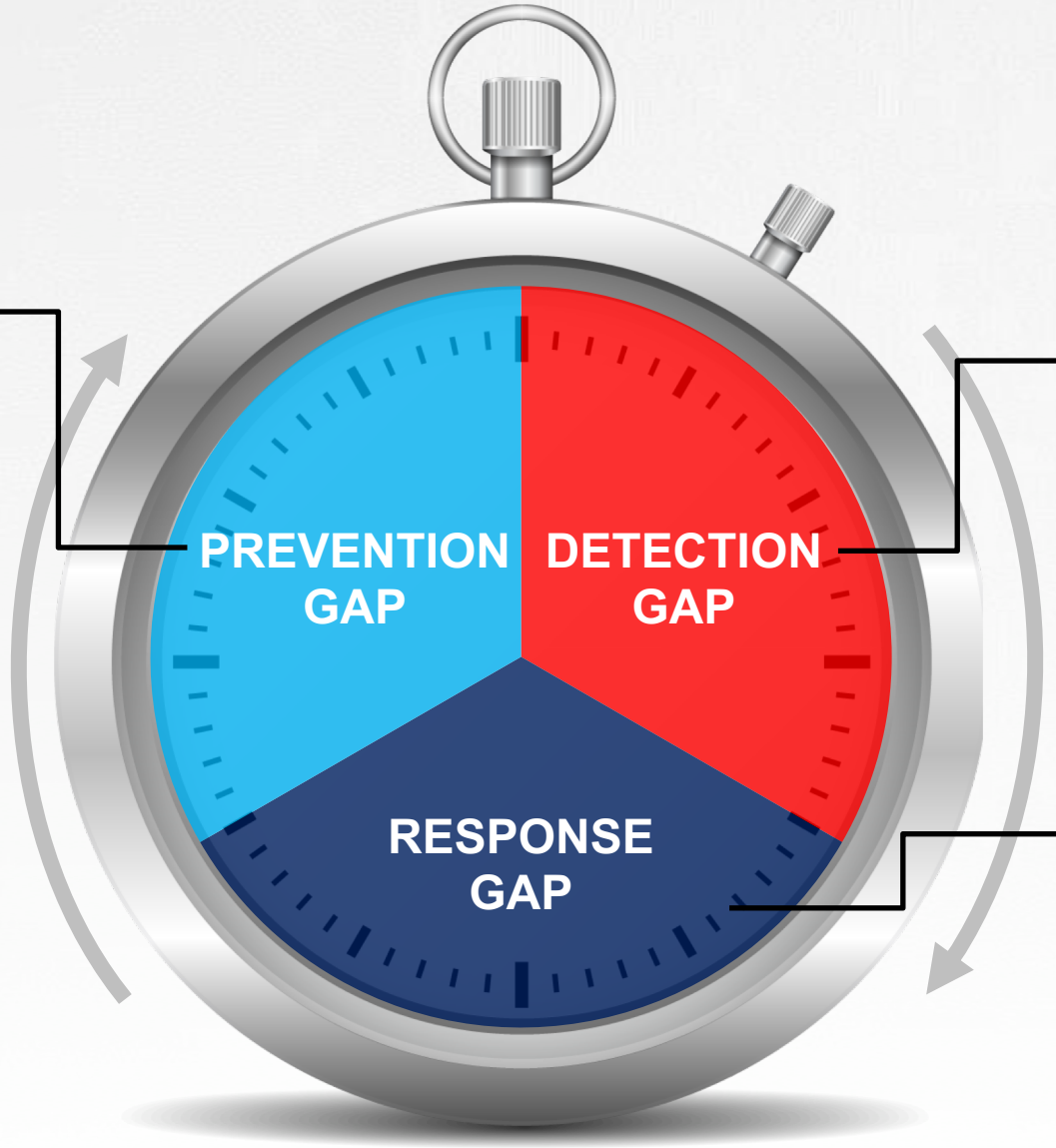
Defense in-depth

Time is always against us – or is it?

Cybersecurity

Prevention Gap
Time to put preventative measures in place to avoid repeated attacks

Can we avoid this from happening again?



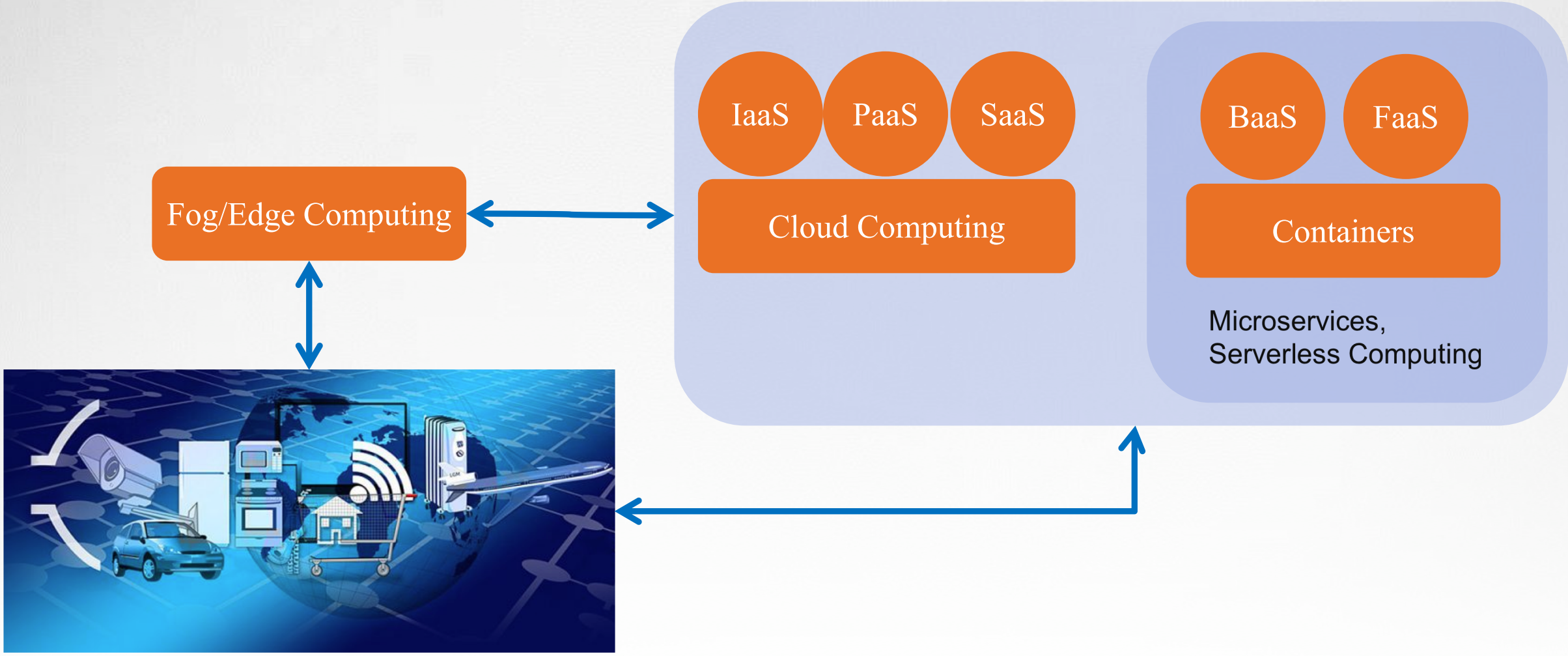
Detection Gap
Time between actual breach and discovery

Have we been breached?

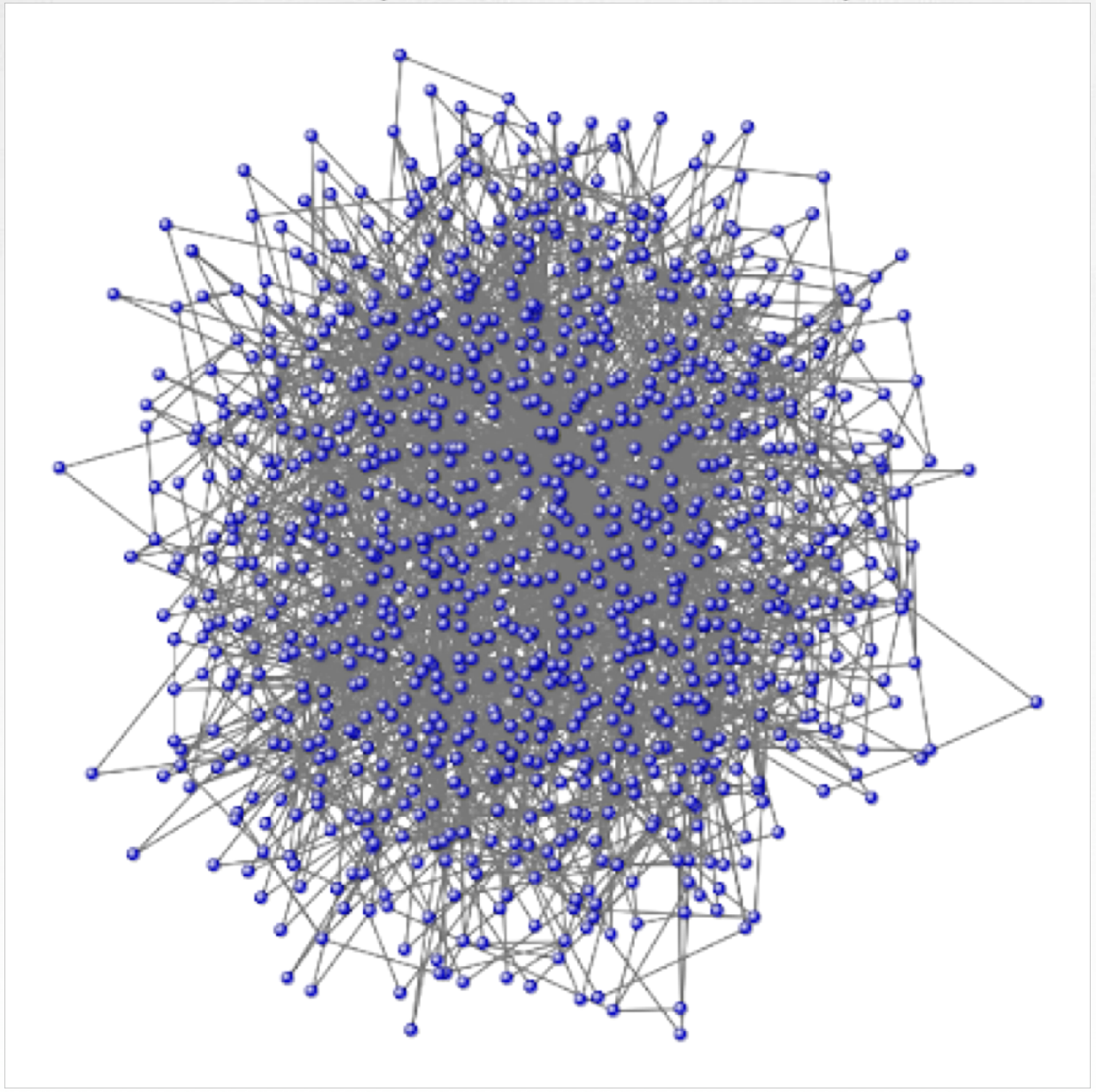
Response Gap
Time between discovery to remediation to limit damage

How bad is it?

From Cloud Evolution to the Internet of Things

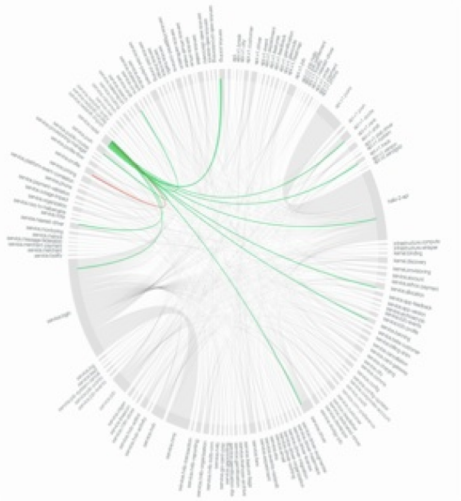


Complexity & Chaos Anyone -- Everyone ??



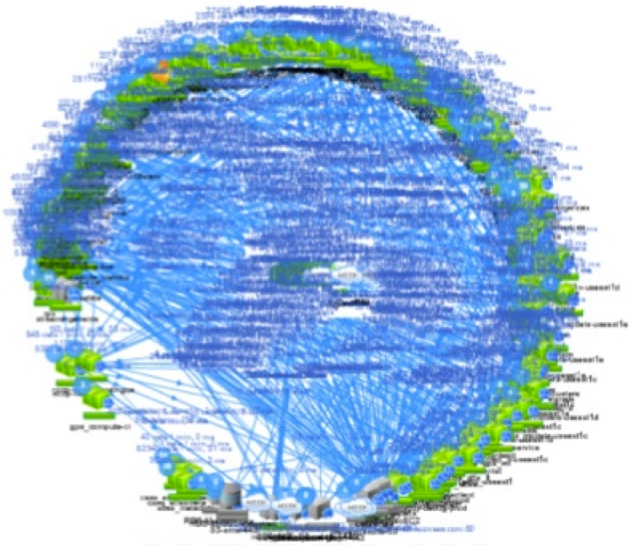
Complexity & Chaos Anyone -- Everyone ??

450 microservices



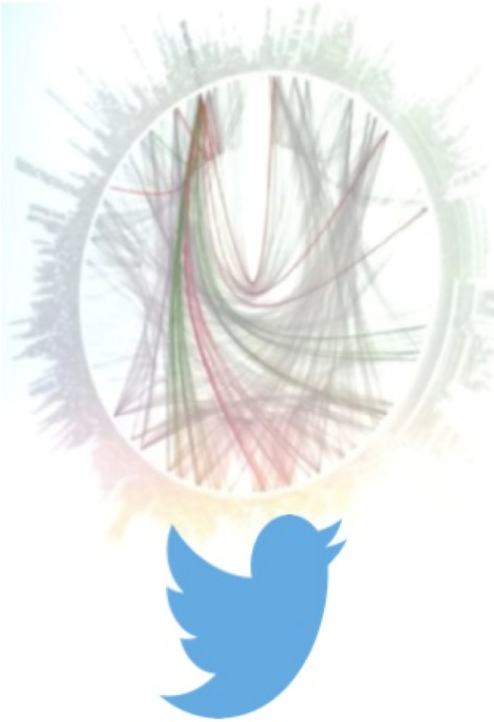
HAIL O

500+ microservices



NETFLIX

500+ microservices



Source:
Netflix: http://www.slideshare.net/BruceWong3/the-case-for-chaos
Twitter: https://twitter.com/adrianco/status/441883572618948608
Hail-o: https://sudo.hailoapp.com/services/2015/03/09/journey-into-a-microservice-world-part-3/

Source: http://contiv.github.io/articles/2016/03/06/scaling-microservices.html

Deception

de·ceive

/dəˈsēv/

verb

(of a person) cause (someone) to believe something that is not true, typically in order to gain some personal advantage.
"I didn't intend to **deceive** people **into** thinking it was French champagne"
synonyms: swindle, defraud, cheat, trick, hoodwink, hoax, dupe, take in, mislead, delude, fool, outwit, lead on, inveigle, beguile, double-cross, gull; More

- (of a thing) give a mistaken impression.
  "the area may seem to offer nothing of interest, but don't be deceived"
- fail to admit to oneself that something is true.
  "enabling the rulers to deceive themselves about the nature of their own rule"

Deception-based Cyberattacks

- Social engineering
- Phishing
- Spam

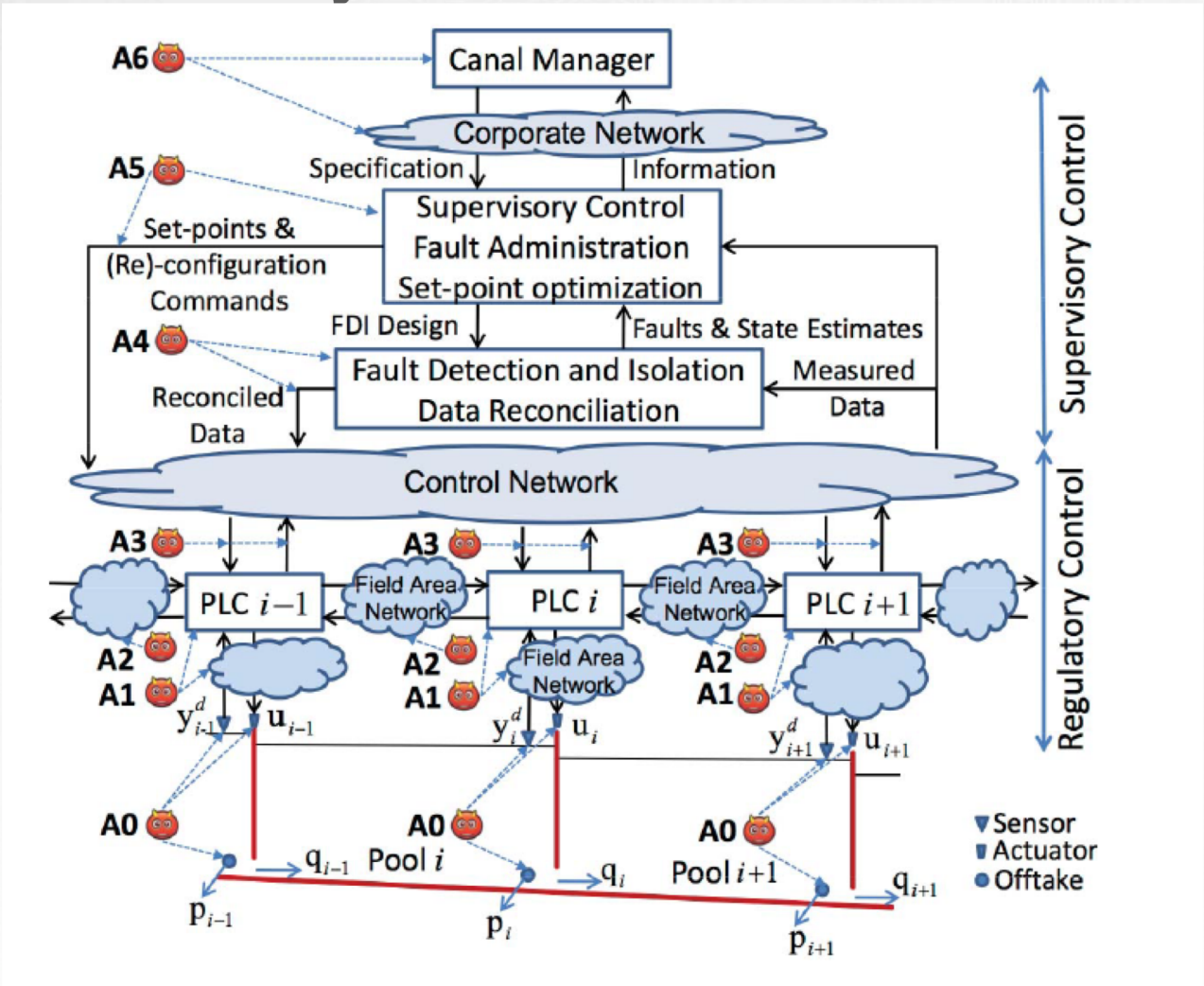
Deception-based Cyberattacks - IIoT

Signal Injection

- Incorrect (spoofed) sensor measurements
- Incorrect (spoofed) control inputs
- Incorrect (spoofed) timestamps
- Incorrect (spoofed) identity information

Deception-based Cyberattacks

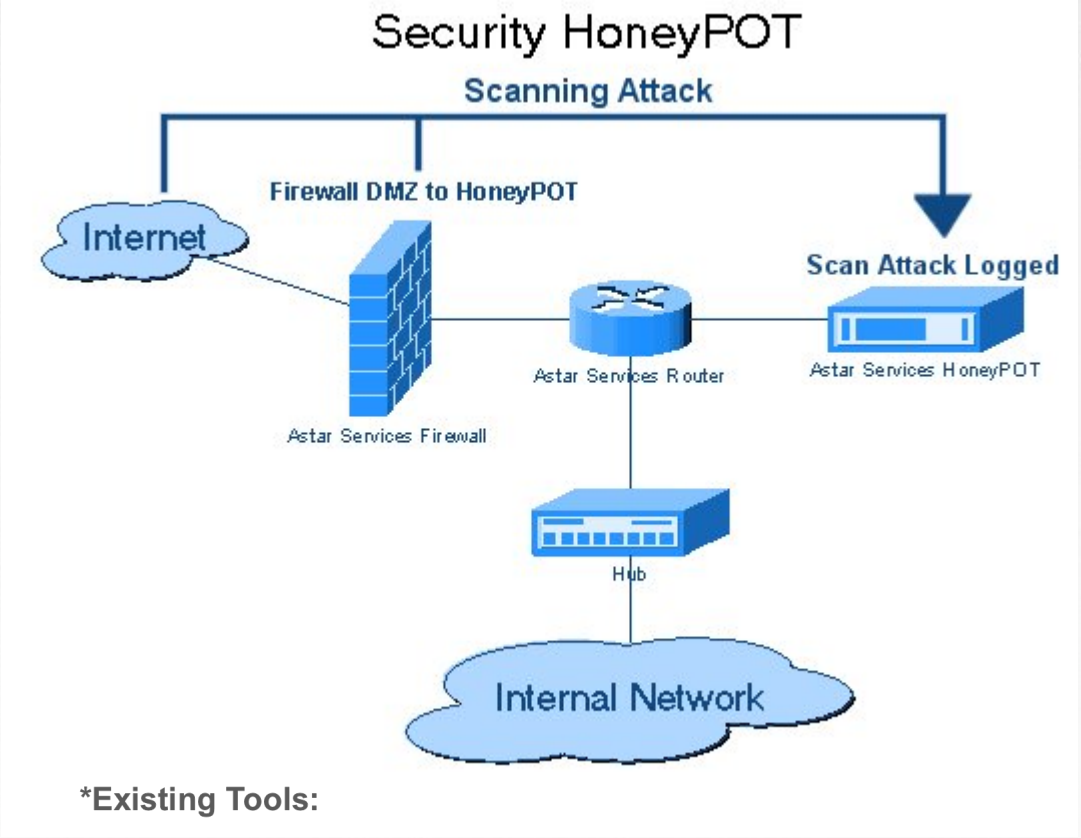
Signal Injection



*Cyber Security of Water SCADA Systems – Part 1: Analysis and Experimentation of Stealthy Deception Attacks; S. Amin et. al.; IEEE Transactions on Control Systems Technology, 2013

Deception-based Cybersecurity

- Honeypots
  - A computing asset used for detecting, deflecting, or counteracting authorized use of information systems (Wikipedia)
  - Can be used to create “Confusion”
    - Confusion induces a time delay on the attack source
    - Gives us more time to counteract appropriately
  - Can be used to increase to cost of attack thereby reducing attack motivation
  - Scale was once upon a time an issue



***Existing Tools:**

https://honeynet.org/

http://www.honeyd.org/

http://conpot.org/

https://github.com/sk4ld/gridpot

http://scadahoneynet.sourceforge.net/ # OLD

Deception-based Cybersecurity

What is dynamic deception

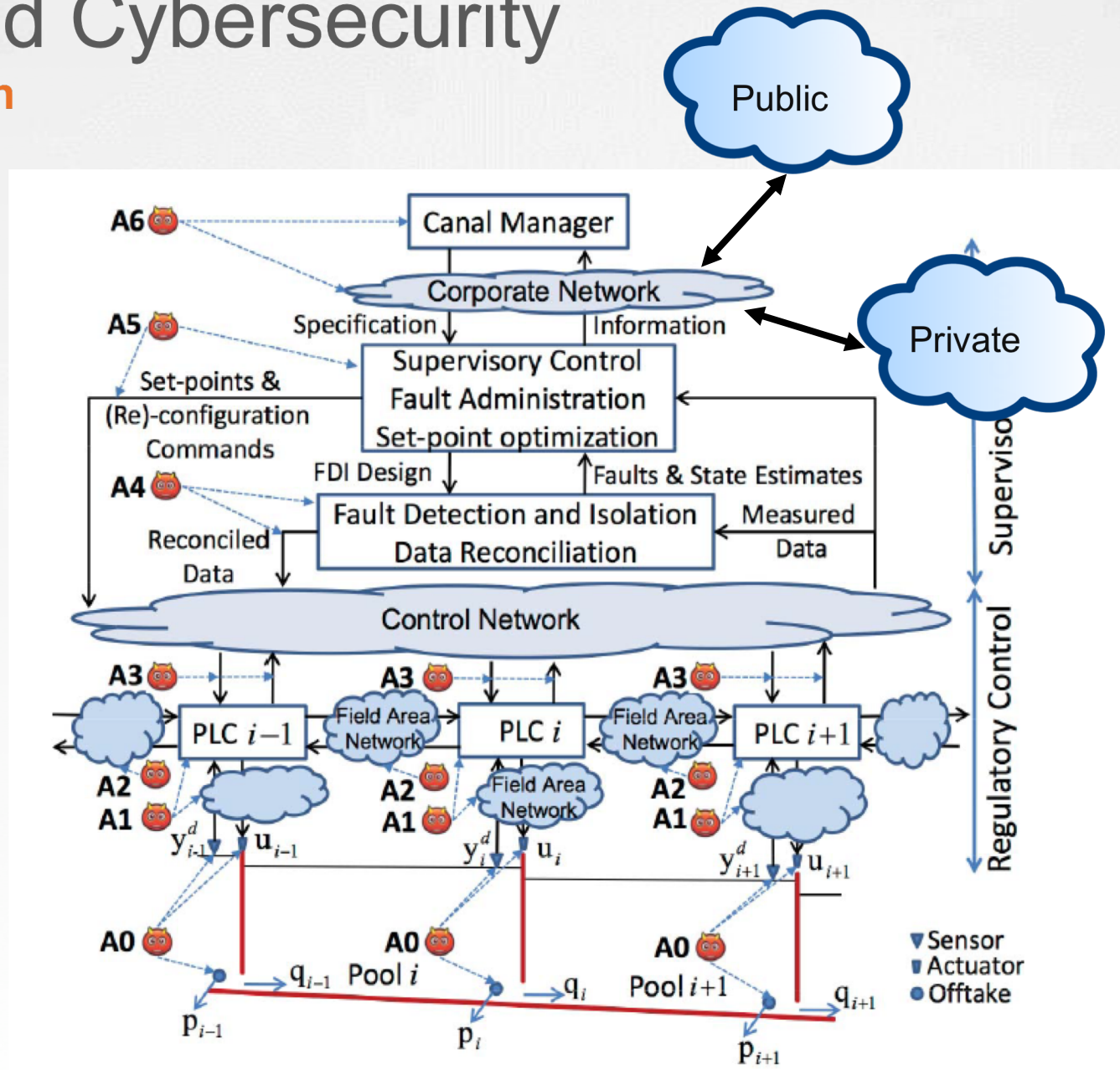
- Primary Goals of this Work
  - **Chaos and Confusion**
    - Using honeypots, **at scale**, to create significant confusion for malicious actors

- Secondary Goals
  - Traditional Outcomes
    - Generating threat intelligence based on the collected data from within the honeypots.
    - Pushing this threat intelligence to our partners via threat intelligence feeds.
    - Implementing real-time controls to stop the attack source based on the gathered threat intelligence

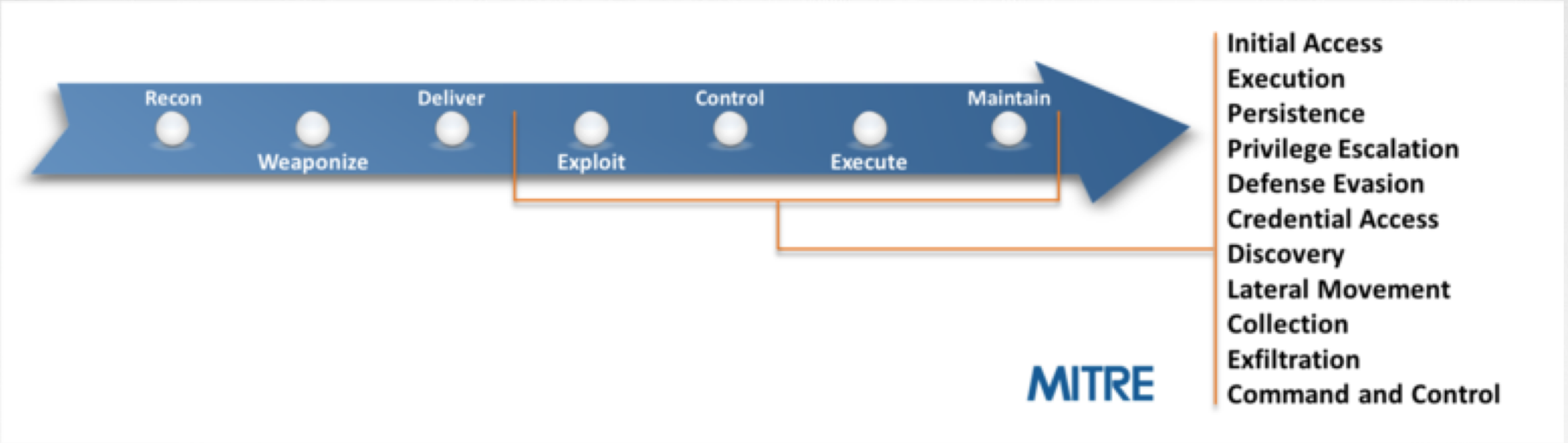
Deception-based Cybersecurity

What is dynamic deception

- Honeypot dynamics
  - Port-based dynamics
  - IP-based dynamics
- **Insider Threat**
  - Lateral movement within after compromise



Embracing Compromise!

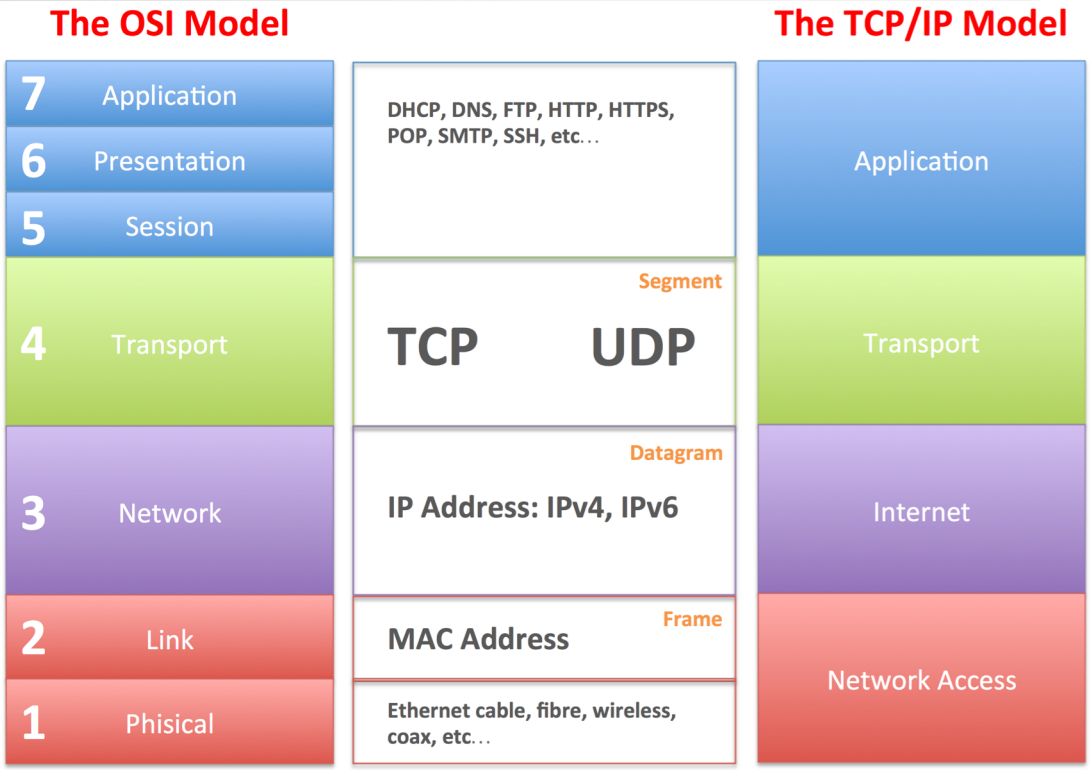


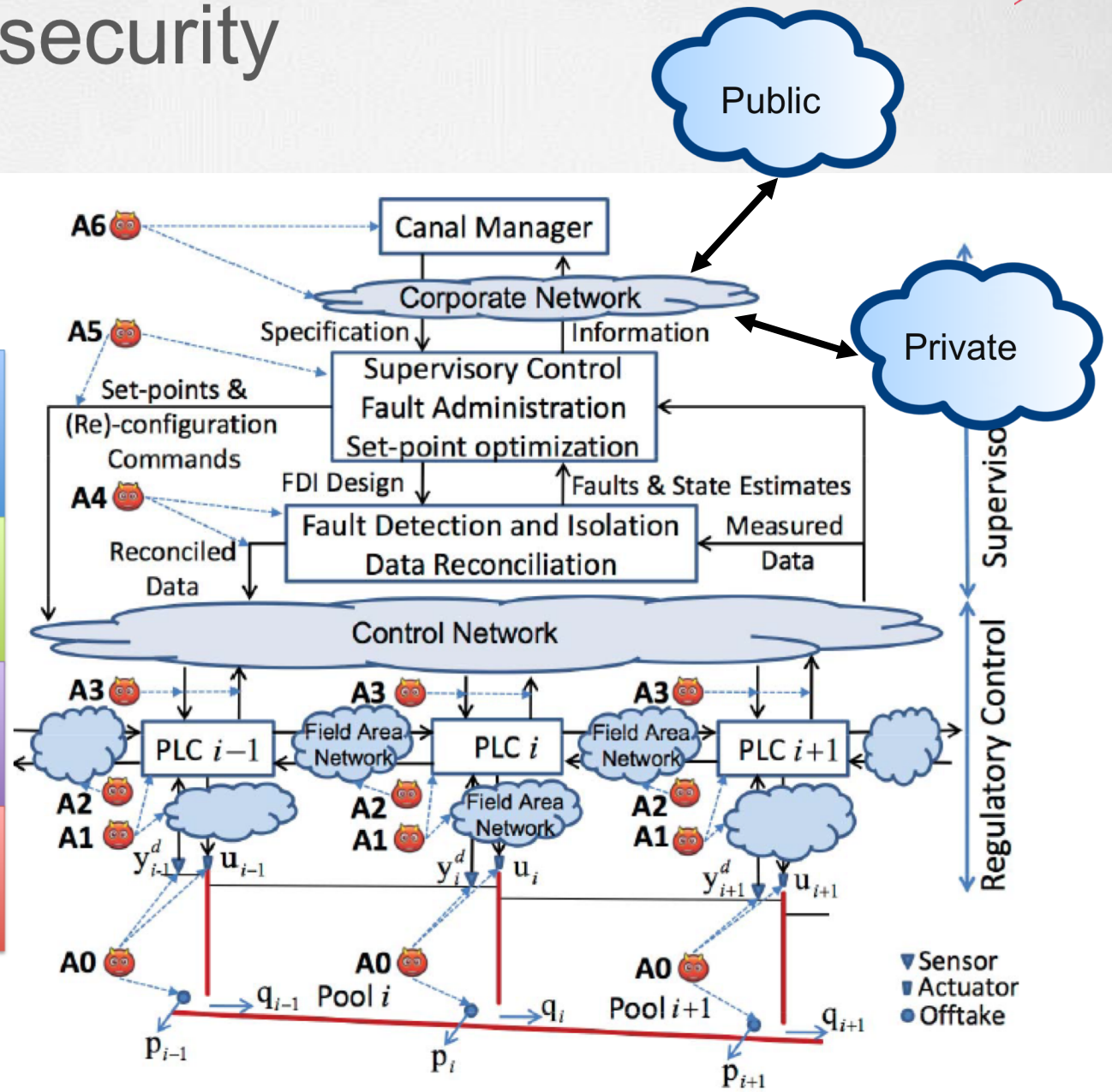
MITRE has developed a curated knowledge base and framework known as Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK). ATT&CK provides knowledge describing behaviors, actions, and processes that a cyber adversary might utilize once **initial access has been gained** within an organization's network.

Source: https://attack.mitre.org/wiki/File:MITRE_attack_tactics.png

Deception-based Cybersecurity

Port and IP dynamics





Deception-based Cybersecurity

Port dynamics

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import socket
import random

server = None
resp = "HTTP/1.1 200 OK\r\nConnection: close\r\n\r\n"

while True:
    if server:
        server.shutdown(socket.SHUT_RDWR)
        server.close()
    else:
        server = socket.socket()
        server.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        host = socket.gethostname()

    port = random.randrange(80,90)
    server.bind((host, port))
    server.listen(1)
    print "Listen on port: %s" % port

    while True:
       client, address = server.accept()
       print 'RECVD FROM: %s' % str(address)
       client.send(resp)
       client.close()
       server.shutdown(socket.SHUT_RDWR)
       server.close()
       server = None
       break
```

Deception-based Cybersecurity

Port dynamics

```
root@lthames-digio:~/ics-dyndec# python simple-dyn.py
Listen on port: 81
RECVD FROM: ('67.205.167.168', 59018)
Listen on port: 88
RECVD FROM: ('67.205.167.168', 56072)
Listen on port: 85
```

```
root@lthames-digio: ~/ics-dyndec
root@lthames-digio:~/ics-dyndec# telnet 67.205.167.168 81
Trying 67.205.167.168...
Connected to 67.205.167.168.
Escape character is '^]'.
HTTP/1.1 200 OK
Connection: close

Connection closed by foreign host.
root@lthames-digio:~/ics-dyndec# telnet 67.205.167.168 81
Trying 67.205.167.168...
telnet: Unable to connect to remote host: Connection refused
root@lthames-digio:~/ics-dyndec# telnet 67.205.167.168 88
Trying 67.205.167.168...
Connected to 67.205.167.168.
Escape character is '^]'.
HTTP/1.1 200 OK
Connection: close

Connection closed by foreign host.
root@lthames-digio:~/ics-dyndec#
```

Deception-based Cybersecurity

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import SocketServer
import socket
import threading
import time
import random

class SimpleTCPHandler(SocketServer.BaseRequestHandler):
    # Must implement this function
    def handle(self):
        resp = """HTTP/1.1 200 OK\r\nDate: Tue, 17 Oct 2017 19:47:29 GMT\r\nExpires: -1\r\nContent-Type: text/html; charset=ISO-8859-1\r\n\r\n"""
        t = threading.current_thread()
        print "Server @ {} handling client {} request".format(t.name, self.client_address)
        self.request.sendall(resp)

class SimpleThreadedServer(SocketServer.ThreadingMixIn, SocketServer.TCPServer):
    pass

class SimpleServer(SocketServer.ThreadingMixIn, SocketServer.TCPServer):
    def __init__(self, port):
        self.host = socket.gethostname()
        self.port = port
        self.allow_reuse_address=True
        try:
            print "Starting @ port: %s" % self.port
            self.server = SimpleThreadedServer((self.host, self.port), SimpleTCPHandler)
            self.server_thread = threading.Thread(target=self.server.serve_forever)
            self.server_thread.daemon = True
            self.server_thread.start()
        except Exception, e:
            self.server=None
            print "Error creating server. Exception: %s" % str(e)


def spin_up():
    population = range(8000, 8900)
    num_ports = 10
    ports = random.sample(population,num_ports)
    servers = list()
    for port in ports:
        s = SimpleServer(port)
        servers.append( s )
    return servers

def spin_down(servers):
    for s in servers:
        if s.server:
            s.server.shutdown()
            s.server.server_close()
```

```python
if __name__ == '__main__':

    while True:
        servers = spin_up()
        time.sleep(15)
        spin_down(servers)
```

Deception-based Cybersecurity

Port dynamics

```
root@lthames-digio:~/ics-dyndec# python simple-multiport-thread-rand.py
Starting @ port: 8194
Starting @ port: 8117
Starting @ port: 8064
Starting @ port: 8477
Starting @ port: 8587
Starting @ port: 8754
Starting @ port: 8515
Starting @ port: 8109
Starting @ port: 8671
Starting @ port: 8242


Starting @ port: 8214
Starting @ port: 8363
Starting @ port: 8081
Starting @ port: 8219
Starting @ port: 8649
Starting @ port: 8514
Starting @ port: 8297
Starting @ port: 8215
Starting @ port: 8619
Starting @ port: 8780
Server @ Thread-21 handling client ('67.205.167.168', 43626) request
```

```
root@lthames-digio:~/ics-dyndec# netstat -tan | grep LIST
tcp        0      0 67.205.167.168:8587     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8109     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8242     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8754     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8117     0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8477     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8671     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8064     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8194     0.0.0.0:*               LISTEN
tcp        0      0 67.205.167.168:8515     0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
root@lthames-digio:~/ics-dyndec# telnet 67.205.167.168 8780
Trying 67.205.167.168...
Connected to 67.205.167.168.
Escape character is '^]'.
HTTP/1.1 200 OK
Date: Tue, 17 Oct 2017 19:47:29 GMT
Expires: -1
Content-Type: text/html; charset=ISO-8859-1

Connection closed by foreign host.
root@lthames-digio:~/ics-dyndec#
```

Deception-based Cybersecurity

Port dynamics

- What issues do we see?
  - Code complexity
  - Light-weight interaction
  - ….
- What can we do about it?
  - Twisted

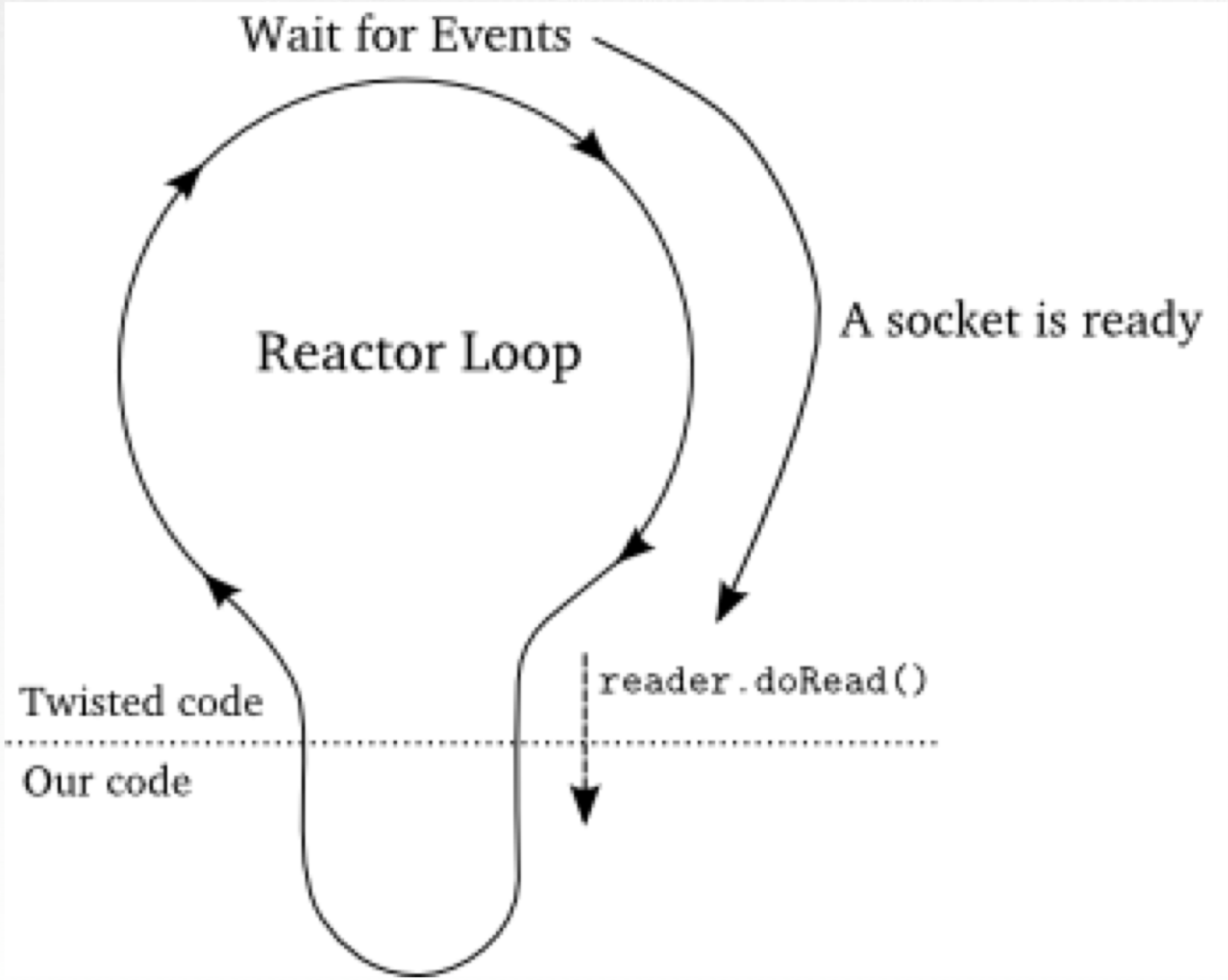
Deception-based Cybersecurity

Port dynamics

- What is Twisted?
  - An event-driven networking engine written in Python
    - Based on a reactive programming model
      - Essentially lets you work with highly asynchronous applications
  - Comes “with batteries”
    - Web servers, Mail Servers, Chat servers and more
  - Let’s the programmer focus on the Application Protocol
  - Many projects out there based on Twisted that fit well with creating honeyports
    - IoT based protocol projects
    - OT based protocol projects

Deception-based Cybersecurity

Port dynamics



Deception-based Cybersecurity

Port dynamics

```
1  from twisted.web.server import Site
2  from twisted.web.static import File
3  from twisted.internet import reactor
4  import random
5
6
7  def rrun():
8      reactor.removeAll()
9      port = random.randrange(8000,8100)
10     print "Listening: %s" % port
11     resource = File('web')
12     factory = Site(resource)
13     reactor.callLater(25, rrun)
14     reactor.listenTCP(port, factory)
15
16
17
18 reactor.callLater(1, rrun)
19 reactor.run()
```

```
root@lthames-digio:~/ics-dyndec# python simple-twisted-web-dyn.py
Listening: 8042
Listening: 8075
Listening: 8020
```

```
root@lthames-digio:~/ics-dyndec# telnet localhost 8020
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET / HTTP/1.1

HTTP/1.1 200 OK
Content-Length: 46
Accept-Ranges: bytes
Server: TwistedWeb/16.0.0
Last-Modified: Sat, 21 Oct 2017 19:18:00 GMT
Date: Sat, 21 Oct 2017 20:07:43 GMT
Content-Type: text/html

<HTML>
<BODY>
Hello World<br>
</BODY>
</HTML>
.
HTTP/1.1 400 Bad Request

Connection closed by foreign host.
root@lthames-digio:~/ics-dyndec#
```

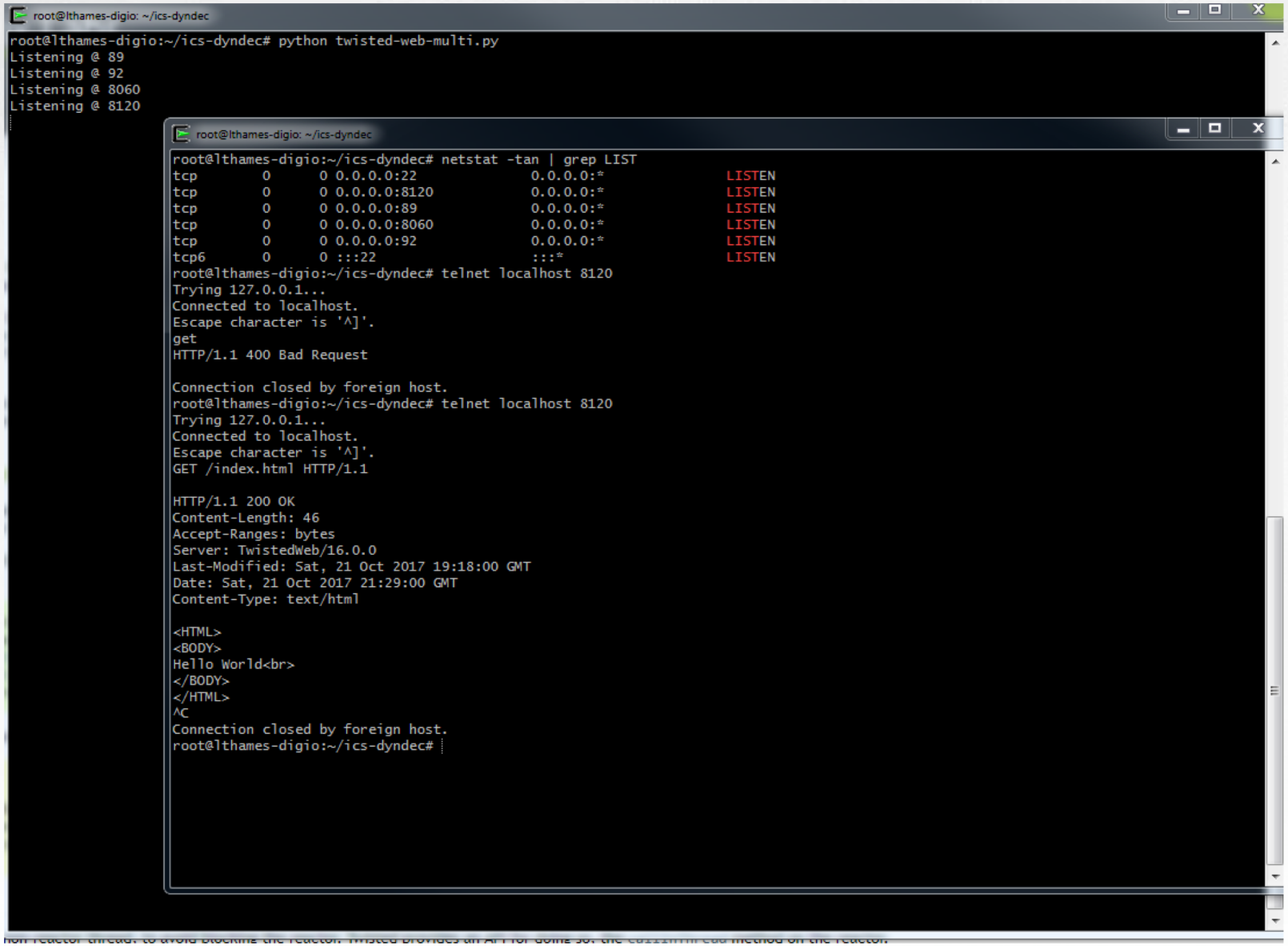
Deception-based Cybersecurity

Port dynamics

```
1  from twisted.web.server import Site
2  from twisted.web.static import File
3  from twisted.internet import reactor
4  import random
5
6
7  class SimpleWeb(object):
8      def __init__(self, port_low, port_high):
9          self.port = random.randrange(port_low, port_high)
10         self.factory = Site( File('web') )
11         print "Listening @ %s" % self.port
12         reactor.listenTCP(self.port, self.factory)
13
14
15 if __name__ == '__main__':
16     s1 = SimpleWeb(80, 90)
17     s2 = SimpleWeb(91, 100)
18     s3 = SimpleWeb(8000, 8100)
19     s4 = SimpleWeb(8101, 8200)
20
21     reactor.run()
```

Deception-based Cybersecurity

Port dynamics



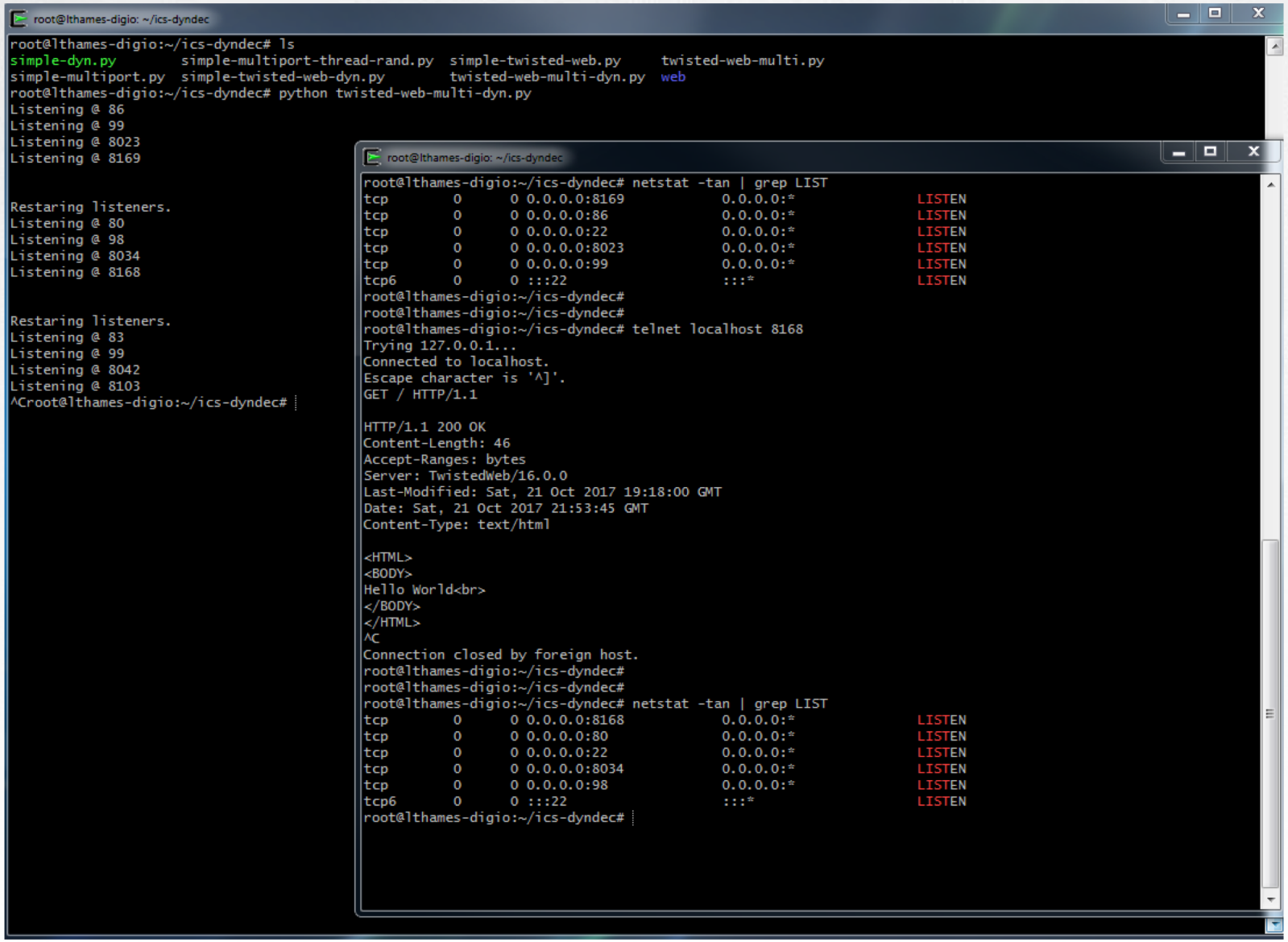
Deception-based Cybersecurity

Port dynamics

```
from twisted.web.server import Site
from twisted.web.static import File
from twisted.internet import reactor
import random


class SimpleWeb(object):
    def __init__(self, port_low, port_high):
        self.port_low = port_low
        self.port_high = port_high
        self.factory = Site( File('web') )
        self.spinUp()

    def spinUp(self):
        self.port = random.randrange(self.port_low, self. port_high)
        print "Listening @ %s" % self.port
        reactor.listenTCP(self.port, self.factory)


def rrun(servers):
    print "\n\nRestaring listeners."
    reactor.removeAll()
    for server in servers:
        server.spinUp()
    reactor.callLater(20, rrun, servers)


if __name__ == '__main__':
    s1 = SimpleWeb(80, 90)
    s2 = SimpleWeb(91, 100)
    s3 = SimpleWeb(8000, 8100)
    s4 = SimpleWeb(8101, 8200)
    servers = [s1, s2, s3, s4]

    reactor.callLater(20, rrun, servers)
    reactor.run()
```

Deception-based Cybersecurity

Port dynamics



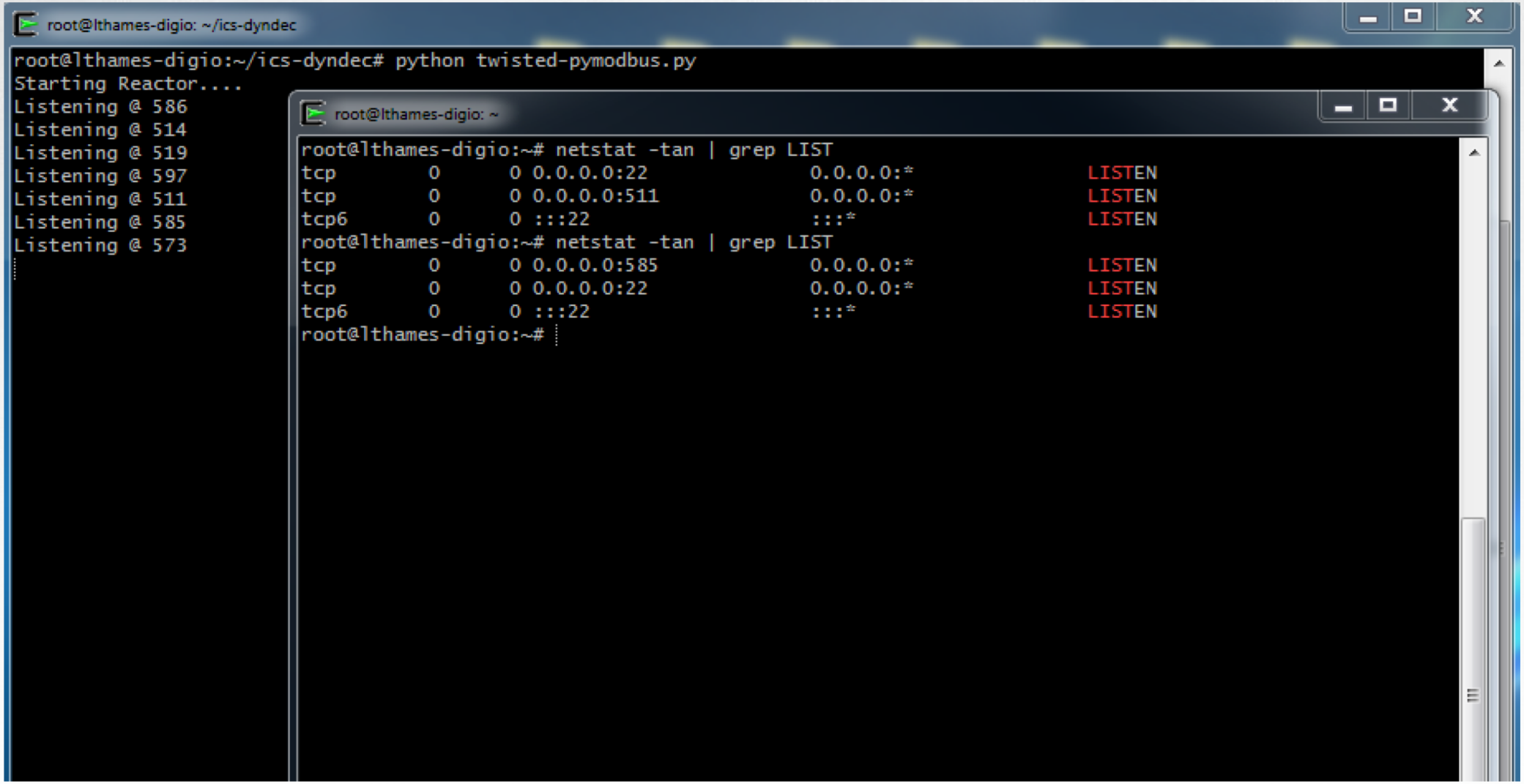
Deception-based Cybersecurity

Port dynamics

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from pymodbus.server.async import ModbusServerFactory
from pymodbus.transaction import ModbusSocketFramer
from pymodbus.device import ModbusDeviceIdentification
from pymodbus.datastore import ModbusSequentialDataBlock
from pymodbus.datastore import ModbusSlaveContext, ModbusServerContext
import random
from twisted.internet import reactor


def rrun(factory):
    reactor.removeAll()
    port = random.randrange(500, 599)
    print "Listening @ %s" % port
    reactor.listenTCP(port, factory)
    reactor.callLater(10, rrun, factory)


store = ModbusSlaveContext(
    di = ModbusSequentialDataBlock(0, [17]*100),
    co = ModbusSequentialDataBlock(0, [17]*100),
    hr = ModbusSequentialDataBlock(0, [17]*100),
    ir = ModbusSequentialDataBlock(0, [17]*100))
context = ModbusServerContext(slaves=store, single=True)


identity = ModbusDeviceIdentification()
identity.VendorName  = 'Pymodbus'
identity.ProductCode = 'PM'
identity.VendorUrl   = 'http://github.com/bashwork/pymodbus/'
identity.ProductName = 'Pymodbus Server'
identity.ModelName   = 'Pymodbus Server'
identity.MajorMinorRevision = '1.0'

framer  = ModbusSocketFramer
factory = ModbusServerFactory(context, framer, identity)

print "Starting Reactor...."
reactor.callLater(2, rrun, factory)
reactor.run()

```

Deception-based Cybersecurity

Port dynamics



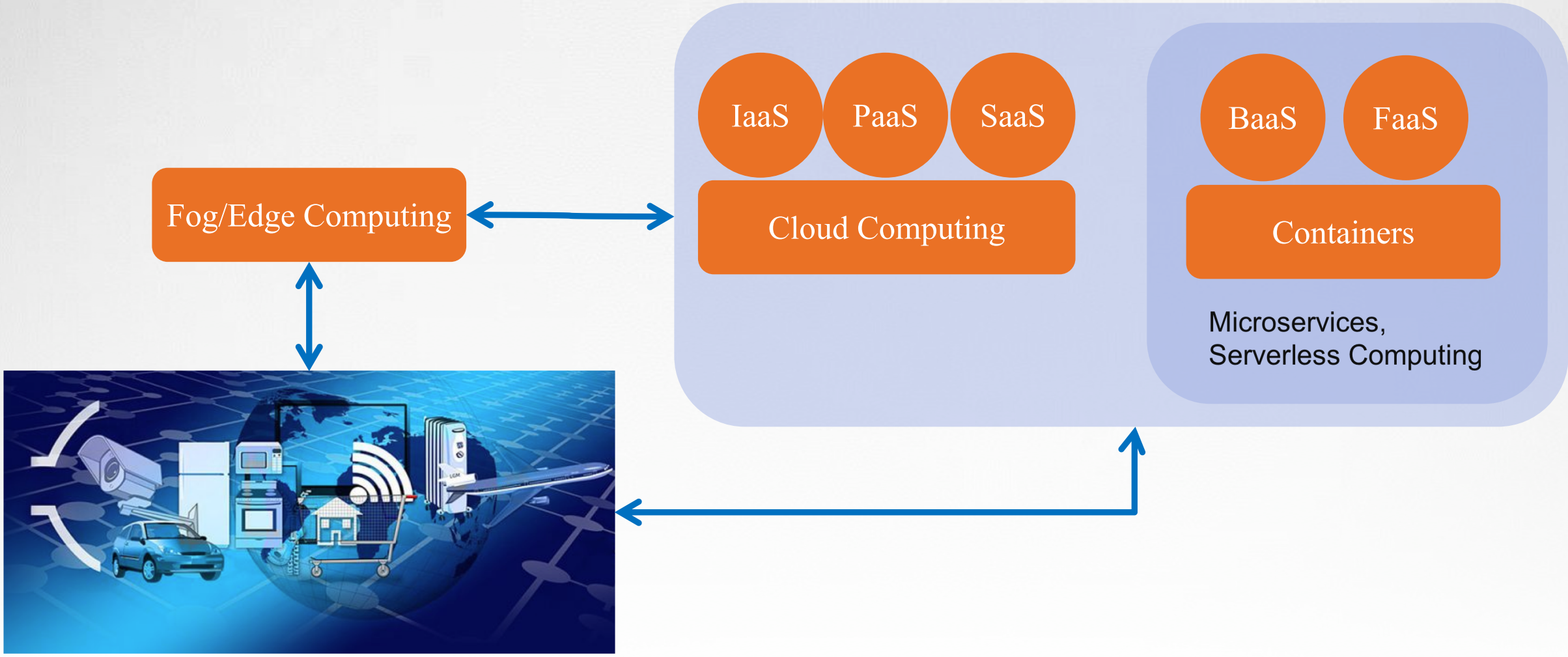
Key To Success

SCALE is KEY!

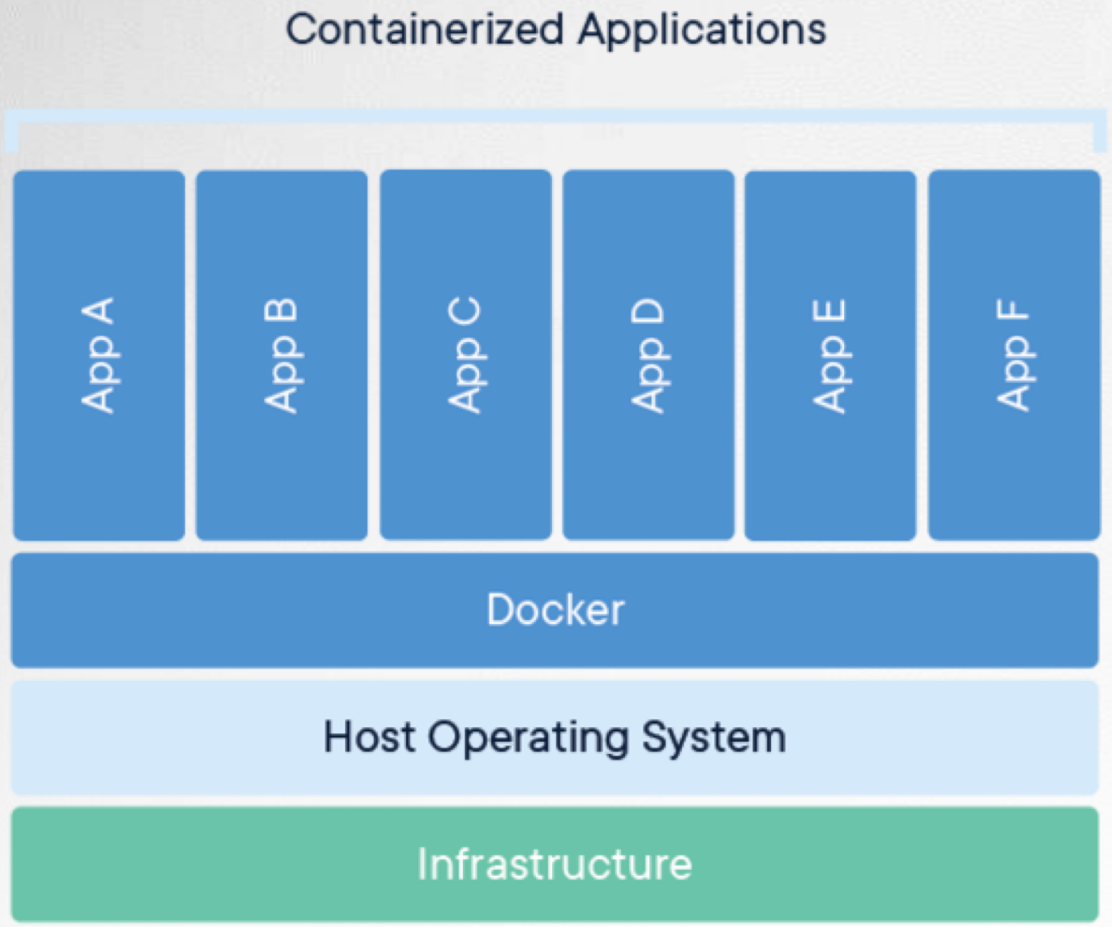
Can we make this tool better?

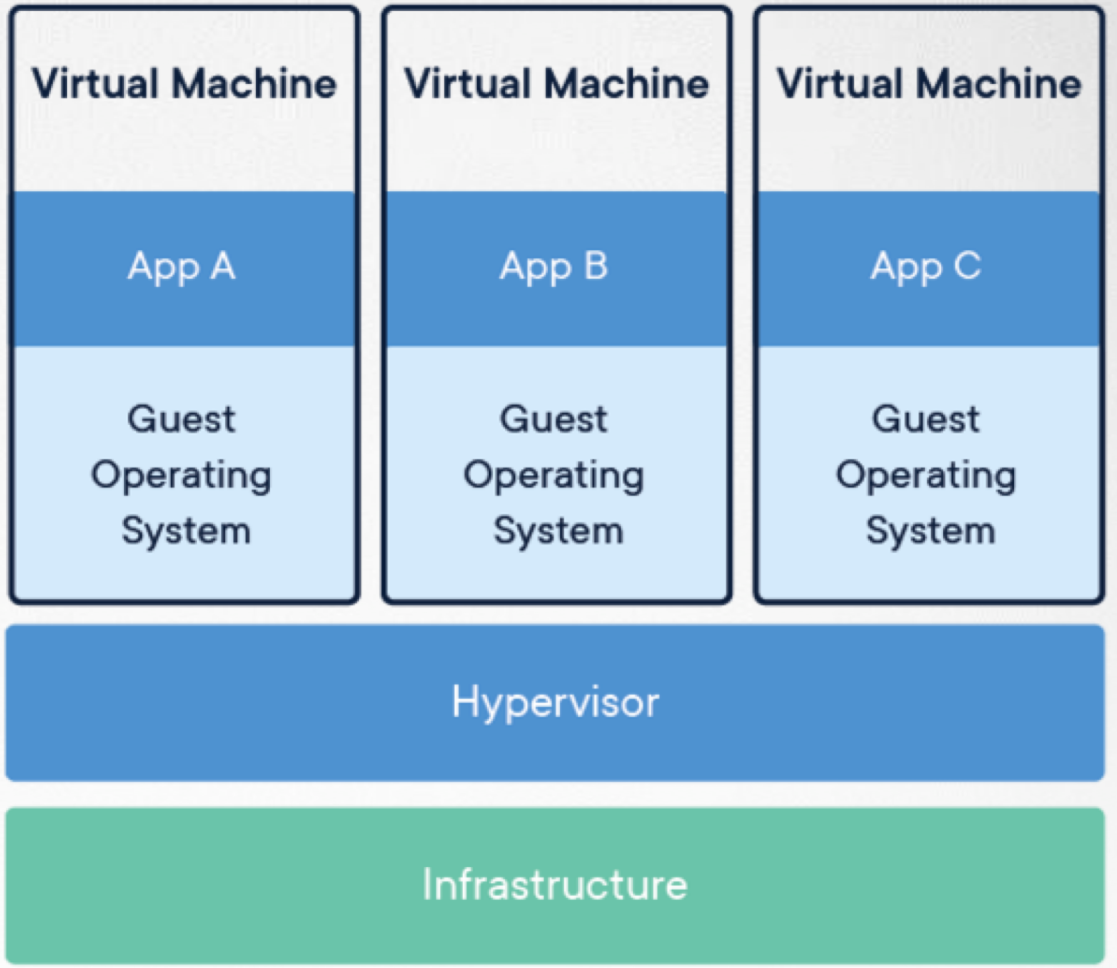
How can we scale?

Containers to the Rescue



What are Containers





Source: https://www.docker.com/resources/what-container

IP Dynamics with Containers

- Fairly Straightforward
  - Unused IP space
  - Container Orchestration
    - Docker Swarm
    - Kubernetes

Port Dynamics with Containers

Required a little work and code changes to Twisted applications

20 lines (13 sloc) | 500 Bytes

```
1   FROM python:2.7-slim
2
3   # Set the working directory to /app
4   WORKDIR /app
5
6   # Copy the current directory contents into the container at /app
7   ADD . /app
8
9   # Install Dependencies
10  RUN apt-get update && apt-get install -y gcc
11
12  # Install any needed packages specified in requirements.txt
13  RUN pip install --trusted-host pypi.python.org -r requirements.txt
14
15  # Make port 80 available to the world outside this container
16  EXPOSE 80
17
18  # Run twisted-web.py when the container launches
19  CMD ["python", "twisted-web.py"]
```

docker run -d -p 8080:80 jlthames2/thddt-web

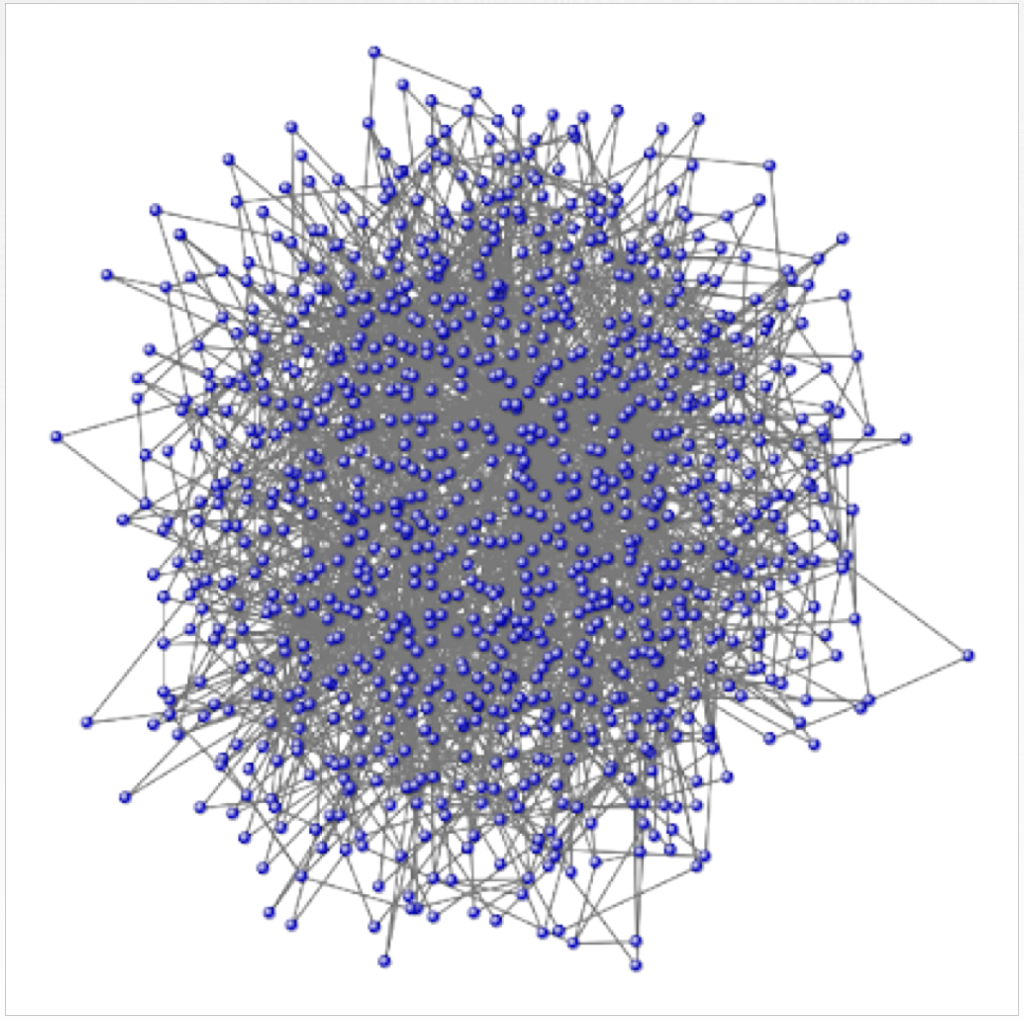
Solution

- Let’s look at the code
- Let’s run a demo

Observations

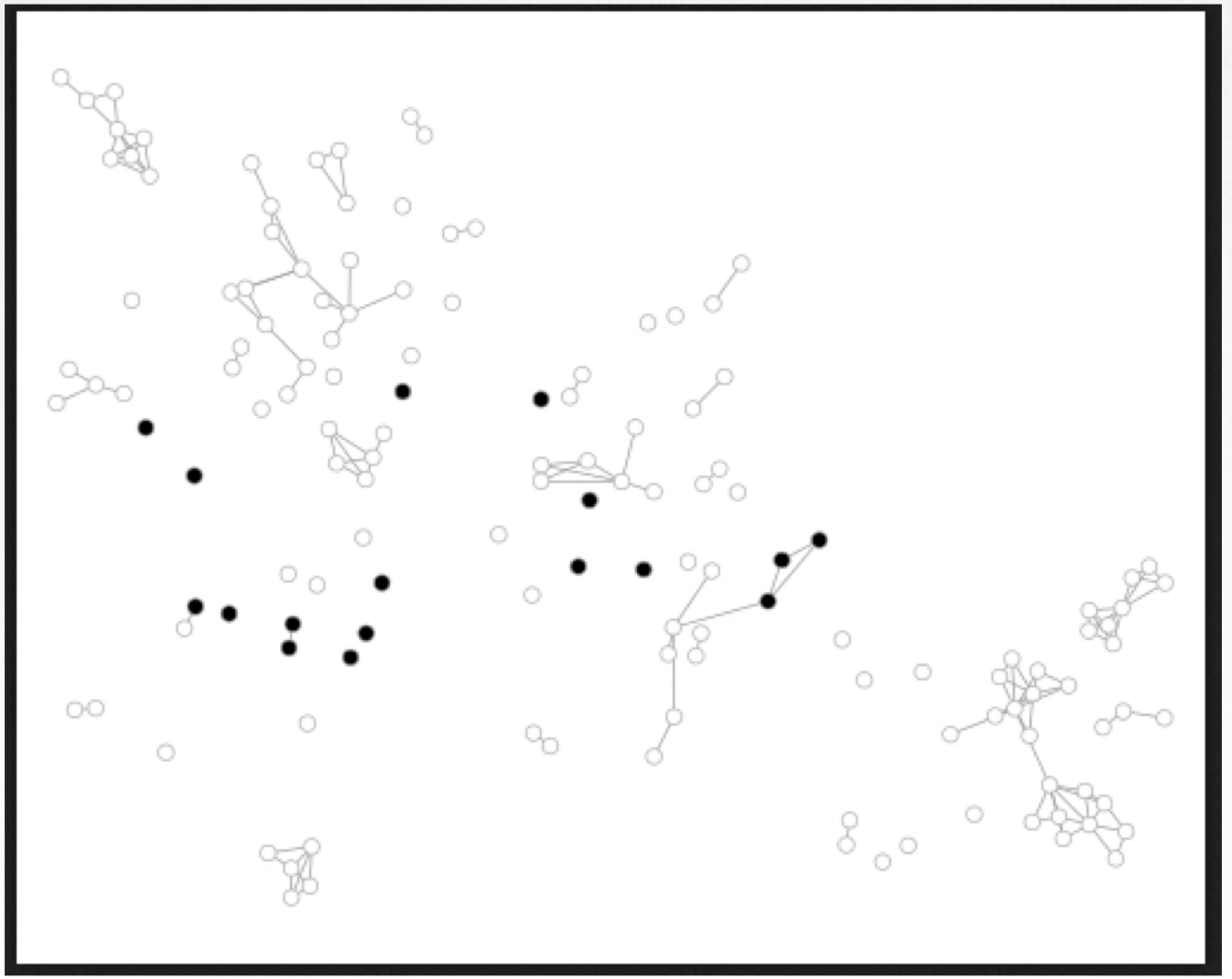
- Goals
  - Chaos and Confusion
    - Cause havoc for the adversary during internal recon
    - Increase the amount of time attackers need to find real environments within our network
      - Protect from insider threats
      - Protect from external threats
    - **Increase** the attacker's **costs**, thereby **reducing** their gains and thereby **shattering** their motivation
- The technology now exists that allows us to **scale** up these types of deception systems so that we can realize more relevant deception

Complex but not enough chaos



Dynamic Network via Dynamic Deception

https://graph-tool.skewed.de



Closing Remarks

- Future Work
  - Add more “batteries” to the twisted ecosystem, specifically addressing the needs of converged IT/OT environments
  - Build out orchestration mechanism, study what is needed for dynamic deception with containers
  - Study attacker behaviors when interacting with Twisted Haystacks
    - What are the best parameters, etc.
- Project Location:
  - https://github.com/jlthames2/ddt
    - Original Project
  - https://github.com/jlthames2/thddt
    - New project with containerized apps (pull from docker hub or build your own containers with my Dockerfiles)

tripwire®

CONFIDENCE: SECURED

Thank you

lthames@tripwire.com
@Lane_Thames