

ENTRY // Xiongmai Security Research #6 // Fuzzing With Radamsa

[\[- \]](#) [\[X \]](#)

Date: 2019-09-04 Author: Ian Rash Index: #6

I'd never heard of this tool, until I happened to stumble upon it in a [LiveOverflow](#) video. The tool is, very neat to say the least. It takes an input "sample" data, and mutates it in various ways to cause potential bad behavior in an application.

From the previous behavioral fuzzing we did, we know what commands are available by default, so we should fuzz those specific items to see if we can get them to misbehave.

```
File.open("./rsrc/op_monitor.txt", "w+") do |file|
  file.print Command::OPMonitor.new(session_id: 0xabcdef00_u32).to_s
end
```

```
File.open("./logs/radamsa/op_monitor.log", "w+") do |file|
  puts "Testing connection"
  socket = MagicSocket.new("192.168.11.109", 34567)
  socket.login "default", Dahua.digest("tluafed")
  xmm = Command::OPMonitor.new
  socket.send_message xmm
  puts "SENT: #{xmm.message}"
  reply = socket.receive_message
  puts "GOT: #{reply.message}"
```

```
1000.times do |x|
  begin
    socket = MagicSocket.new("192.168.11.109", 34567)
    socket.login "default", Dahua.digest("tluafed")
    message = `radamsa ./rsrc/op_monitor.txt`
    file.puts "SENT: #{message.inspect}"
    socket.send message
    reply = socket.receive_message
    file.puts "GOT: #{reply.message.inspect}"
  rescue e : MagicError::SocketException
    puts "SOCKET DOWN! #{e.inspect}"
    raise e
  rescue e : MagicError::Exception
    file.puts "ERROR: #{e.inspect}"
    puts "ERROR: #{e.inspect}"
  rescue e
    file.puts "BAD ERROR: #{e.inspect}"
    puts "BAD ERROR: #{e.inspect}"
  end
end
end
```

I make a message for OPMonitor, and output it to a file, then send that file through Radamsa, and send it's fuzzing data to the camera. Within about 100 or so tries, it ended up finding a way to disrupt the client and the camera server for about 120 seconds while the camera reboots. This string takes the camera down via ping, connection, etc. This means the camera itself actually get rebooted.

◀ ▶

We can say for certain that there are some oddities and inconsistencies about how this protocol works, which tends to

 Fuzzing
  Security Research
  Exploit PoC Development