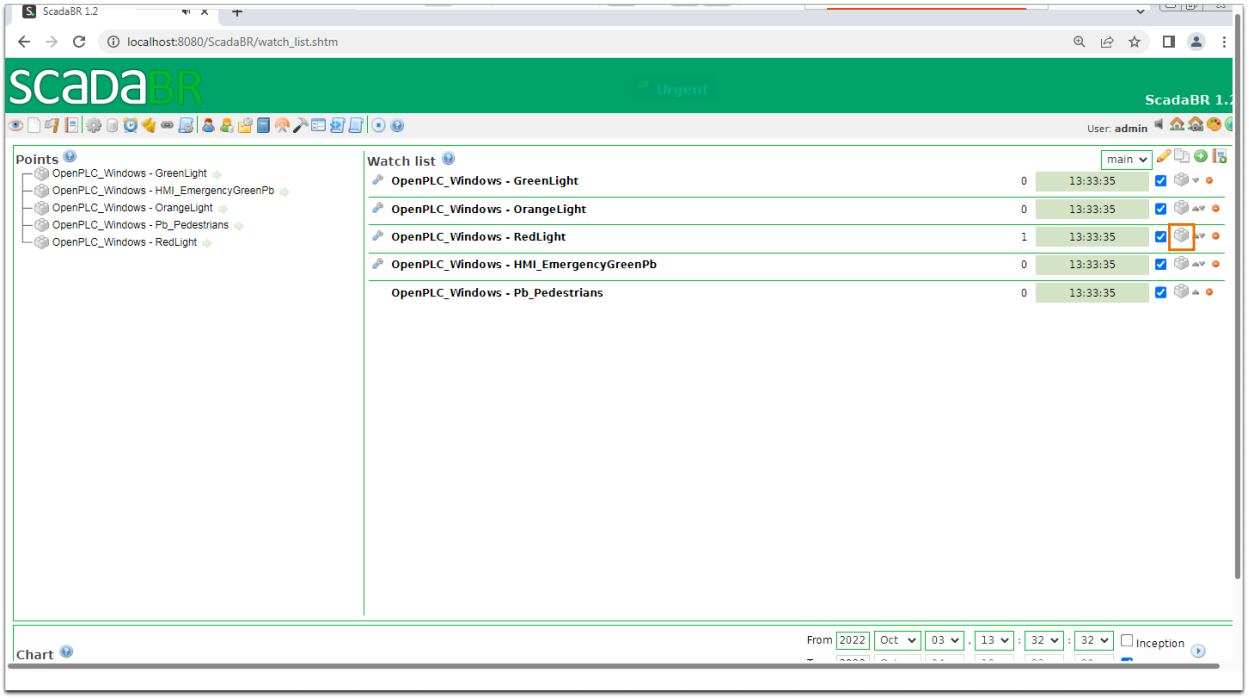
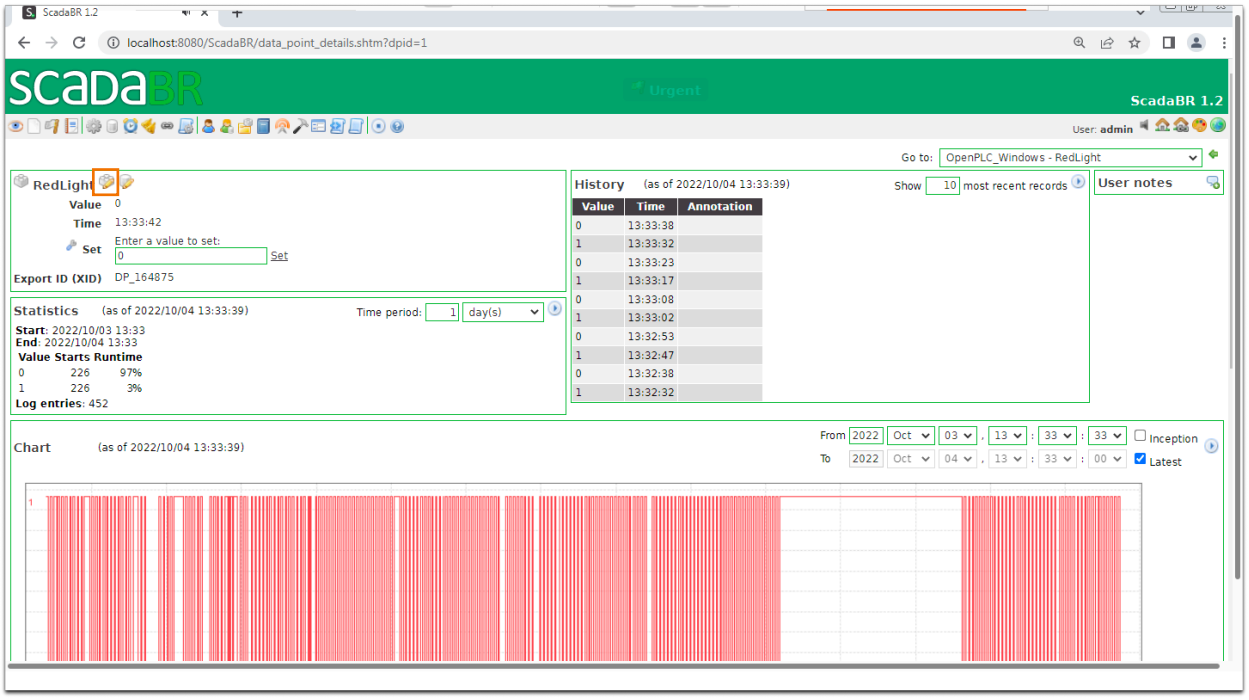
ScadaBR event handle

To showcase the result, I set up an experiment.
When the state of red light changes twice (0 -> 1, 1 -> 0) in 10 seconds, activate emergency (send 1 to 803). When the conditional not met, deactivate emergency (send 0 to 803).

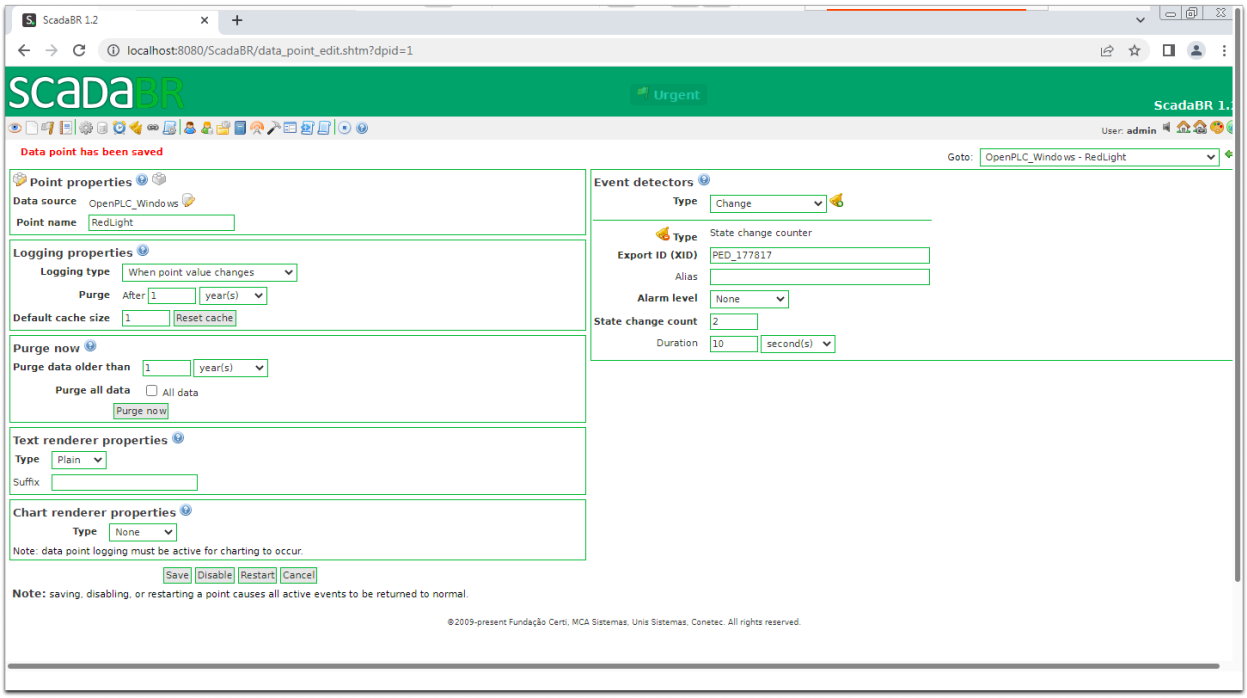
Click the orange box to show data point details.



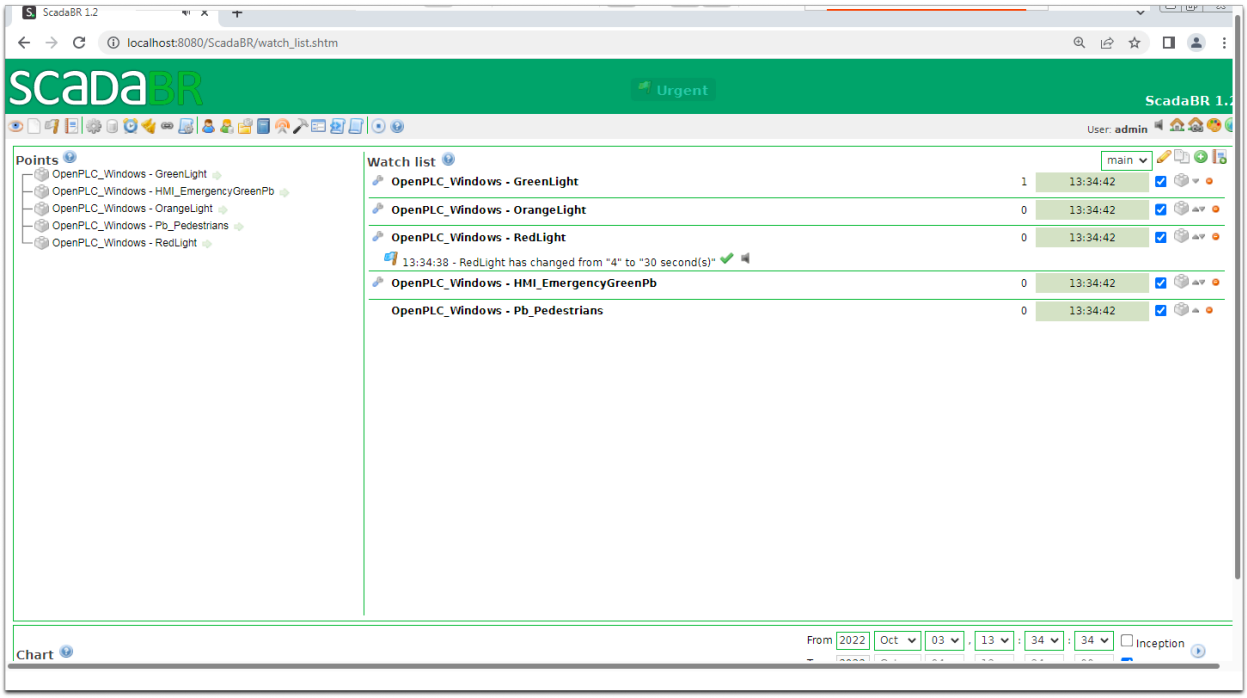
Click edit (orange box)



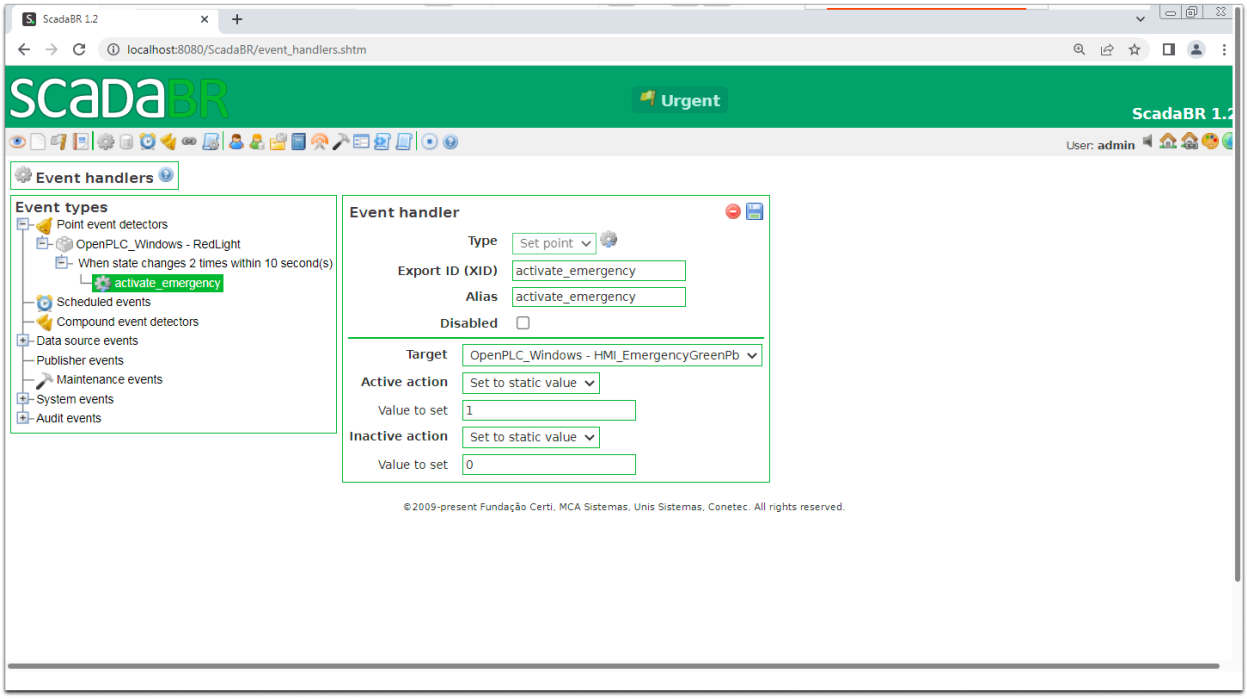
Configure event detector on the right hand side.



We can see alarm notification from the event on the watch list.



Set up event handler to react to corresponding event. In this case, set HMI_EmergencyGreenPb to 1 when active, set to 0 when inactive.



The pcap file is in google drive (event_detect_and_handle.pcapng). With Wireshark filter “ip.addr == 192.168.204.133 && tcp.port == 502”, we can see the time and corresponding lights below, and it aligns with what we expect.

```
5.87  sec red
11.90 sec not red
11.96 sec write 1
15.97 sec write 0
20.90 sec red
20.96 sec write 1
30.92 sec write 0
```

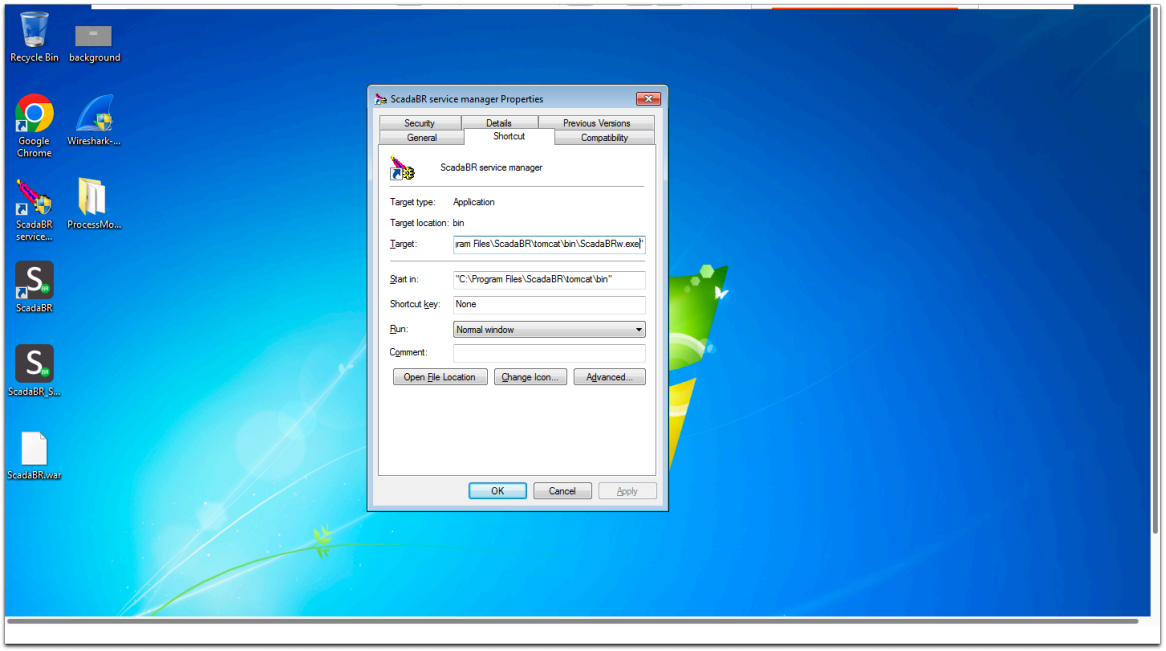
Dynamic Analysis

I used Process Monitor to monitor processes. The version 3.84 works on Windows 7. (https://web.archive.org/web/20210901132850/https://docs.microsoft.com/en-us/sysinternals/downloads/procmon)

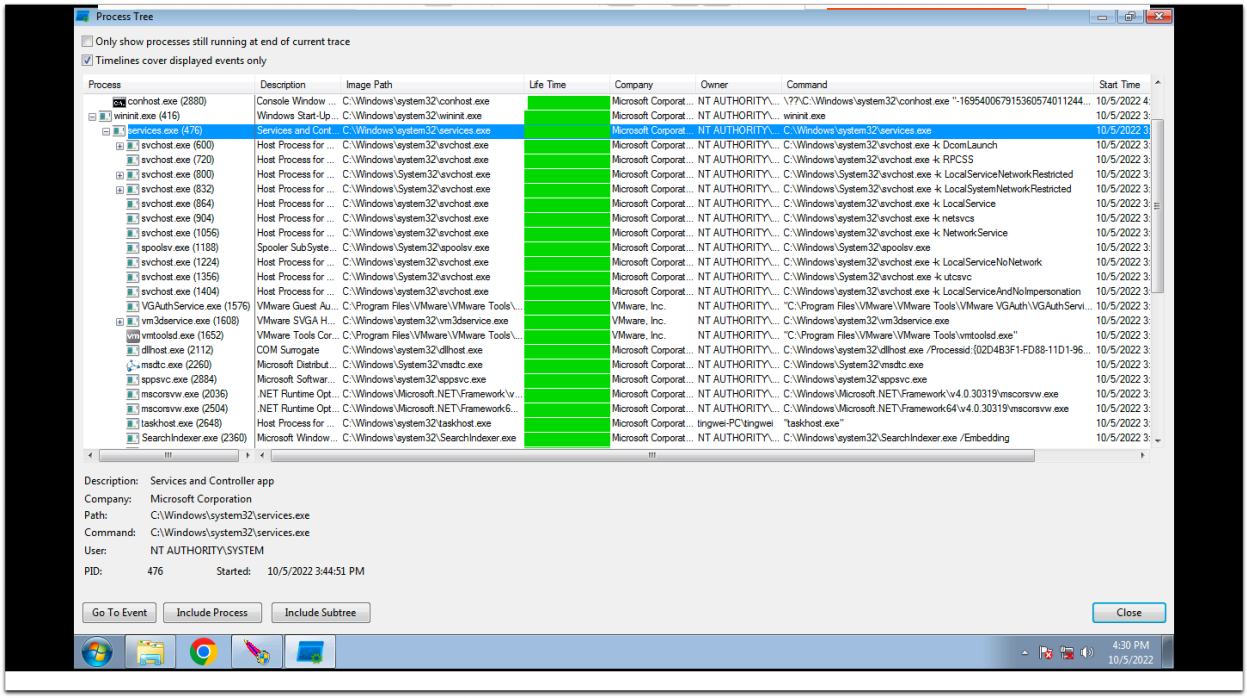
Observation on starting ScadaBR

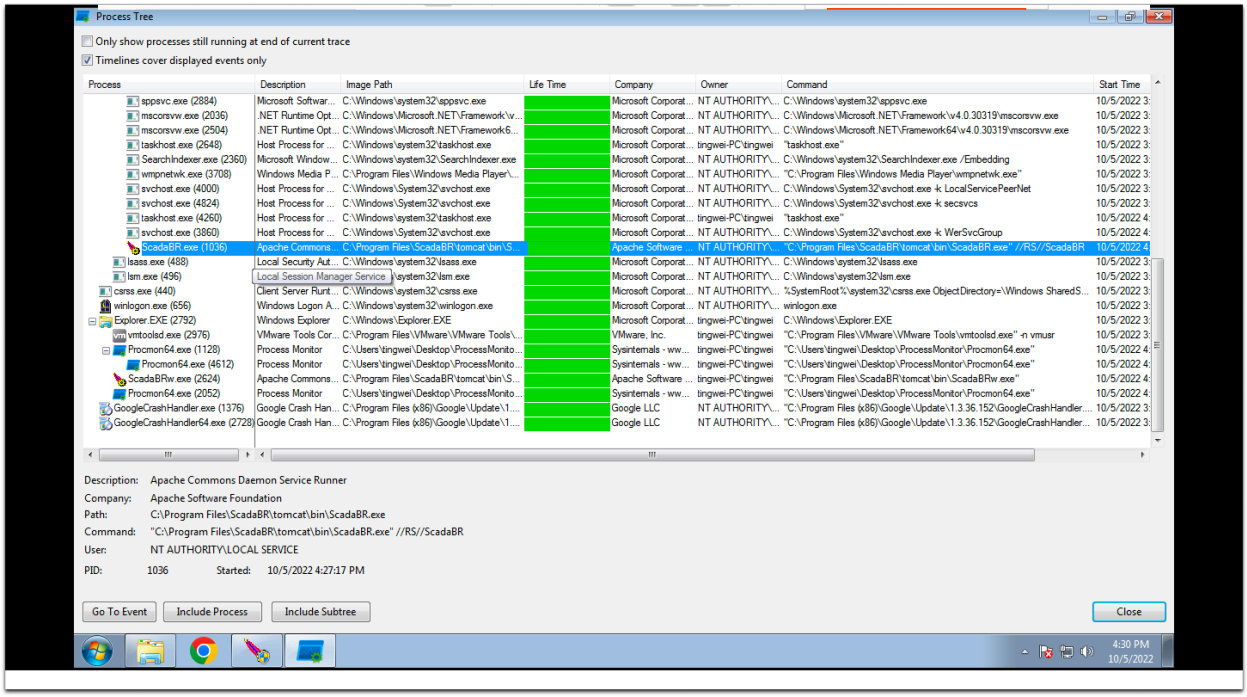
The log file is in google drive (StartScadaBRw.PML)

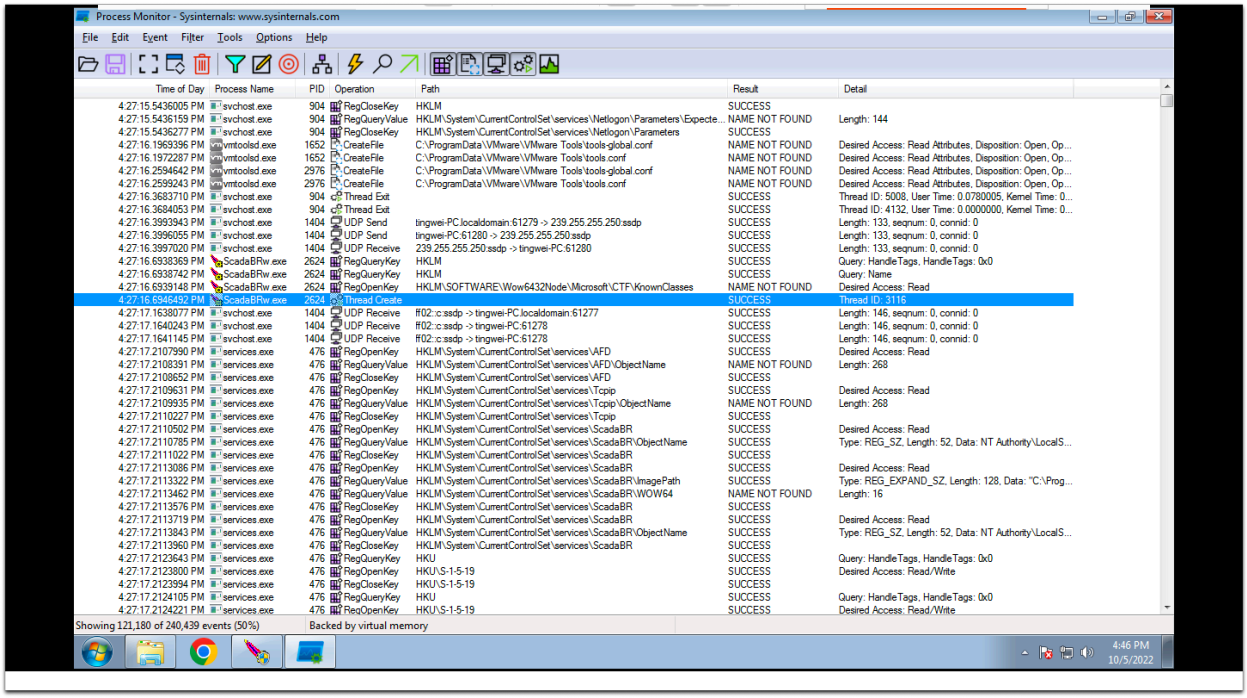
ScadaBR service manager is the shortcut of ScadaBRw.exe



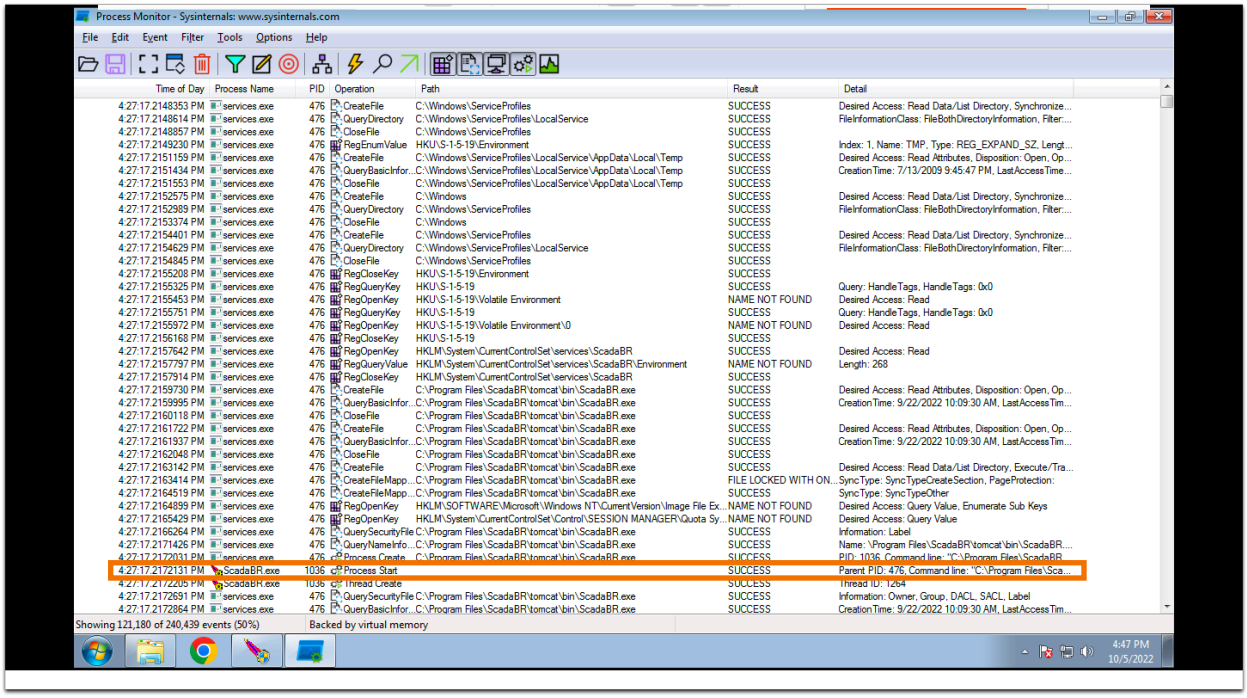
When we press “Start” in ScadaBRw, it
creates a thread and call services.exe to start “ScadaBR.exe”, which runs the tomcat Web App.







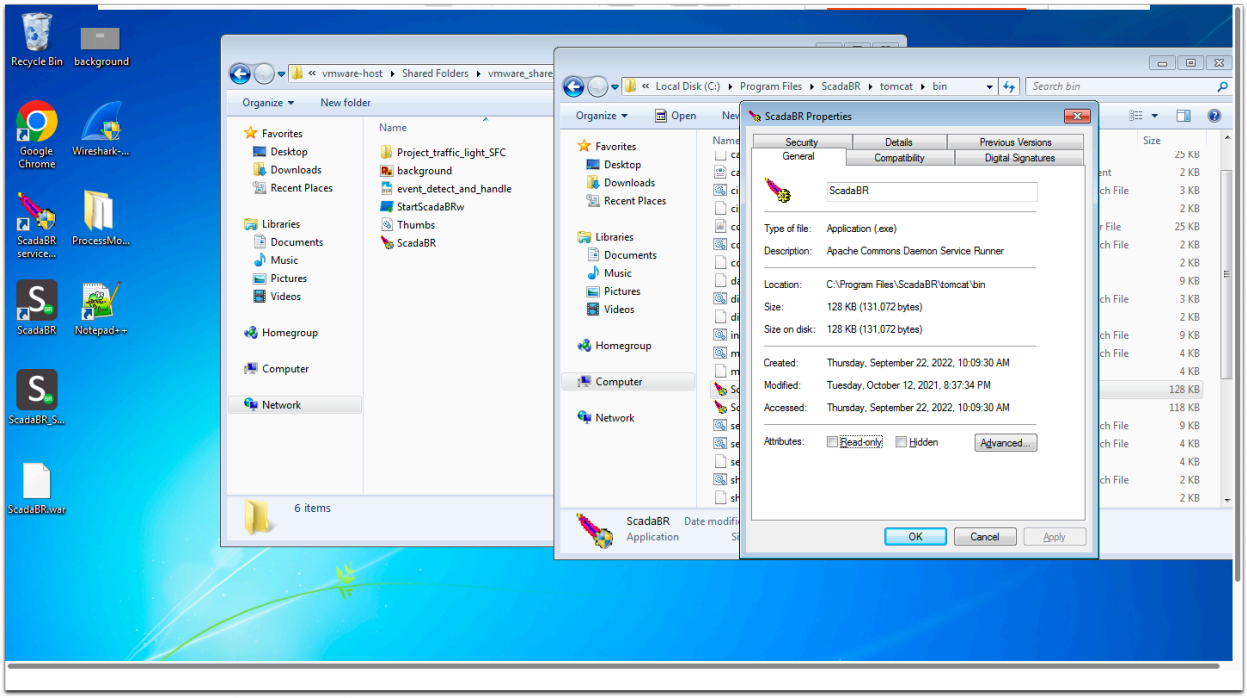
ScadaBR.exe starts



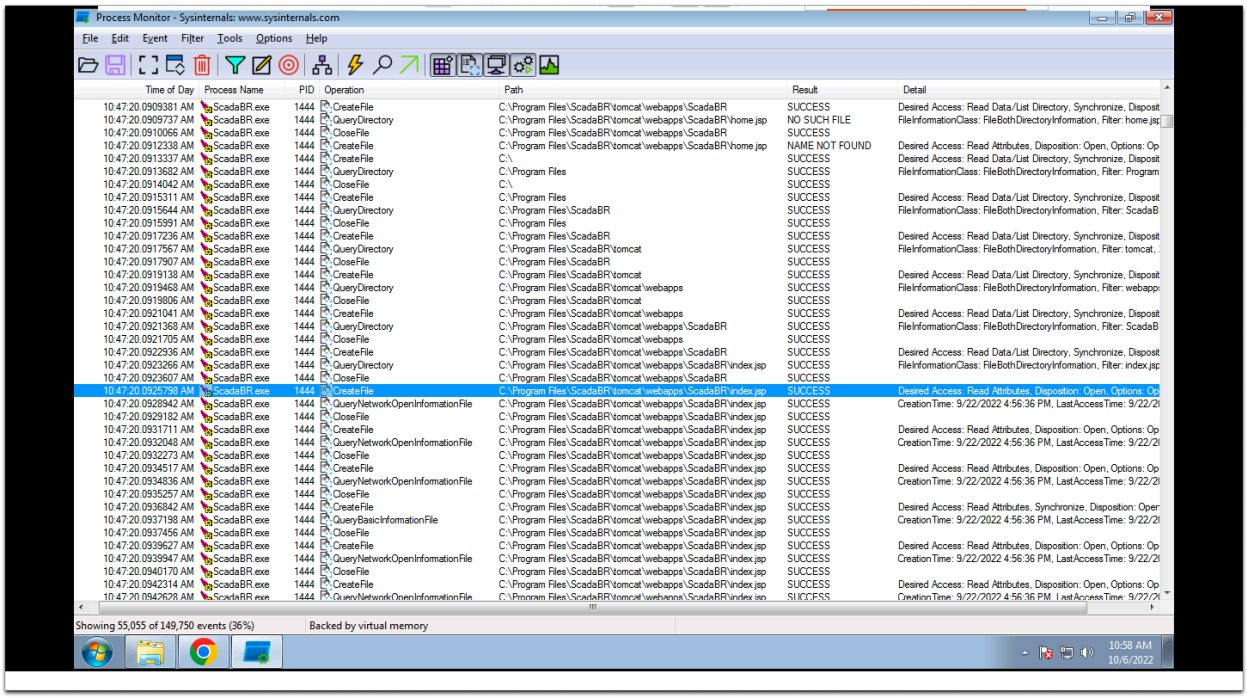
Observation after starting ScadaBR

The log file is in google drive (ScadaBR_Process.PML)

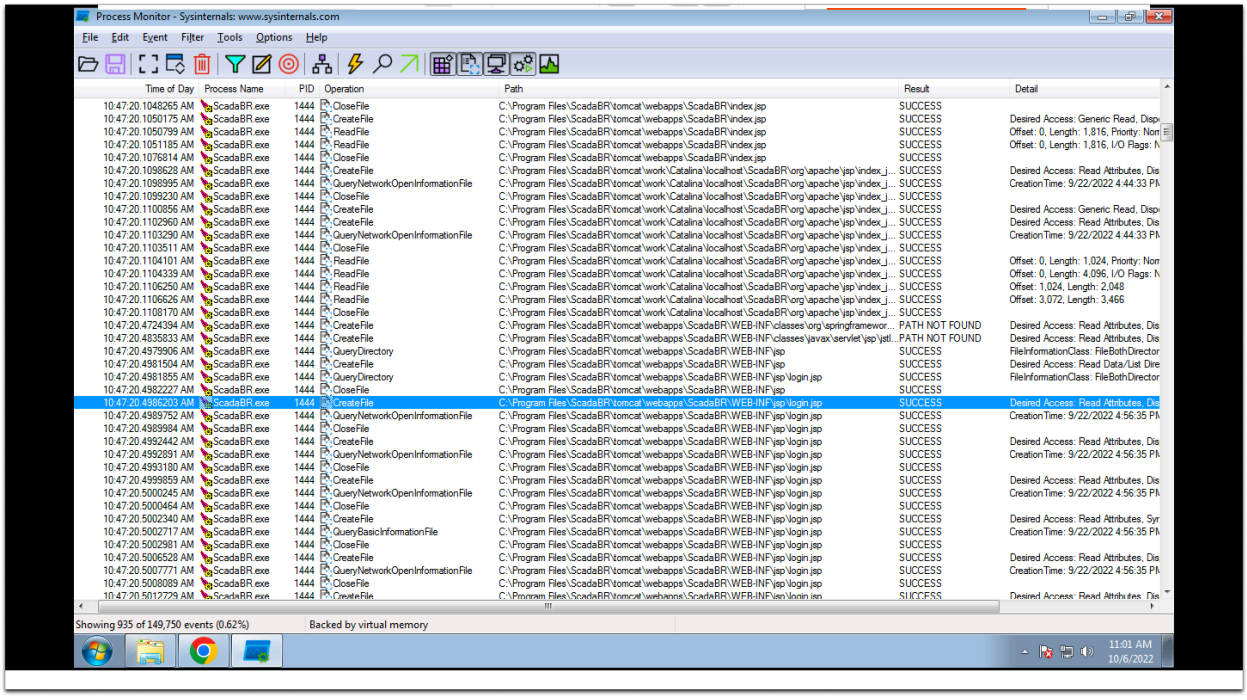
ScadaBR.exe is a service runner. Actions on ScadaBR are executed by jsp, jar, and java files.



Access index.jsp when open http://localhost:8080/ScadaBR

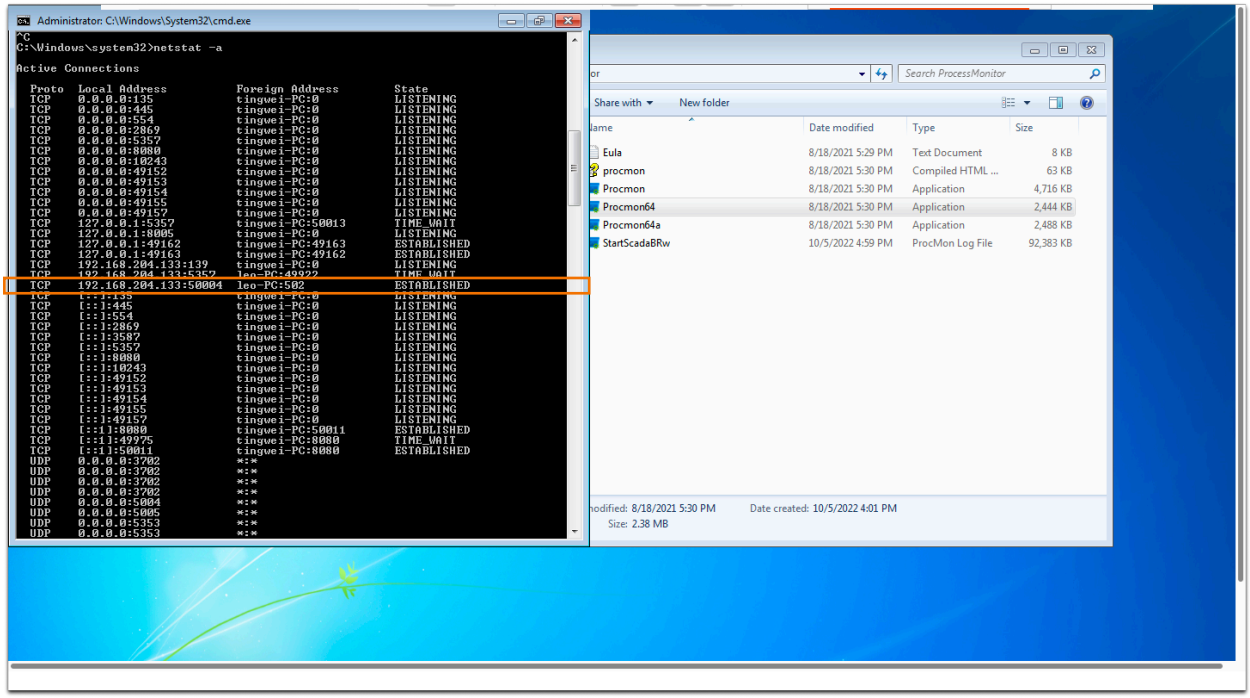


Access login.jsp to perform login.

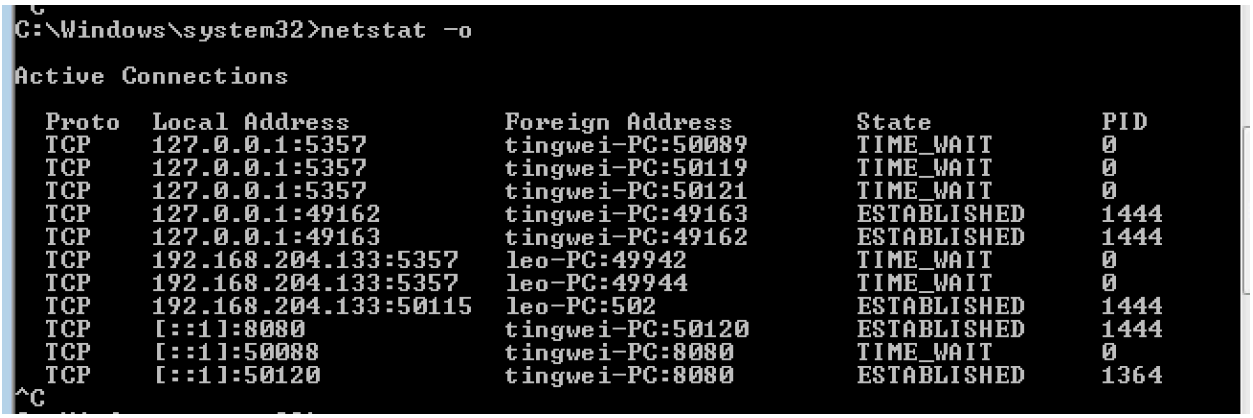


Also, there are watchList.jsp, dataSourceList.jsp, etc.

It only connects to modbus (port 502) on the other virtual machine running OpenPLC.

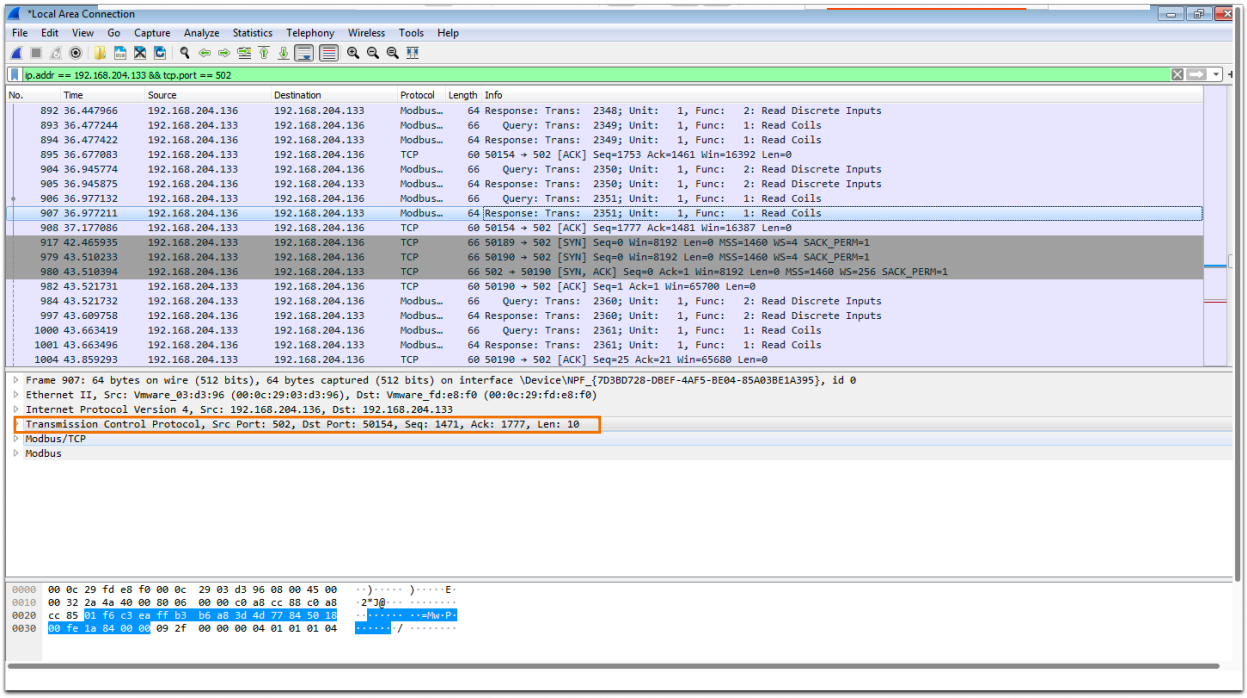


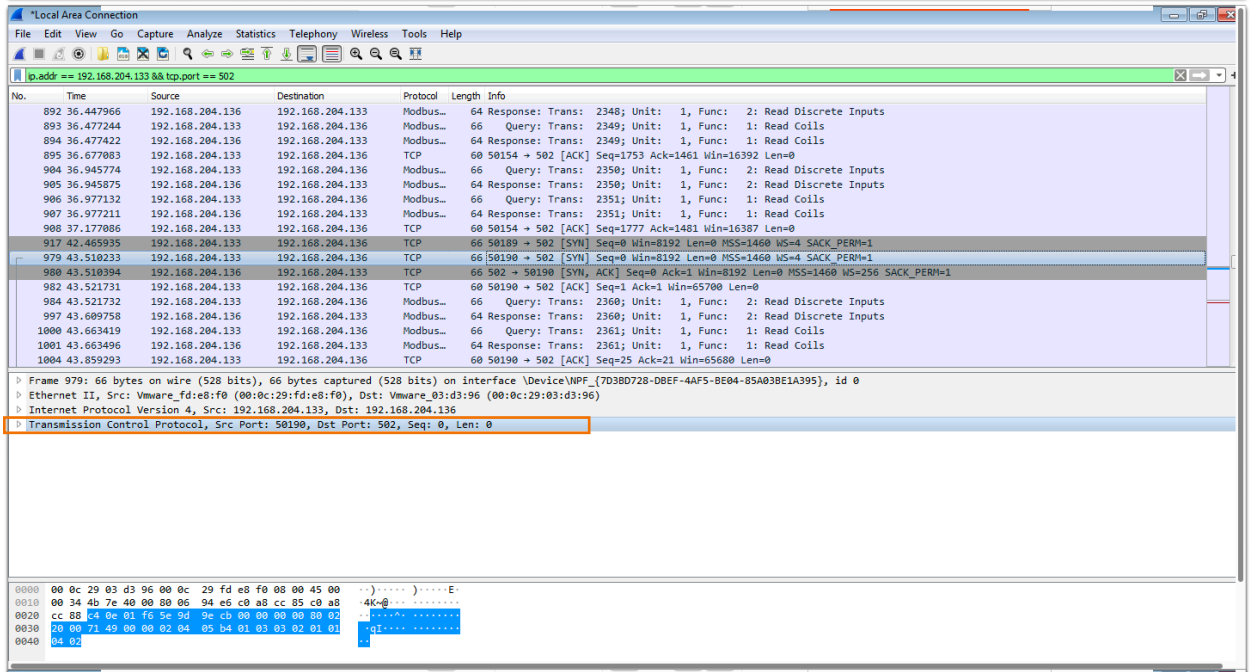
Port 502 is connected with PID 1444

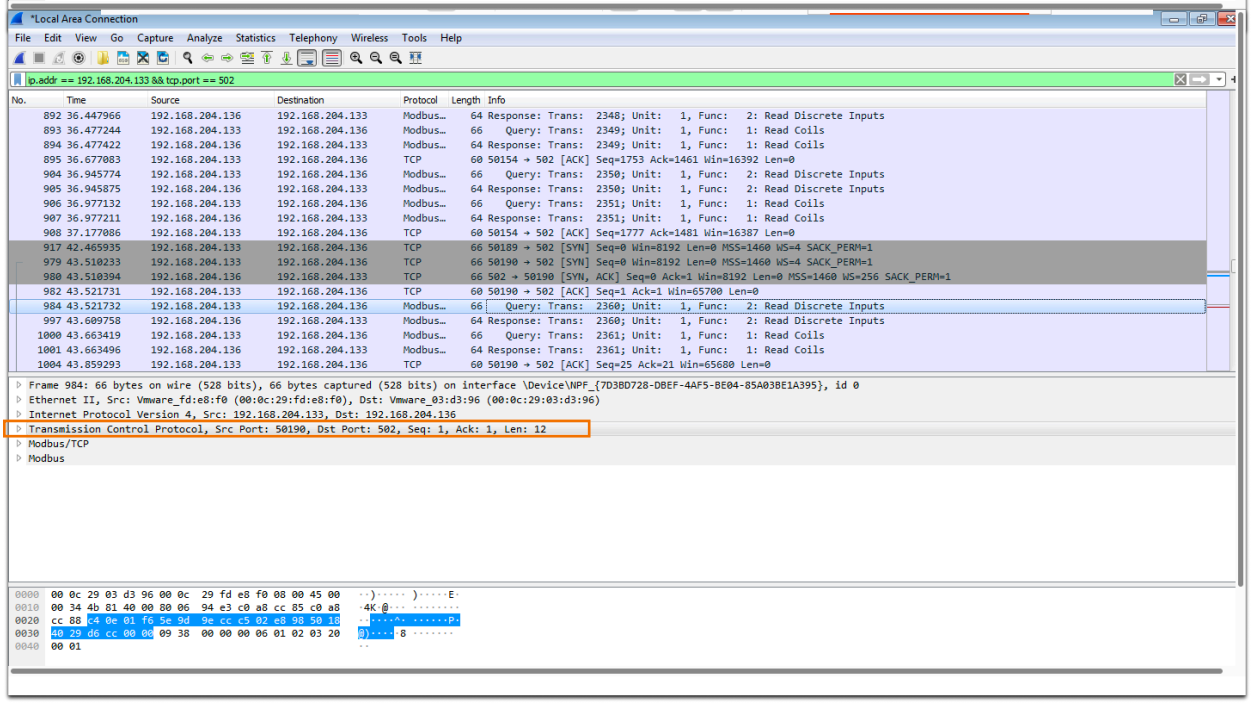


And PID 1444 is ScadaBR.exe

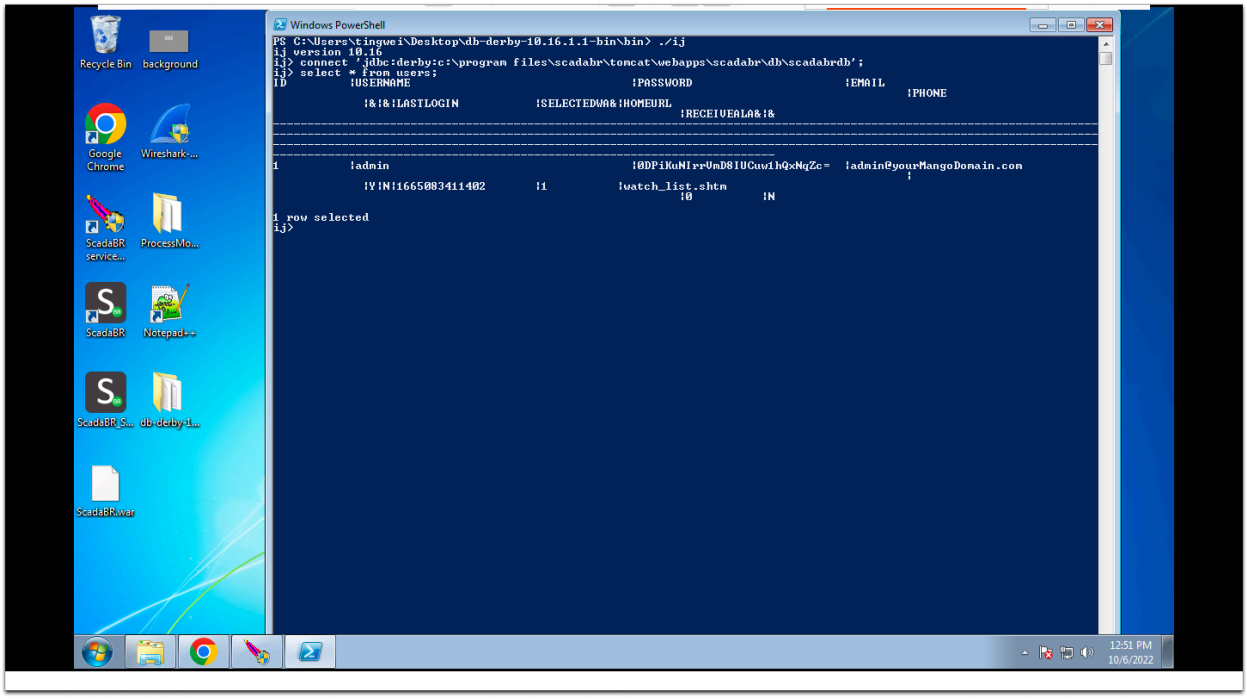
The source port of ScadaBR changes regularly. In this case, it changes from port 50154 to 50190.


*Local Area Connection
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.addr == 192.168.204.133 && tcp.port == 502
No. Time Source Destination Protocol Length Info
892 36.447966 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2348; Unit: 1, Func: 2: Read Discrete Inputs
893 36.477244 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2349; Unit: 1, Func: 1: Read Coils
894 36.477422 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2349; Unit: 1, Func: 1: Read Coils
895 36.677083 192.168.204.133 192.168.204.136 TCP 60 50154 → 502 [ACK] Seq=1753 Ack=1461 Win=16392 Len=0
904 36.945774 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2350; Unit: 1, Func: 2: Read Discrete Inputs
905 36.945875 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2350; Unit: 1, Func: 2: Read Discrete Inputs
906 36.977132 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2351; Unit: 1, Func: 1: Read Coils
907 36.977211 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2351; Unit: 1, Func: 1: Read Coils
908 37.177086 192.168.204.133 192.168.204.136 TCP 60 50154 → 502 [ACK] Seq=1777 Ack=1481 Win=16387 Len=0
917 42.465935 192.168.204.133 192.168.204.136 TCP 66 50189 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
979 43.510233 192.168.204.133 192.168.204.136 TCP 66 50190 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
980 43.510394 192.168.204.136 192.168.204.133 TCP 66 502 → 50190 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
982 43.521731 192.168.204.133 192.168.204.136 TCP 60 50190 → 502 [ACK] Seq=1 Ack=1 Win=65700 Len=0
984 43.521732 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2360; Unit: 1, Func: 2: Read Discrete Inputs
997 43.609758 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2360; Unit: 1, Func: 2: Read Discrete Inputs
1000 43.663419 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2361; Unit: 1, Func: 1: Read Coils
1001 43.663496 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2361; Unit: 1, Func: 1: Read Coils
1004 43.859293 192.168.204.133 192.168.204.136 TCP 60 50190 → 502 [ACK] Seq=25 Ack=21 Win=65680 Len=0
Frame 907: 64 bytes on wire (512 bits), 64 bytes captured (512 bits) on interface \Device\NPF_{7D3BD728-DBEF-4AF5-BE04-85A03BE1A395}, id 0
Ethernet II, Src: Vmware_03:d3:96 (00:0c:29:03:d3:96), Dst: Vmware_fd:e8:f0 (00:0c:29:fd:e8:f0)
Internet Protocol Version 4, Src: 192.168.204.136, Dst: 192.168.204.133
Transmission Control Protocol, Src Port: 502, Dst Port: 50154, Seq: 1471, Ack: 1777, Len: 10
Modbus/TCP
Modbus
0000 00 0c 29 fd e8 f0 00 0c 29 03 d3 96 08 00 45 00
0010 00 32 2a 4a 40 00 80 06 00 00 c0 a8 cc 88 c0 a8
0020 cc 85 01 f6 c3 ea ff b3 b6 a8 3d 4d 77 84 50 18
0030 00 fe 1a 84 00 00 09 2f 00 00 00 04 01 01 01 04

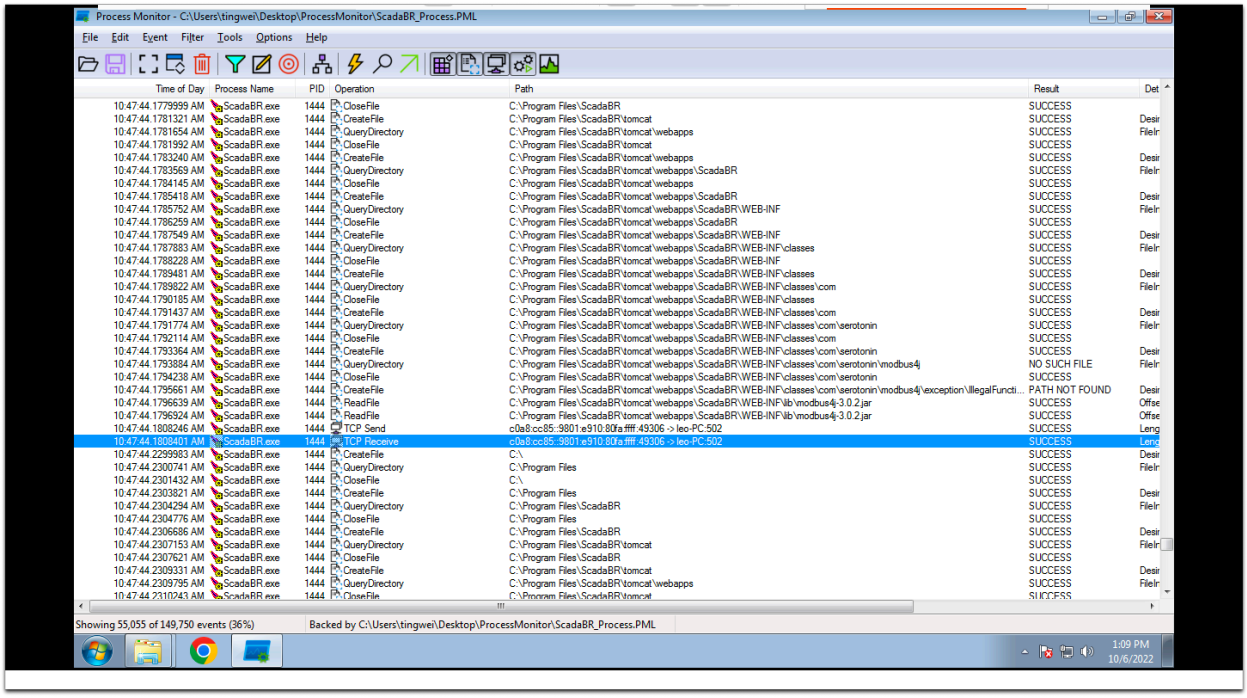

*Local Area Connection
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.addr == 192.168.204.133 && tcp.port == 502
No. Time Source Destination Protocol Length Info
892 36.447966 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2348; Unit: 1, Func: 2: Read Discrete Inputs
893 36.477244 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2349; Unit: 1, Func: 1: Read Coils
894 36.477422 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2349; Unit: 1, Func: 1: Read Coils
895 36.677083 192.168.204.133 192.168.204.136 TCP 60 50154 → 502 [ACK] Seq=1753 Ack=1461 Win=16392 Len=0
904 36.945774 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2350; Unit: 1, Func: 2: Read Discrete Inputs
905 36.945875 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2350; Unit: 1, Func: 2: Read Discrete Inputs
906 36.977132 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2351; Unit: 1, Func: 1: Read Coils
907 36.977211 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2351; Unit: 1, Func: 1: Read Coils
908 37.177086 192.168.204.133 192.168.204.136 TCP 60 50154 → 502 [ACK] Seq=1777 Ack=1481 Win=16387 Len=0
917 42.465935 192.168.204.133 192.168.204.136 TCP 66 50189 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
979 43.510233 192.168.204.133 192.168.204.136 TCP 66 50190 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
980 43.510394 192.168.204.136 192.168.204.133 TCP 66 502 → 50190 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
982 43.521731 192.168.204.133 192.168.204.136 TCP 60 50190 → 502 [ACK] Seq=1 Ack=1 Win=65700 Len=0
984 43.521732 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2360; Unit: 1, Func: 2: Read Discrete Inputs
997 43.609758 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2360; Unit: 1, Func: 2: Read Discrete Inputs
1000 43.663419 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2361; Unit: 1, Func: 1: Read Coils
1001 43.663496 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2361; Unit: 1, Func: 1: Read Coils
1004 43.859293 192.168.204.133 192.168.204.136 TCP 60 50190 → 502 [ACK] Seq=25 Ack=21 Win=65680 Len=0
Frame 979: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{7D3BD728-DBEF-4AF5-BE04-85A03BE1A395}, id 0
Ethernet II, Src: Vmware_fd:e8:f0 (00:0c:29:fd:e8:f0), Dst: Vmware_03:d3:96 (00:0c:29:03:d3:96)
Internet Protocol Version 4, Src: 192.168.204.133, Dst: 192.168.204.136
Transmission Control Protocol, Src Port: 50190, Dst Port: 502, Seq: 0, Len: 0
0000 00 0c 29 03 d3 96 00 0c 29 fd e8 f0 08 00 45 00
0010 00 34 4b 7e 40 00 80 06 94 e6 c0 a8 cc 85 c0 a8
0020 cc 88 c4 0e 01 f6 5e 9d 9e cb 00 00 00 00 80 02
0030 20 00 71 49 00 00 02 04 05 b4 01 03 03 02 01 01
0040 04 02


*Local Area Connection
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.addr == 192.168.204.133 && tcp.port == 502
No. Time Source Destination Protocol Length Info
892 36.447966 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2348; Unit: 1, Func: 2: Read Discrete Inputs
893 36.477244 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2349; Unit: 1, Func: 1: Read Coils
894 36.477422 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2349; Unit: 1, Func: 1: Read Coils
895 36.677083 192.168.204.133 192.168.204.136 TCP 60 50154 → 502 [ACK] Seq=1753 Ack=1461 Win=16392 Len=0
904 36.945774 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2350; Unit: 1, Func: 2: Read Discrete Inputs
905 36.945875 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2350; Unit: 1, Func: 2: Read Discrete Inputs
906 36.977132 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2351; Unit: 1, Func: 1: Read Coils
907 36.977211 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2351; Unit: 1, Func: 1: Read Coils
908 37.177086 192.168.204.133 192.168.204.136 TCP 60 50154 → 502 [ACK] Seq=1777 Ack=1481 Win=16387 Len=0
917 42.465935 192.168.204.133 192.168.204.136 TCP 66 50189 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
979 43.510233 192.168.204.133 192.168.204.136 TCP 66 50190 → 502 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK_PERM=1
980 43.510394 192.168.204.136 192.168.204.133 TCP 66 502 → 50190 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
982 43.521731 192.168.204.133 192.168.204.136 TCP 60 50190 → 502 [ACK] Seq=1 Ack=1 Win=65700 Len=0
984 43.521732 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2360; Unit: 1, Func: 2: Read Discrete Inputs
997 43.609758 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2360; Unit: 1, Func: 2: Read Discrete Inputs
1000 43.663419 192.168.204.133 192.168.204.136 Modbus… 66 Query: Trans: 2361; Unit: 1, Func: 1: Read Coils
1001 43.663496 192.168.204.136 192.168.204.133 Modbus… 64 Response: Trans: 2361; Unit: 1, Func: 1: Read Coils
1004 43.859293 192.168.204.133 192.168.204.136 TCP 60 50190 → 502 [ACK] Seq=25 Ack=21 Win=65680 Len=0
Frame 984: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{7D3BD728-DBEF-4AF5-BE04-85A03BE1A395}, id 0
Ethernet II, Src: Vmware_fd:e8:f0 (00:0c:29:fd:e8:f0), Dst: Vmware_03:d3:96 (00:0c:29:03:d3:96)
Internet Protocol Version 4, Src: 192.168.204.133, Dst: 192.168.204.136
Transmission Control Protocol, Src Port: 50190, Dst Port: 502, Seq: 1, Ack: 1, Len: 12
Modbus/TCP
Modbus
0000 00 0c 29 03 d3 96 00 0c 29 fd e8 f0 08 00 45 00
0010 00 34 4b 81 40 00 80 06 94 e3 c0 a8 cc 85 c0 a8
0020 cc 88 c4 0e 01 f6 5e 9d 9e cc c5 02 e8 98 50 18
0030 40 29 d6 cc 00 00 09 38 00 00 00 06 01 02 03 20
0040 00 01

It’s using Apache Derby Database. Data source and point information are store in here. Here’s is an example of accessing the database from “ij”, which is a tool provided by Derby.



It shows that we are sending TCP Modbus traffic to OpenPLC, may be related to modbus4j-3.0.2.jar executed earlier.



There are different classes running inside that jar file, for message reading and writing purposes.