Twisted Haystack: Protecting Industrial Systems with Dynamic Deception

Lane Thames, PhD
10/3/2018
Who Am I?

Lane Thames

Vulnerability and Exposure Research Team (Vert)

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 with out computing future
  • New solution opportunities
• Deception & Dynamic Deception
• Implementing Dynamic Deception with Python and Twisted
• Scalable Dynamic Deception with Containers
• Conclusions and Future Work
From Cloud Evolution to the Internet of Things

Fog/Edge Computing

Cloud Computing
- IaaS
- PaaS
- SaaS

Containers
- BaaS
- FaaS

Microservices, Serverless Computing
Industrial IoT
Internet Connectivity: An Inevitability

Data Source: https://icsmap.shodan.io/
Cloud-based Design & Manufacturing (CBDM)

**Internet of Services**
- Ideation
- Design, CAD
- FEM Analysis
- Tool Path Generation
- Simulation
- Data Storage
- Rapid Prototyping
- Manufacturing
- Logistics

**Industrial Internet of Things**
- CAD Systems
- CAE Systems
- 3D Printers
- Lathes
- Milling machines
- Production Facilities
- Transport hubs

**THINGS realize bespoke SERVICES**
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

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?

Can we avoid this from happening again?
From Cloud Evolution to the Internet of Things

Fog/Edge Computing

Cloud Computing

IaaS  PaaS  SaaS

Containers

BaaS  FaaS

Microservices, Serverless Computing
Complexity & Chaos Anyone -- Everyone ??
Complexity & Chaos Anyone -- Everyone ??

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

dec·e·ive
/deˈsēv/

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

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
- DevOps Tool Chains

- Insider Threat
  - Lateral movement within after 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

The OSI Model

7 Application
6 Presentation
5 Session
4 Transport
3 Network
2 Link
1 Physical

The TCP/IP Model

<table>
<thead>
<tr>
<th>Segment</th>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td></td>
<td>DNS, FTP, HTTP, HTTPS, POP, SMTP, SSH, etc…</td>
</tr>
<tr>
<td>Transport</td>
<td></td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>Internet</td>
<td></td>
<td>IP Address: IPv4, IPv6</td>
</tr>
<tr>
<td>Network Access</td>
<td></td>
<td>MAC Address</td>
</tr>
<tr>
<td>Physical</td>
<td></td>
<td>Ethernet cable, fibre, wireless, coax, etc…</td>
</tr>
</tbody>
</table>

Public

Private

A5 Supervisory Control
- Fault Administration
- Set-point optimization

A4 Fault Detection and Isolation
- Data Reconciliation

A3 Control Network
- PLC i-1 Field Area Network
- PLC i Field Area Network
- PLC i+1 Field Area Network

A2 PLC i
- A0 Pool i
- A0 Pool i+1

A1 Field Area Network
- Sensors
- Actuators
- Offtake

This image is part of the Bioinformatics Web Development tutorial at http://www.cellbio.com/bioinformatics_web_development/ © cellbio.com, all rights reserved.
Deception-based Cybersecurity

Port dynamics

```
#!/usr/bin/env python
#
import socket
import random

server = None
resp = "HTTP/1.1 200 OK
Connection: close"

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 'RECV FROM: %s' % str(address)
        client.send(resp)
        client.close()
        server.shutdown(socket.SHUT_RDWR)
        server.close()
        server = None
        break
```
Deception-based Cybersecurity

Port dynamics
Deception-based Cybersecurity

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

class SimpleTCPServer((SocketServer, BaseRequestHandler):
    # Must implement this function
def handle(self):
        try:
            self.request.sendall('Welcome to

class SimpleThreadServer((SocketServer, ThreadingMixIn, SocketServer.TCPServer):
    def __init__(self, port):
        self._host = socket.gethostname()
        self.port = port
        self.allow_reuse_address=True
        self.server = SimpleTCPServer((self._host, self.port), SimpleTCPServer)
        self_thread = threading.Thread(target=self.serve_forever, daemon=True)
        self_thread.start()
        except Exception, e:
            print 'Error creating server. Exception: %s %s' % (str(e)

    def start(self):
        pass

    def shutdown(self):
        pass

    def spin_up()
        time.sleep(15)
        spin_down(servers)
```
Deception-based Cybersecurity

Port dynamics

```
root@thames-digio:/ics-dyndec# python simple-multi-port-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@thames-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 67.205.167.168:8294 0.0.0.0:* LISTEN
 tcp 0 0 67.205.167.168:8153 0.0.0.0:* LISTEN
 tcp 0 0 :::22 :::* LISTEN

root@thames-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@thames-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 honeypots
    • IoT based protocol projects
    • OT based protocol projects
Deception-based Cybersecurity

Port dynamics

- Wait for Events
- Reactor Loop
- A socket is ready
- Twisted code
- reader.doRead()
- Our code
Deception-based Cybersecurity

Port dynamics

```
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()
```
Deception-based Cybersecurity

Port 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 %d" % (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()
```
Deception-based Cybersecurity

Port dynamics
Deception-based Cybersecurity

Port 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 @ %d" % self.port
        reactor.listenTCP(self.port, self.factory)

def rrun(servers):
    print "\n\nResuming 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

```python
#!/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 run(Factory):
    reactor.removeAll()
    port = random.randrange(500, 599)
    print("Listening @ %s & port" % port)
    reactor.listenTCP(port, Factory)
    reactor.callLater(10, run, Factory)

store = ModbusSlaveContext()
bi = ModbusSequentialDataBlock(0, [17]*100)
co = ModbusSequentialDataBlock(0, [17]*100)
hI = ModbusSequentialDataBlock(0, [17]*100)
framer = ModbusSocketFramer
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'

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

Fog/Edge Computing

Cloud Computing
- IaaS
- PaaS
- SaaS

Containers
- BaaS
- FaaS

Microservices, Serverless Computing
What are Containers

Containerized Applications

App A  App B  App C  App D  App E  App F

Docker

Host Operating System

Infrastructure

Virtual Machine

App A

Guest Operating System

Virtual Machine

App B

Guest Operating System

Virtual Machine

App C

Guest Operating System

Hypervisor

Infrastructure

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

```
FROM python:2.7-slim

# Set the working directory to /app
WORKDIR /app

# Copy the current directory contents into the container at /app
ADD . /app

# Install Dependencies
RUN apt-get update && apt-get install -y gcc

# Install any needed packages specified in requirements.txt
RUN pip install --trusted-host pypi.python.org -r requirements.txt

# Make port 80 available to the world outside this container
EXPOSE 80

# Run twisted-web.py when the container launches
CMD ['python', 'twisted-web.py']
```
docker run -d -p 8080:80 jthames2/thddt-web
Solution

- Let’s look at the code
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](https://github.com/jlthames2/ddt)
    - Original Project
  - [https://github.com/jlthames2/thddt](https://github.com/jlthames2/thddt)
    - New project with containerized apps (pull from docker hub or build your own containers with my Dockerfiles)
Thank you
lthames@tripwire.com
@Lane_Thames