our measurements. Additionally, the webpage explains the cause and scope of our measurements.

Lastly, scans must be no larger or more frequent than is necessary and must, if possible, be spread out, to not overwhelm remote networks or cause needless load [DWH13]. We scanned once a week, to allow observations over time, while spreading the load on remote hosts over time by using a random permutation of the list of IP addresses to scan and by spreading the scan to approximately 24 hours. Since we only found small numbers of hosts, scanning subsets of the public IPv4 space was not a valid option.

Conclusive, to the best of our belief we respected and obeyed the recommended practices. Further, additional to the recommended practices we never attempt to exploit security problems, guess passwords, or change device configuration. Since we are aware of the low computing power of ICS devices and the possibly serious implications of overloading or crashing real ICS devices we proceeded carefully. A special environment, utilizing honeypots, was setup and used for testing and development in all applicable scenarios.

### 3.3.2 Methodology and Workflow of Scanners

In this section, we introduce the methodology and workflow of our scanner setup used to accomplish the defined goals: A customizable, scanner capable of finding and retrieving banner data of the complete set of reachable ICS devices using S7, EtherNet/IP or SNMP protocol in the public IPv4 address space.

Due to prior experience and great performance, ZMap, in an extended version for capturing TCP options, was used as a TCP SYN and UDP application layer scanner (cf. Section 2.3). It scanned the SNMP hosts to retrieve banner data and searched for open ports of the S7 and EtherNet/IP protocols in the public IPv4 address space. Once open ports for S7 or EtherNet/IP are found ZGrab2 is used for application layer scanning to extract banner data.

In the following, our modified and extended ZGrab2 module for S7 and the completely newly developed EtherNet/IP module are discussed, and our workflow for retrieving SNMP device banner data is introduced.

#### 3.3.2.1 Siemens S7 Scanner Module

S7 communication is a proprietary protocol developed by Siemens for their S7 product family. Our application layer scanners goal is, to retrieve device metadata that allows for an identification of the device, which enables us to identify the devices, so vulnerabilities can be searched using the NVD. Therefore, multiple SZLs, which are datasets holding identifying device metadata, need to be retrieved from remote Siemens devices using the S7 protocol.

To allow for the retrieval of more and substantially important data, e.g., firmware or hardware version of devices, we rebuild the ZGrab2 S7 packet parser. Additionally, our rebuild parser fixed bugs in the default ZGrab2 which, in rare cases, could lead to a misidentification of data fields. Further, our extended ZGrab2 version also

saves communication metadata, i.e., all error code fields from the S7 parameters for all S7 responses and extends the scan procedure to allow for a better detection of honeypots.

Figure 3.1 shows a sequence diagram of the S7 scan procedure using ZGrab2 and ZMap to gather SZLs from a remote host. The different phases of the interaction are indicated by color. The scanning workflow, the data gathered, and further details are discussed phase by phase alongside the interactions in the figure.

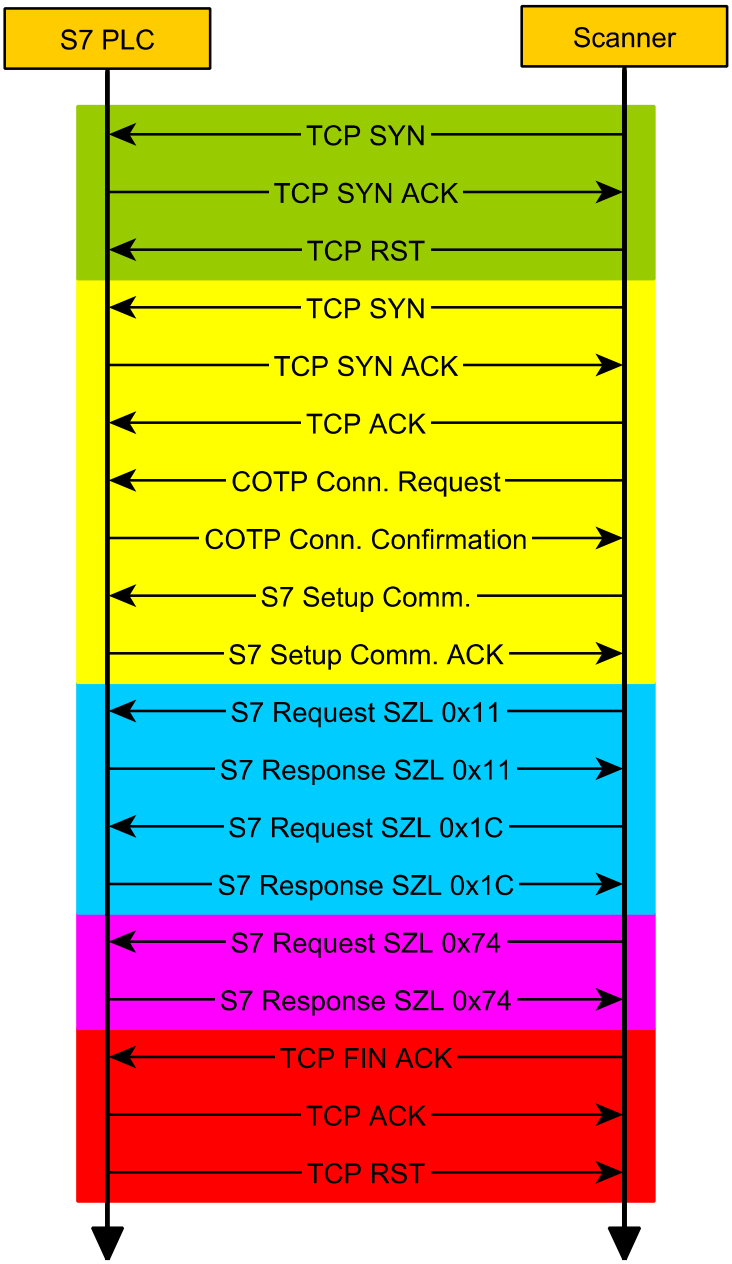


**Figure 3.1** A sequence diagram depicting the interactions between ZMap/ZGrab2, called scanner, and a remote S7 PLC, to retrieve device banner data. The five scan phases are indicated by color.

### Connection Establishment

In the green phase ZMap performs a TCP SYN scan, to find a host offering a service on port 102, the default ISO on TCP port [RC87], as described in Section 2.3. If ZMap finds a host, all the TCP options in the TCP SYN ACK, the window size and the IPID are captured and saved. The corresponding IP address is transmitted to ZGrab2.

The yellow phase begins, where ZGrab2 establishes an S7 connection. Therefore, ZGrab2 first conducts a regular TCP handshake, followed by a COTP connection request packet starting the COTP handshake, which is used to establish the ISO connection. As described in Section 2.1.1.1 we need to define the maximal TPDU size and the TSAPs. We chose protocol specific values that are successfully used in related work, i.e., 1024 bit maximum TPDU, source TSAP of `0x100` and two possible values for the destination TSAP: `0x102`, or if necessary, after the COTP connection request failed `0x200` in a retry [Tea, Lyo, MMA+16]. This approach was chosen due to considerations regarding load minimization on remote hosts, by only resending one COTP connection request. The COTP handshake ends successfully with step eight, the COTP connection confirmation packet.

After all transport layer connections are established ZGrab2 starts the S7 handshake, as described in Section 2.1.1.1, by sending the S7 Setup Communication request. In this request communication parameters are proposed. The maximal number of unacknowledged messages is set to one for the scanner and PLC and the PDU length is set to 480. The S7 device sends back an S7 ACK. If no error is present in the header, then the connection is fully set up using the negotiated communication parameters, which are present in the response and ZGrab2 starts querying data.

### Application Layer Interactions

Once all connections are setup the blue phase begins, in which all SZLs holding identifying device information are requested, using the default ZGrab2 approach. Therefore, the scanner sends an S7 request function for reading an SZL with ID `0x0011`, the Module Identification, as described in Paragraph 2.1.1.1.

| | | | |
|---|---|---|---|
| 0x0001 | 6ES7␣315-2EH14-0AB0␣ | 0x00c000040001 | } S7 Record |
| 0x0006 | 6ES7␣315-2EH14-0AB0␣ | 0x00c000040001 | } S7 Record |
| 0x0007 | ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣ | 0x00c056030206 | } S7 Record |
| 0x0081 | Boot␣Loader␣␣␣␣␣␣␣␣␣ | 0x000041200909 | } S7 Record |

**Figure 3.2** Excerpts from an S7 response packets data part for the Module Identification SZL (ID: `0x0011`) request as observed with Wireshark. The undocumented record `0x0081` is present. The middle data field is a decoded ASCII representation, all other fields are raw hex strings.

Figure 3.2 shows the actual Module Identification SZL holding metadata that is present in the response to the read Module Identification SZL `0x0011` request. The

SZL is made up of individual records which are identified by a two-byte hex-number, followed by a 20-byte data field usually holding Siemens order numbers and six bytes of record specific data, where version information can be present.

According to Siemens documentation for the S7-300/400 [Sie10], the Module Identification SZL (ID: `0x0011`) is made up of three data records, the identification of the module with index `0x0001`, the basic hardware using index `0x0006`, and the basic firmware identified by index `0x0007`.

The default ZGrab2 only extracts data, encoded as ASCII, from the middle 20-byte fields, where Siemens order numbers are. The order number from record `0x0001` is saved as "module_id", the one from record `0x0006` as "hardware" and the last order number from record `0x0007` is saved as "firmware". This information suffices to identify a device, but not to assess possible vulnerabilities, since no version information is captured, although present in the packet.

However, our extended ZGrab2 additionally saves all further potentially useful data present. Hardware and firmware version information could be extracted from the six-byte fields in the end of the records. Since the Siemens SZL documentation differs substantially from our measured results, we verify our results and interpretation of data in Section 3.4. Multiple undocumented records in the SZL, which are only returned by certain devices in certain versions were noticed and are also captured by our scanner, which can be seen in Figure 3.2 where the record with ID `0x0081` is shown.

The next interactions of the S7 scan process are a request for the Component Identification SZL with ID `0x001C` followed by a corresponding response. This response for the Component Identification request is made up of data records. Possible information present in this packet is: the (automation) system, plant, and module name, which can be set by the user, the serial number of the module, and the memory card, the modules location, the modules Original Equipment Manufacturer (OEM) ID, Copyright, and CPU profile information. No further undocumented information could be found in any Component Identification responses and all data was recorded by the default ZGrab2. The information provided by the Component Identification response are mainly useful for the detection of honeypots, discussed in detail in Chapter 4 and can be helpful in identifying device owners.

We not only rebuild the parsing module, to save more device data, but also extended the scan procedure, which is shown in the pink phase, only conducted by the extended ZGrab2. First the Status of the Module LEDs SZL with ID `0x0074`, is requested and then the corresponding response is saved in its entirety. The use for this data is presented in Section 4.3.2. This interaction marks the end of the application layer scan.

The transport layer connections get terminated in the orange phase and the scan ends.

### Output

All data and metadata recorded by ZGrab2 is outputted in JSON format and appended to a `.json` file. Figure 3.3 presents an example output. In total our S7 scanner can capture up to 27 distinct data fields, but in the outputs all empty ones are

```
{"ip": "X.X.X.X", "data": {"siemens": {"status": "suc-
cess", "protocol": "siemens", "result": {"is_s7":
true, "system": "S7-300", "module": "CPU 315-2 PN/DP",
"plant_id": "XXXXXXXXXXX", "copyright": "Original
Siemens Equipment", "serial_number": "S C-XXXXXXXXXXX",
"module_type": "CPU 315-2 PN/DP", "module_id": "6ES7
315-2EH14-0AB0 ", "hardware": "6ES7 315-2EH14-0AB0 ",
"firmware": " ", "version": "3.2.6", "versionbyte":
"V", "versionhardware": "4.0.1", "versionhardwarebyte":
"\u0000", "versionmodule": "4.0.1", "versionmodulebyte":
"\u0000", "unknown11": "Boot Loader \u0000\u0000A \t\t",
"unknown11num": "0129", "conpoterror": "d4.6", "elfer-
ror": "0.0", "einscerror": "0.0"}, "timestamp": "2020-
03-16T00:00:00Z"}}}
```


**Figure 3.3** Data our extended ZGrab2 grabbed from a remote host in JSON format. `conpoterror`, `elferror`, and `eincerror` represent the "Error code" for the different SZL requests. A value other than `0.0` implies a failed request. The IP address, serial number, plant name, and timestamp are altered.

omitted. The data fields `system`, `module`, `plant_id`, `copyright`, `serial number`, and `module_type` are obtained by parsing the data from the Component Identification SZL request. The next three entries: `module_id`, `hardware`, and `firmware` are the Siemens order numbers from the Module Identification SZL data, which is also extracted by the default ZGrab2 parser. All remaining data fields are added additionally by our extensions. The firmware version is encoded as `version`, the hardware version as `versionhardware`. However, only the first digit in the field `versionhardware` encodes the actual hardware version. The latter two digits can be omitted, since they do not hold information which could be identified and only remain for compatibility with other modules. The field `unknown11num` gives the index of an undocumented S7 record in the Module Identification SZL response, the corresponding data returned is saved in the `unknown11` field accordingly. Error fields, i.e., `elferror`, `eincerror`, and `conpoterror` hold the values of the Error field from the S7 responses parameter part. Since data from the Component Identification SZL request is present, there is no error in the `eincerror` field, the same applies for the Module Identification SZL request and the field `elferror`. However, no output of the Led Status SZL request is present. This can be explained by the error `0xd406` ("Required information currently unavailable") occurring, which can be seen in the corresponding field `conpoterror`, where a representation of the error is shown. Finally, all remaining data fields not introduced hold information which, after thorough evaluation, proved not usable for the research conducted in this work.

#### 3.3.2.2 Ethernet/IP

EtherNet/IP is an industrial protocol utilized by close to 600 different vendors of industrial devices. Our ZGrab2 EtherNet/IP scanning module generally has the same goal as the S7 scanner module: to retrieve identifying device metadata that enables