Implementing Security Control Loops in Security Autonomous Response Networks

Hristo Dimitrov

SNE University of Amsterdam & TNO
Supervisors: Marc X. Makkes & Robert J. Meijer

July 3, 2014

Introduction

Imagine your banking website or application does not work!

Introduction

Imagine your banking website or application does not work!

ANNOYING!!!

Introduction

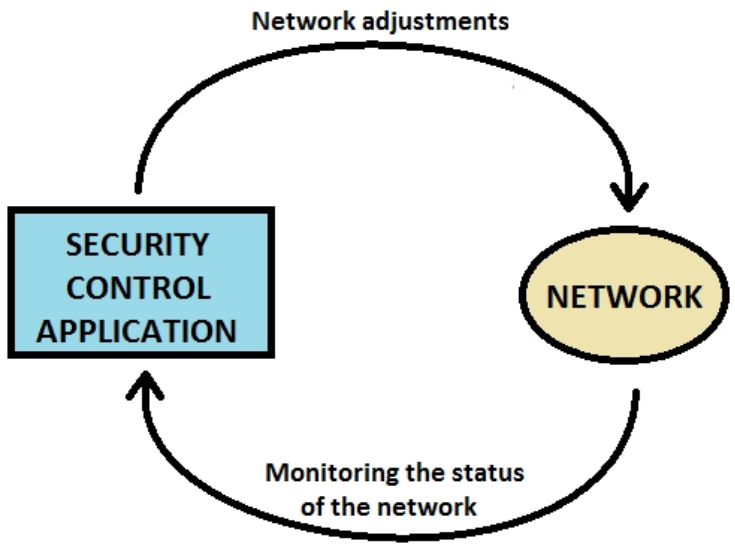
Imagine your banking website or application does not work!

ANNOYING!!!

- A way for adopting **the best countermeasures technologies** which are available
- Support for very **complex networks**
- Easier **organizing the security** of company networks
- **Faster response** times

Introduction

- **Software Defined Networks (SDNs)** are out there...
- Implementing **Security as a Service (SaaS)**
- By using **control loops**
- **Share security modules** with other companies and organizations



Introduction

Security Autonomous Response Networks - Software Defined Networks that adjust themselves in order take care of security threats and risks

Research Questions

How could a security control loop be implemented as a software solution?

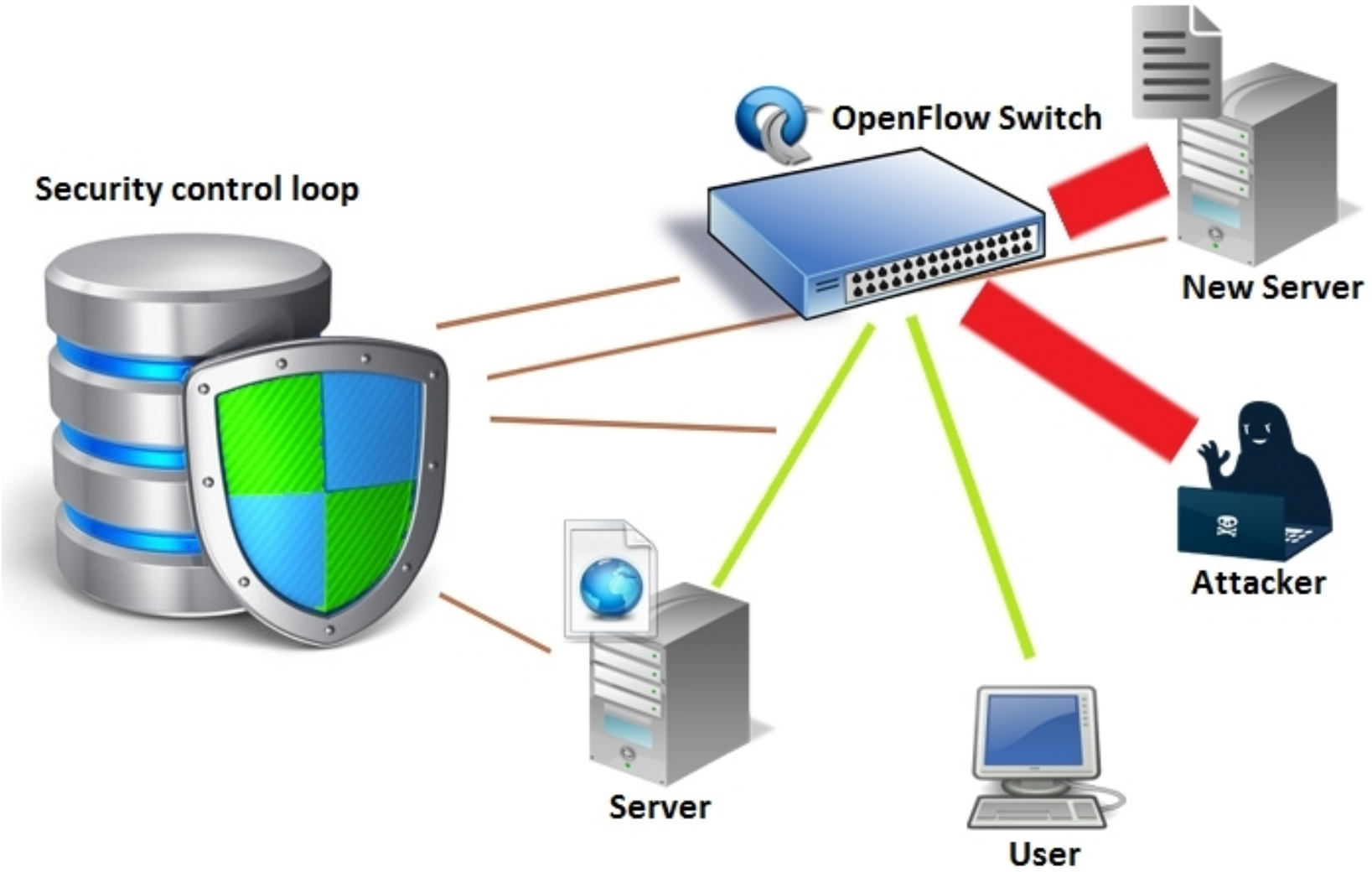
- *What properties should the implementation of a Security Autonomous Response Network have, in order to make it beneficial and effective against security threats?*
- *How can a Security Autonomous Response Network decide on which response will be better to execute in a given situation?*

Attack Isolation Control Loop

Attack Isolation Control Loop

Attack Isolation Control Loop

Attack Isolation Control Loop



Attack Isolation Control Loop

- Creating topology
- Testing the Network
- Start Services
- Start Control Loop
  - Collect TCP Connections Statistics
  - Check Number Of Connections
  - (Determine Potential Attacks)
  - **(Create New Server)**
  - **(Redirect Traffic To It)**

```
#check for attacks
if dos == True :
    #Define attributes
    counter +=1
    print "Counter:", counter
    hosts[counter] = "nh%s" % counter
    print "Host:", hosts[counter]
    hostips[hosts[counter]] = "10.0.0.%s" % (n+counter)
    print "IP:", hostips[hosts[counter]]
    hostints[hosts[counter]] = "%s-eth0" % hosts[counter]
    print "Host interface:", hostints[hosts[counter]]
    switchints[hosts[counter]] = "s1-eth%s" % (n+counter)
    print "Switch interface", switchints[hosts[counter]]

    #Create new host and redirect the old one
    print h1.cmd( "kill -9", fileserverpid)
    h = net.addHost( hosts[counter] , cpu=1/8 )
    time.sleep(2)
    net.addLink( h, s1, **distrlinkopts )
    s1.attach(switchints[hosts[counter]])
    print h.cmd( "ifconfig", hostints[hosts[counter]]
                , hostips[hosts[counter]] )
    print "Redirecting now..."
    print h1.cmd( "~/mininet/examples/redirect.py %s &"
                % hostips[hosts[counter]] )
    print "Redirected!"
    print h.cmd( 'cd ~/fileserver/')
    print h.cmd( 'python -m SimpleHTTPServer 8000 > /dev/null 2>&1 &')
    #Test the newly created host
    print h2.cmd( 'cd ~')
    print "h2 wget http://%s:8000/test_10K.img" % (h1.IP())
    print "h2 time curl http://%s:8001/index.html" % (h1.IP())
```

Moving resources to new server

Attack Limiting Control Loop

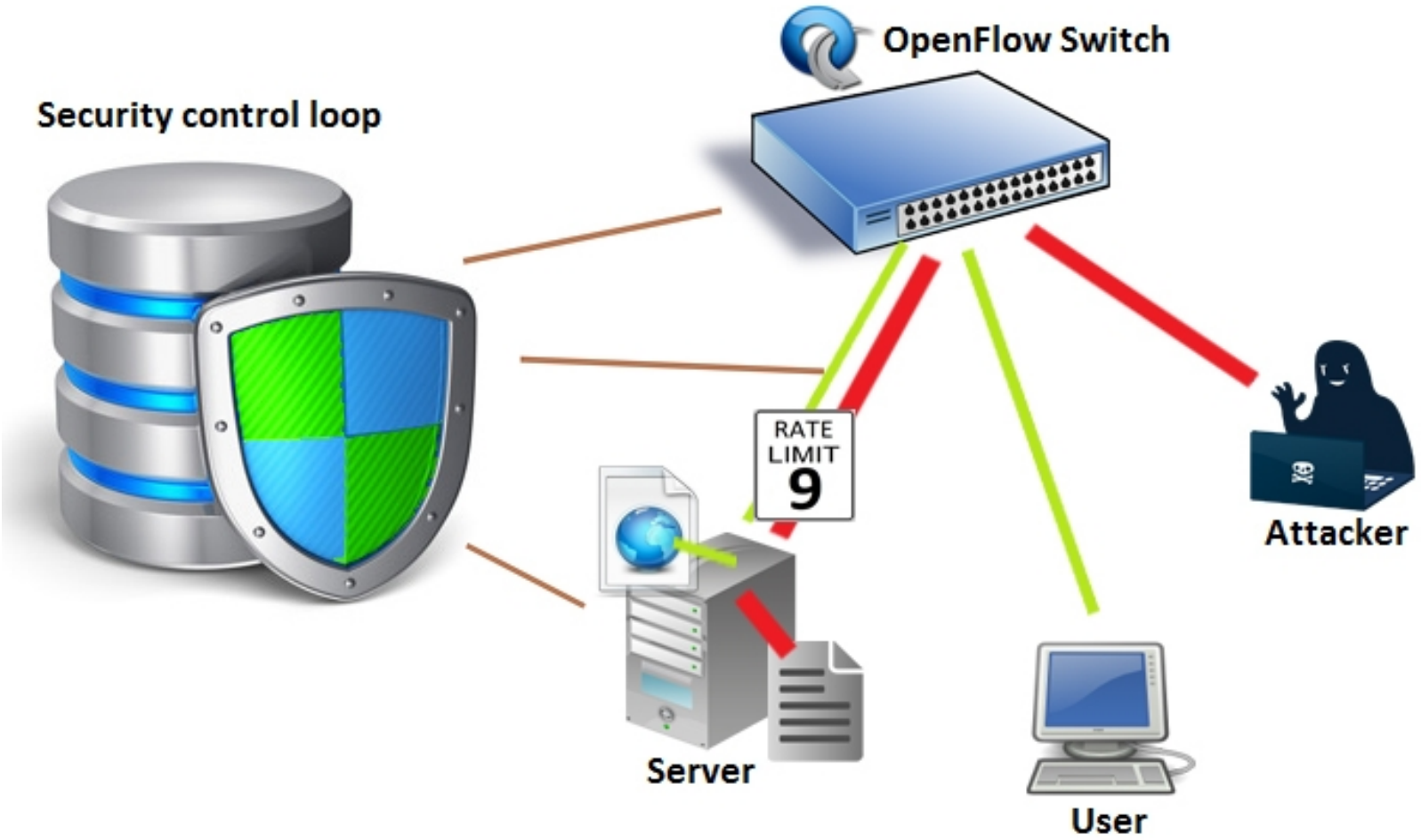
Attack Limiting Control Loop

Attack Limiting Control Loop

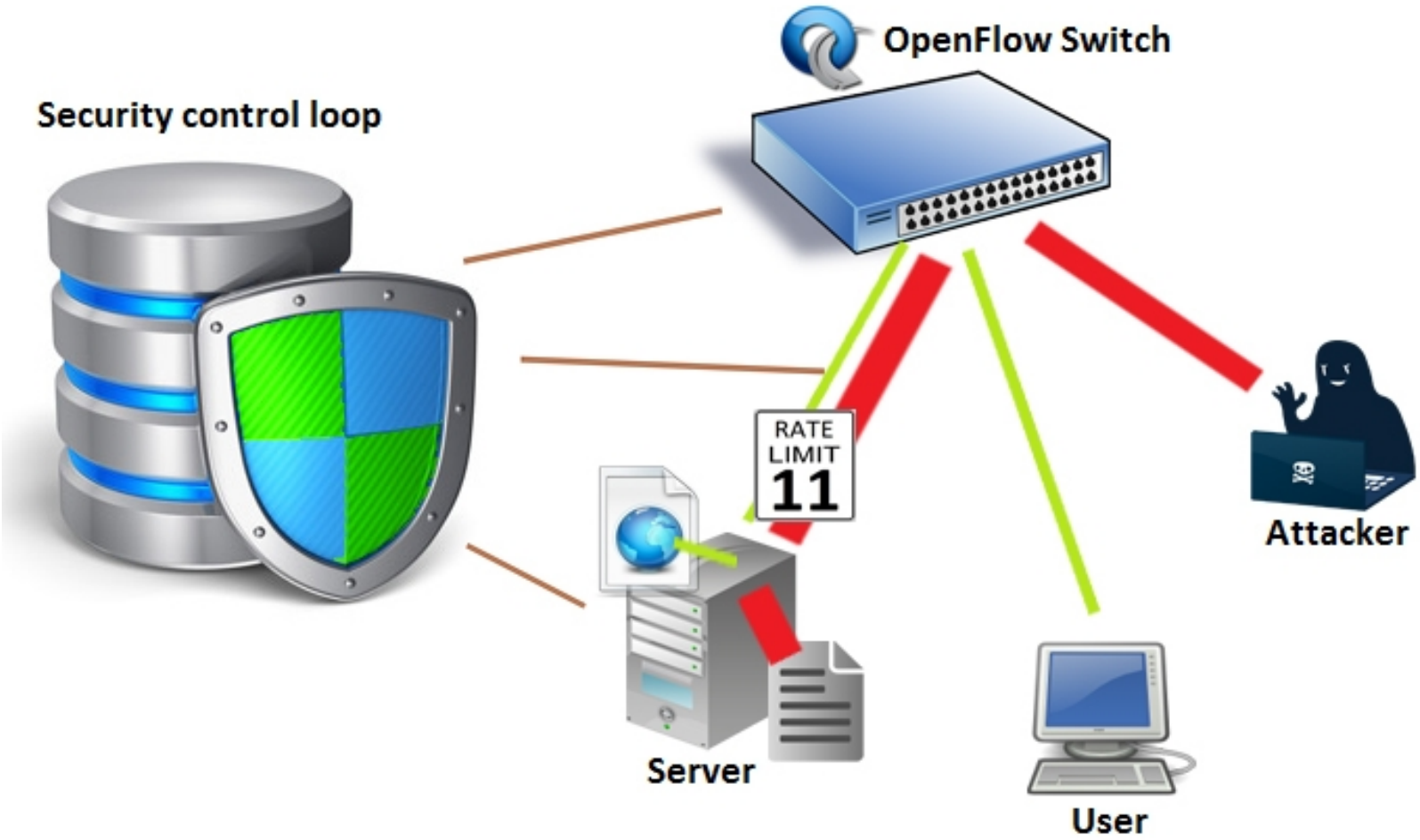
Attack Limiting Control Loop

Attack Limiting Control Loop

Attack Limiting Control Loop



Attack Limiting Control Loop



Attack Limiting Control Loop

- Creating topology
- Testing the Network
- Start Services
- Start Control Loop
  - Collect TCP Connections Statistics
  - Check Number Of Connections
  - **(Determine Potential Attacks)**
  - (Collect Bandwidth Statistics)
  - (Adjust Rate Limits)
  - (Implement New Rate Limits)

```
print "Determining potential attack vectors..."
attsrcip = ""
attdstipport = ""
attsrcips = {}
attdstipports = {}
if ncon > 10 :
    for i in range(1, (ncon+1)):
        if results[i].split()[2] == "tcp" :
            attdstipport = results[i].split()[3]
            attsrcip = results[i].split()[5].split(":")[0]
            if attsrcips.has_key(attsrcip):
                attsrcips[attsrcip] += 1
            else:
                attsrcips[attsrcip] = 1
            if attdstipports.has_key(attdstipport):
                attdstipports[attdstipport] += 1
            else:
                attdstipports[attdstipport] = 1
    print "Destinations:", attdstipports
    print "Sources:", attsrcips
    asi = attsrcips.keys()
    attsrcip = asi[0]
    for i in range(1, len(asi)):
        if attsrcips[asi[i]] > attsrcips[attsrcip]:
            attsrcip = asi[i]
    adip = attdstipports.keys()
    attdstipport = adip[0]
    for i in range(1, len(adip)):
        if attdstipports[adip[i]] > attdstipports[attdstipport]:
            attdstipport = adip[i]
```

Determine potential attacks vectors

Attack Isolation Results

Attack Limiting Results

Conclusions

(What properties should the implementation of a Security Autonomous Response Network have, in order to make it beneficial and effective against security threats?)

- **Software Modularity** - Scalability, Reusable and pluggable modules
- **Company Infrastructure Modularity** - Flexibility, More options for responses to security threats

Conclusions

(How can a Security Autonomous Response Network decide on which response will be better to execute in a given situation?)

Responses to security threats should be:

- **Classified** - based on which problems they can solve
- **Rated** - based on their effectiveness

Questions

Please ask your questions now, thank you!