Very hard to even do a replay attack on a controller do to TCP sequence numbers (will just drop all previous flows) so if encrypted and can not break encryption, replay attack difficult.

(Good explanation here: <http://packetlife.net/blog/2010/jun/7/understanding-tcp-sequence-acknowledgment-numbers/> )

Maybe we focus solely on a switch that is compromised (i.e. you have a switch's private key and a controller's public key)

One secondary effect is we no longer need to mess with encryption since we are assuming we can correctly represent a single switch.

Our thesis: In an SDN environment, compromising a single switch can collapse your entire network

Part 1: Research on compromising switches (demonstrate switches currently have vulnerabilities that allow a signal switch to be compromised):

<http://www.infoworld.com/d/security/cisco-patches-vulnerabilities-in-some-security-appliances-switches-and-routers-228551>

<http://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20120711-ctms>

Part 2: Ways to effect the controller:

Overload controller: What types of messages can the switch send to the controller to repeatedly to effect it?

Can the switch misrepresent traffic to get more data (i.e. send that mac address xxx belong to him when it does not)?

First --- Comments on this? Any issues? I think simplifying the project is good. We are honestly looking for a 2 page write up so this makes thing easier.

So what does our test set-up look like?

--- I brain stormed about this for a bit and decided the easiest thing to do was not to steal a TCP connection from an existing switch, but rather to make our own switch in python. I have wiresharked the TCP handshake between a controller and mininet (simply start your controller, start wireshark (filter = of), and then run sudo mn). I am including the wireshark of a single session in 4 formats - use whatever is easier for you. The entire handshake is very small (approx 10 packet sends from the switch). There then is a periodic liveness poll that we need to respond to. Each time I believe the packet contents are the same - I am hoping the only difference is the seq/ack numbers.

Processing forward:

Step 1: Record exact packets from switch to a file

     -- I wrote (found) sniffer.py to do this. I am having a time looking (making readable) the data contents. I write out packets from port=6633 to a file, but the file is empty in some cases.

Step 2: Create simple socket to send messages to controller (done)

Step 3: Replay same responses as real switch to create controller

Step 4: Determine

Where I am currently:

I created two files. One is called sniffer.py.