RSAConference2018
San Francisco | April 16 – 20 | Moscone Center

SESSION ID: SPO3-W12

HONEYPOTS 2.0: DEFENDING INDUSTRIAL SYSTEMS WITH DYNAMIC DECEPTION

Lane Thames, PhD

Senior Security Researcher
Tripwire, Inc.
@Lane_Thames

Motivation

Who has the upper hand in cybersecurity? The good guys or the bad buys? Why?

Agenda

- Industrial Internet of Things
- Cybersecurity challenges for the Industrial Internet of Things
- Deception Technologies for Cybersecurity
  - Honeypots
- Dynamic Deception
  - Next generation Honeypots
  - Scale

RSAConference2018

NOW MATTERS

#RSAC

INDUSTRIAL INTERNET OF THINGS AND ITS CYBERSECURITY CHALLENGES

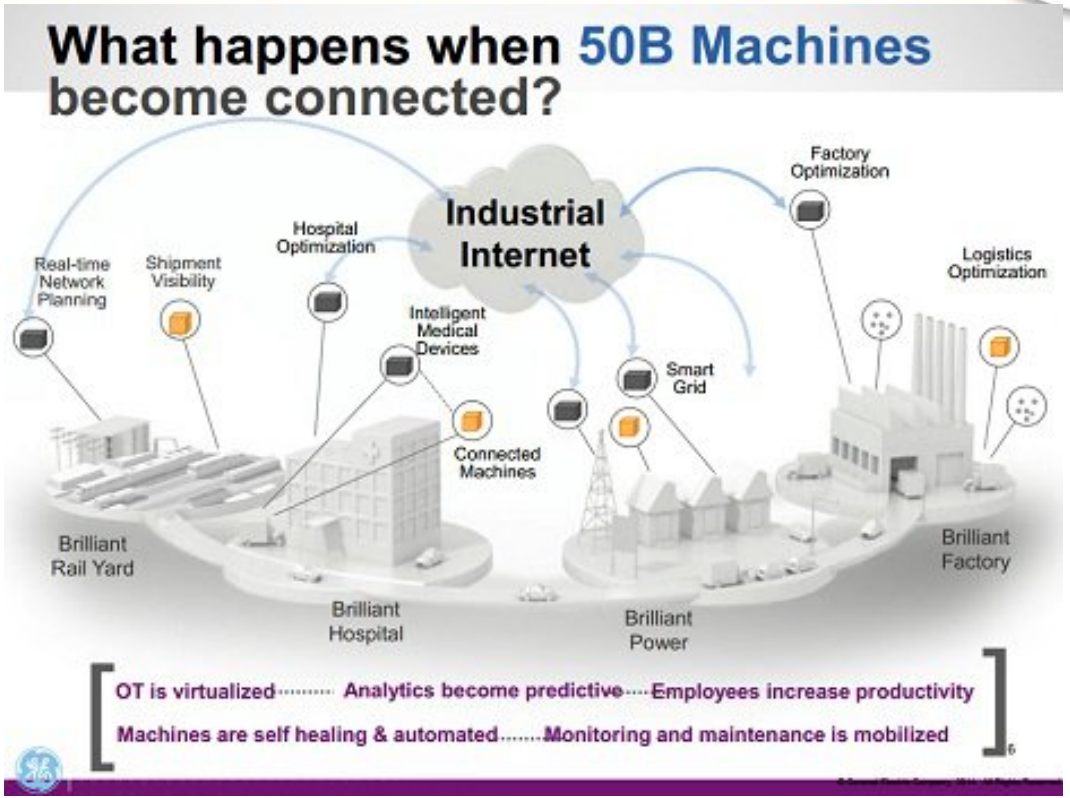
Industrial Internet of Things

- Smart Power Grids, Smart Logistics, Smart Inventory, Smart Machine Diagnostics
- Self-monitoring, Group-monitoring
- Self-configuration, Group-configuration
- Self-healing, Group-healing

Provides:

- Operational Efficiencies
- Outcome-driven Processes
- Machine-to-Human Collaboration

Countless Value Creation Opportunities



Industrial Internet of Things

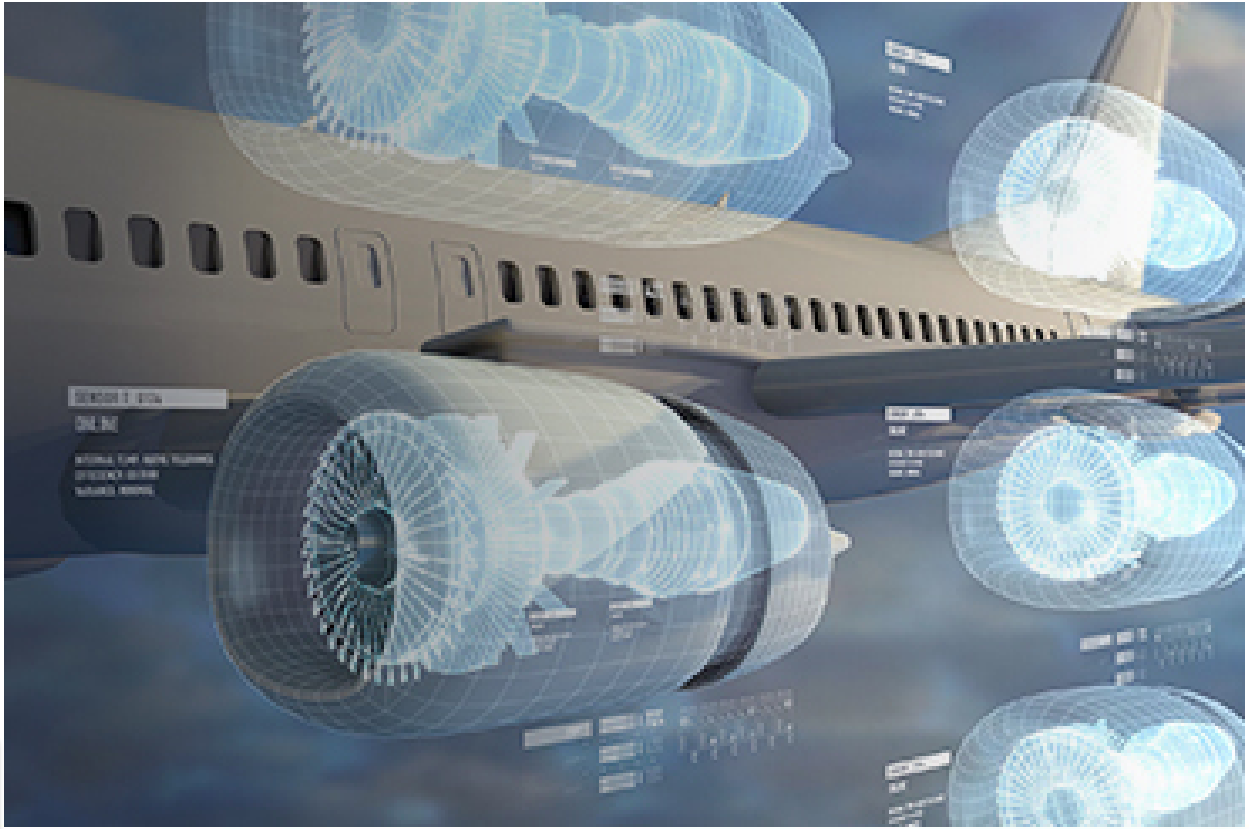
What is a "Digital Twin"?

Wikipedia:

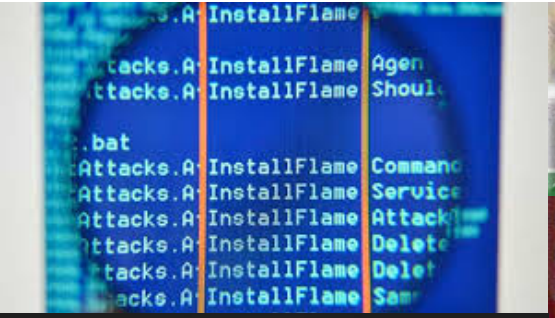
Digital twin refers to a digital replica of physical assets, processes and systems that can be used for various purposes. The digital representation provides both the elements and the dynamics of how an Internet of Things device operates and lives throughout its life cycle.

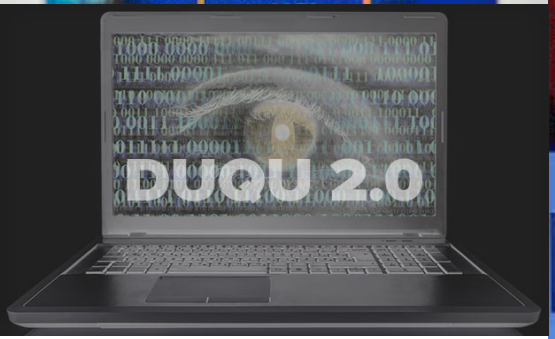
Digital Twins integrate artificial intelligence, machine learning and software analytics with data to create living digital simulation models that update and change as their physical counterparts change.

Industrial Internet of Things



Industrial Internet of Things: What will prevent us from achieving its full potential?





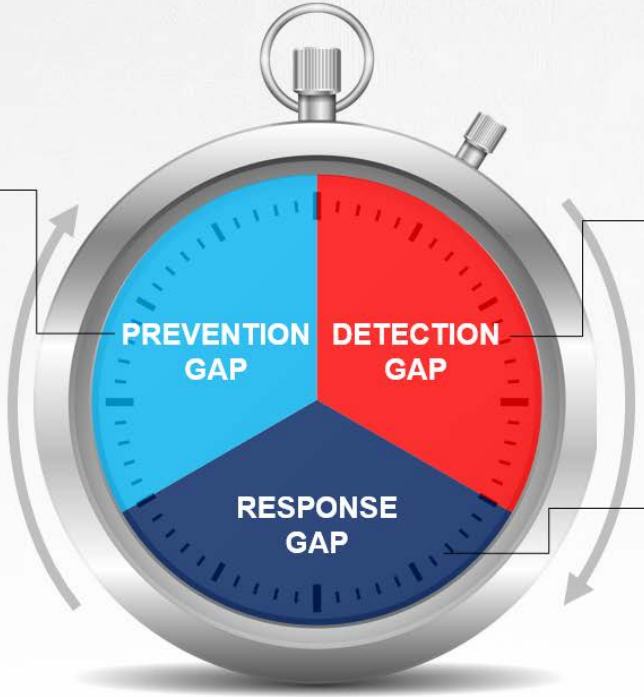
tripwire

Our Approximate Cybersecurity Solution

Time is always against us. How can we change that?

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?

RSAConference2018

NOW MATTERS

#RSAC

DECEPTION TECHNOLOGIES FOR CYBERSECURITY AND DYNAMIC DECEPTION

Deception Technologies

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

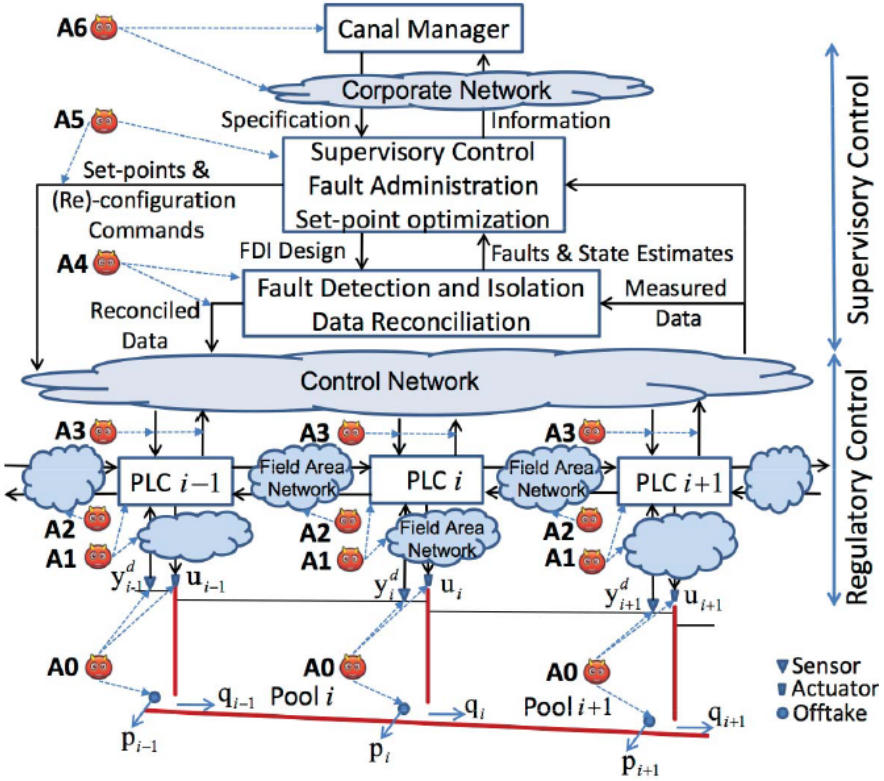
- (of a thing) give a mistaken impression.
  "the area may seem to offer nothing of interest, but don't be deceived"
- fail to admit to oneself that something is true.
  "enabling the rulers to deceive themselves about the nature of their own rule"

Deception-based Cyberattacks - General

- Social Engineering
- Phishing
- Spam

Deception-based Cyberattacks – IIoT Specific

- Spoofed Signals
  - Sensor measurements
  - Control inputs
  - Timestamps
  - Identity information



*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

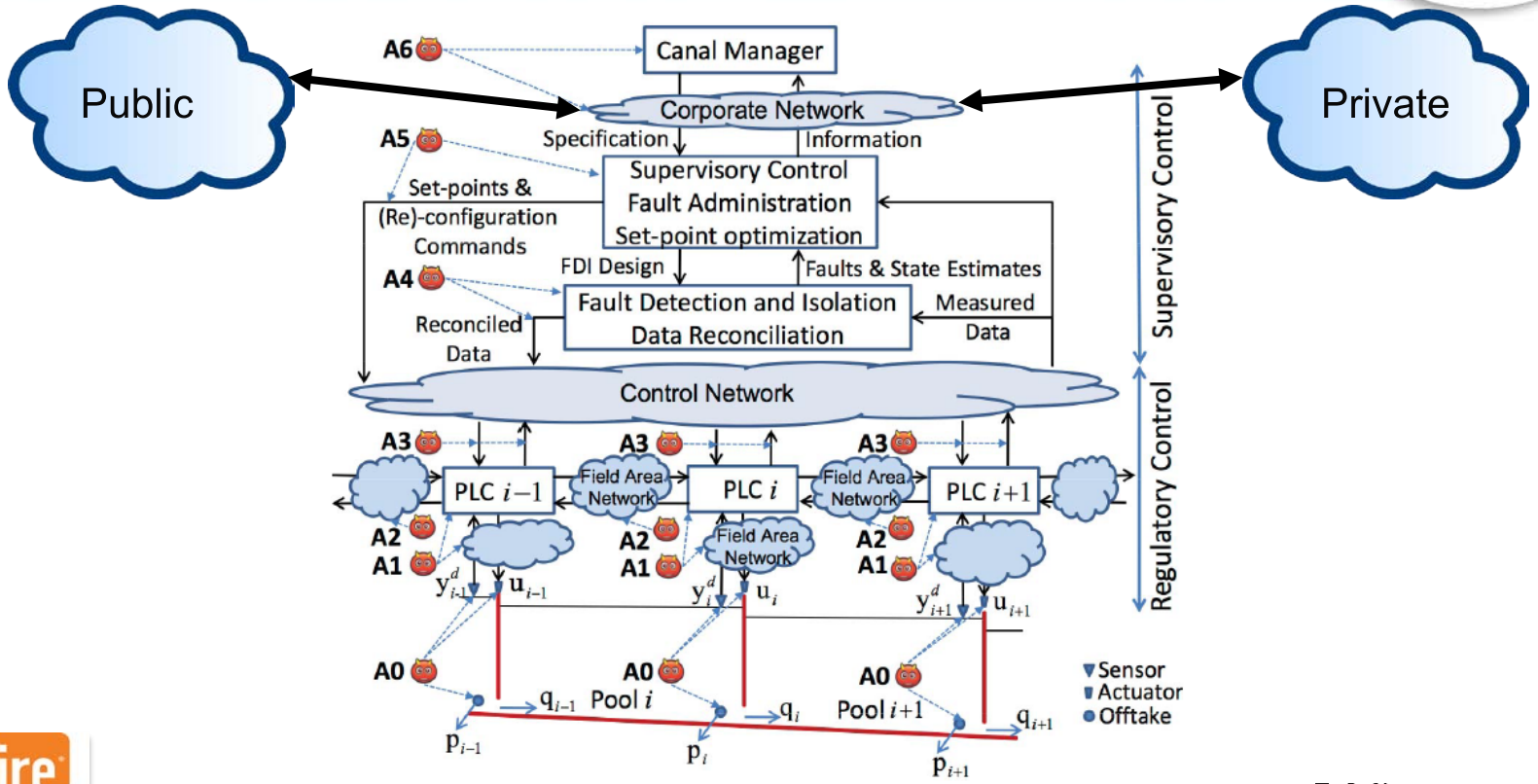
Deception-based Cybersecurity

- 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

Deception-based Cybersecurity

- Honeypots & Dynamic Deception
  - IP-based dynamics
    - DevOps Tool Chains
  - Port-based dynamics
    - Software-based implementation
    - Managed/Deployed via DevOps Tool Chains
- Goals:
  - Primary: Create significant confusion via scale for attackers in such a way to cause delays for their activities
  - Secondary: Use dynamic deception at scale to detect real-time attacks, to generate threat intelligence, and to implement real-time controls

Deception-based Cybersecurity



Dynamic Deception: Port-based Dynamics


```python
#!/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
```

Dynamic Deception: Port-based Dynamics

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

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

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

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

Dynamic Deception: Port-based Dynamics

#RSAC

```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\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)
```

tripwire

RSAConference2018

Dynamic Deception: Port-based 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 ('[redacted], 43626) request
```

```
root@lthames-digio:~/ics-dyndec# netstat -tan | grep LIST
tcp        0      0 [redacted]:8587     0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8109     0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8242     0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8754     0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8117     0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22          0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8477     0.0.0.0:*               LISTEN
tcp        0      0 [redacted]8671      0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8064     0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8194     0.0.0.0:*               LISTEN
tcp        0      0 [redacted]:8515     0.0.0.0:*               LISTEN
tcp6       0      0 :::22               :::*                    LISTEN
root@lthames-digio:~/ics-dyndec# telnet [redacted]8780
Trying [redacted]...
Connected to [redacted].
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#
```

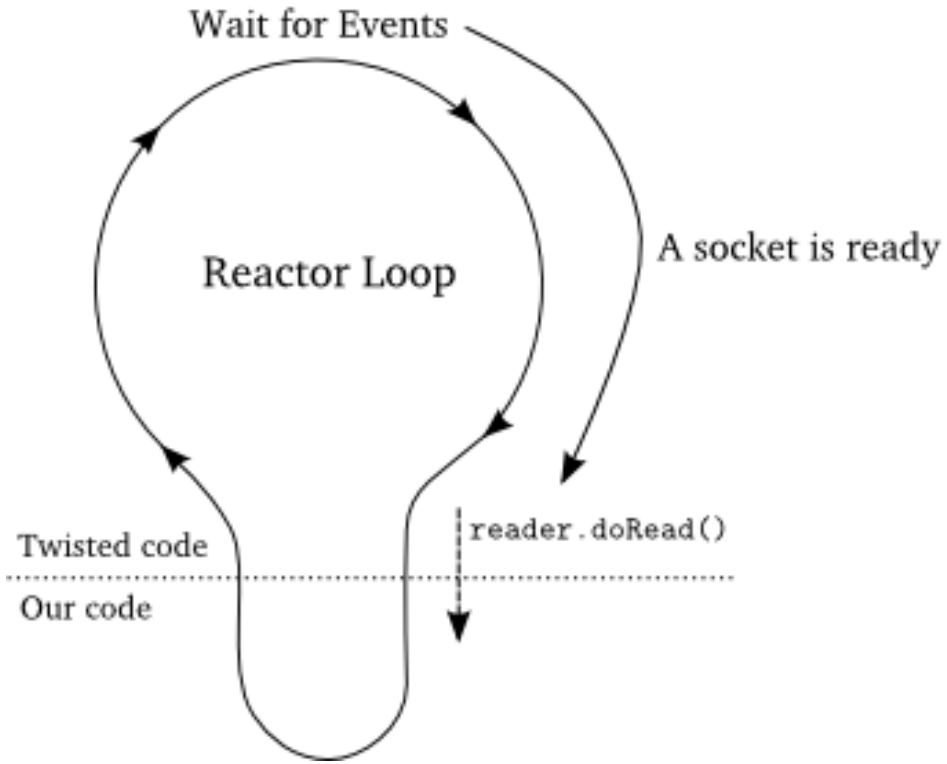
Dynamic Deception: Port-based Dynamics

- Problems with the aforementioned approach?
  - Code complexity
  - Light-weight honeypot interaction
- We can solve these problems with ‘Twisted’!

Dynamic Deception: Port-based 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, SSH servers, Chat servers and many more
  - Let’s the programmer focus on the Application Protocol
  - Many projects available based on Twisted that fit well with creating honeypots
    - IoT based projects
    - OT (Operational Technology) based projects

Dynamic Deception: Port-based Dynamics



Dynamic Deception: Port-based Dynamics

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


def rrun():
    reactor.removeAll()
    port = random.randrange(8000,8100)
    print "Listening: %s" % port
    resource = File('web')
    factory = Site(resource)
    reactor.callLater(25, rrun)
    reactor.listenTCP(port, factory)



reactor.callLater(1, rrun)
reactor.run()
```

```
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#
```

Dynamic Deception: Port-based Dynamics


```python
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 = random.randrange(port_low, port_high)
        self.factory = Site( File('web') )
        print "Listening @ %s" % self.port
        reactor.listenTCP(self.port, self.factory)


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

    reactor.run()
```

Dynamic Deception: Port-based Dynamics

#RSAC

```
root@lthames-digio: ~/ics-dyndec
root@lthames-digio:~/ics-dyndec# python twisted-web-multi.py
Listening @ 89
Listening @ 92
Listening @ 8060
Listening @ 8120
```

```
root@lthames-digio: ~/ics-dyndec
root@lthames-digio:~/ics-dyndec# netstat -tan | grep LIST
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8120            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:89              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8060            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:92              0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
root@lthames-digio:~/ics-dyndec# telnet localhost 8120
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
get
HTTP/1.1 400 Bad Request

Connection closed by foreign host.
root@lthames-digio:~/ics-dyndec# telnet localhost 8120
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
GET /index.html 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 21:29:00 GMT
Content-Type: text/html

<HTML>
<BODY>
Hello World<br>
</BODY>
</HTML>
^C
Connection closed by foreign host.
root@lthames-digio:~/ics-dyndec#
```

Dynamic Deception: Port-based Dynamics

```python
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()
```

Dynamic Deception: Port-based Dynamics

```
root@lthames-digio: ~/ics-dyndec
root@lthames-digio:~/ics-dyndec# ls
simple-dyn.py        simple-multiport-thread-rand.py  simple-twisted-web.py     twisted-web-multi.py
simple-multiport.py  simple-twisted-web-dyn.py        twisted-web-multi-dyn.py  web
root@lthames-digio:~/ics-dyndec# python twisted-web-multi-dyn.py
Listening @ 86
Listening @ 99
Listening @ 8023
Listening @ 8169


Restaring listeners.
Listening @ 80
Listening @ 98
Listening @ 8034
Listening @ 8168


Restaring listeners.
Listening @ 83
Listening @ 99
Listening @ 8042
Listening @ 8103
^Croot@lthames-digio:~/ics-dyndec#
```

```
root@lthames-digio: ~/ics-dyndec
root@lthames-digio:~/ics-dyndec# netstat -tan | grep LIST
tcp        0      0 0.0.0.0:8169            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:86              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8023            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:99              0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
root@lthames-digio:~/ics-dyndec#
root@lthames-digio:~/ics-dyndec#
root@lthames-digio:~/ics-dyndec# telnet localhost 8168
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 21:53:45 GMT
Content-Type: text/html

<HTML>
<BODY>
Hello World<br>
</BODY>
</HTML>
^C
Connection closed by foreign host.
root@lthames-digio:~/ics-dyndec#
root@lthames-digio:~/ics-dyndec#
root@lthames-digio:~/ics-dyndec# netstat -tan | grep LIST
tcp        0      0 0.0.0.0:8168            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:8034            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:98              0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
root@lthames-digio:~/ics-dyndec#
```

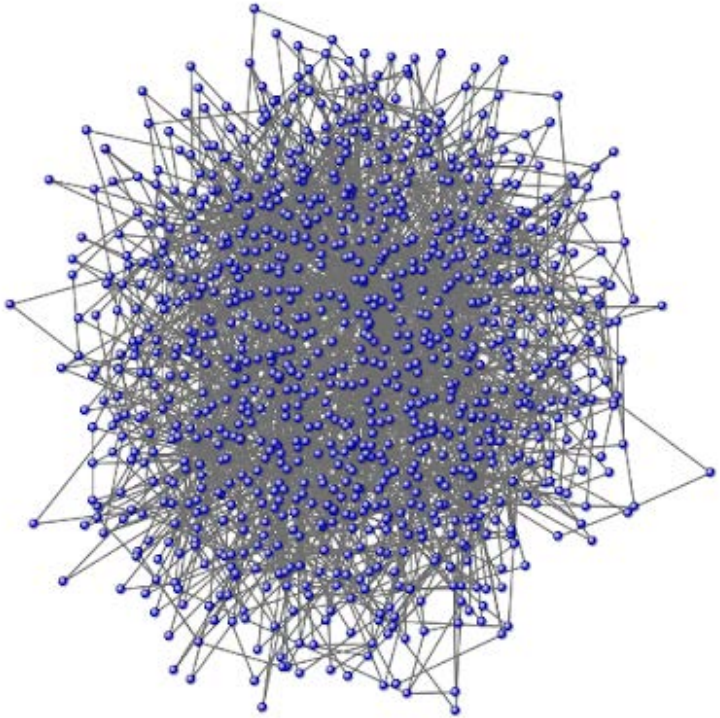
Dynamic Deception: Port-based Dynamics


```
#!/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
```

Dynamic Deception: Port-based Dynamics


```python
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()
```

Dynamic Deception: **Scale**



Summary

- Industrial Internet of Things
- Dynamic Deception
  - Dynamic & Static Honeypots
    - Port Based Dynamics
    - IP Based Dynamics
    - Scale
- Python Twisted Networking Framework
- Code available at Github:
  - https://github.com/jlthames2/ddt

Apply What You Have Learned Today

- Next week you should:
  - Consult with your IT/IS teams. Consider taking advantage of Honeypots and scalability with DevOps Tool Chains
- In the first three months following this presentation you should:
  - Deploy honeypots within your networks using unused IP space.
  - Consider using the DDT as a guide to have your IT/IS staff implement honeypots with a mixture of static (traditional) and dynamic instances
- Within six months you should:
  - Integrate data collected by your new honeypots into your threat intelligence feeds, and possibly be creating real-time security controls based on this intelligence
  - Consider sharing your threat intelligence with the larger community, at least in terms of IP sources and other indicators of compromise

THANKS FOR ATTENDING!

QUESTIONS?