

## Forum:[Hack Forums](#)

Thread: [TelcoXploiting Series EP1: Getting SS7 access free \[GUIDE\] - being updated](#)

**Interconnector** - August 20, 2025, 6:16 pm

A long, long waited guide has arrived for you guys. Hope you really appreciate it, took me 10 years.

Brief description of what this includes:

- Scanning of SCTP servers with an SS7 stack behind it.
- Setting up a local SS7 stack for comms.
- Understanding intercepting, locating, spoofing.

If you need a simulation server, let me know.

RequirementsUbuntu 22 (or similar Linux)

1 VPS ^

Java 17+ (LTS works great), Maven, VS Code (optional)

Wireshark

-----  
In order to find an endpoint, you can simply use the -sY parameter in nmap, or preferably make your own. Script is available if needed, but nmap can do a fine job.

Ports CANNOT be relied on, so 2905 for M3UA, the ones for SUA and such, forget them. You'd be wasting your time digging further than you have to.

To avoid it being shut down instantly, PM me for the IP subnet to scan for if you want to get a hands-on experience of a positive.

Example nmap line: nmap -v --open -sY 0.0.0.0/24 -p0-65535

Any init\_acks will be listed as successful hits, there's other parameters for cookie acks too. "nmap -h" for help.

Remember, an SCTP server is not always an endpoint with a stack, so attempt to send an ASUP or ASPUP to see if it responds with an abort, a data chunk of nothing, when it's supposed to show MTP3 and M3UA information in Wireshark.

-----  
Worst part.

Now you need to get yourself the proper version of jSS7(and/or jDiameter if you have experience), Java, and SCTP packages required for the machine.

Maven Pom example:

(too long thread post so PM)

These are required for the Java project we are going to make.

I'd recommend downloading VS Code, installing maven, Java SDK 23? I think so. The whole Java and Maven extension package.

Let's get the basics down in Java, if you have installed the above correctly and linked the runtime Java exec in VS code, it will automatically pick the right imports required. Typically, do prefer Mobicents only, use Restcomm for SCTP mainly, here's a messy import section: (too long thread post so PM)

This is enough for the rest of this guide's code to work without import issues, if installed correctly. (The full Java file can be sent in PM to avoid indexing conflicts)

In our main class, let's set the required variables we need for connectivity to SCTP and some M3UA&SCCP information.

The list of signalling point codes to use are here: <https://www.itu.int/pub/T-SP-Q.708B> - always look for the newest version of the current year. Your local SPC is found in the configuration of the stack on the server that you got handed by God. Routing lists and configuration files are available in PM also, to avoid indexing.

```
Code    protected final static int CLIENT_SPC = TO_BE_FILLED;
        protected final static int SERVER_SPC = TO_BE_FILLED;
        protected final int NETWORK_INDICATOR = 0;
        protected final int SERVICE_INIDCATOR = -1; // SCCP1
        protected final int CLIENT_SSN = 8;
        protected final int SERVER_SSN = 6; // HLR
        protected final static String LocalGT = "TO_BE_FILLED";
        protected final static String RemoteGT = "TO_BE_FILLED";

        protected final String RemoteMSISDN = "TO_BE_FILLED";
        protected final String RemoteSMSC = "TO_BE_FILLED";

        protected final String CLIENT_IP = "YOUR IP";
        protected int CLIENT_PORT = 8012;
        protected final String SERVER_IP = "TARGET ENDPOINT IP";
        protected int SERVER_PORT = TO_BE_FILLED;
```

I won't post the entire code in the thread itself, this is to explain these important variables.

CLIENT\_SPC is the point code you've been assigned, or you've taken.

SERVER\_SPC is the point code of the remote SCCP you're reaching out to.

Now, depending on how you choose to route, there is different ways to do this over IP(SIGTRAN) - In this case we'll be routing via GT.

A GT is a global title, and that is basically a phone number looking number. It's in reality what could

be an HLR, VLR, MSC, etc.

The opposite part(your target's carriers STP) will(if it can and has a contract) be able to route with something alike regex expressions. Everything is configured from top to bottom, we'll get to it.

This was a lot of technical stuff, yes, sounds clever, but what now? I want to clear some myths, and dead legends.

[Anytimeinterrogation](#) is a request that is very "famous" for tracking. It is dead now, no carrier will accept it anymore.

## Tracking

You fetch the IMSI of the MSISDN, the subscriber. The IMSI is basically your international internal number in the system, it declares what your carrier is by MCC and MNC numbers, which indicates what carrier it specifically is, the rest is randomly generated and assigned I assume.

Once you've gotten the IMSI, by either `SendRoutingInfoForSM`, `SendIMSI`, `SendParameters`, perhaps others too. `dtest.info` is a fantastic place for documentation, the best in my opinion.

You go make a [ProvideSubscriberLocation](#) OR a [ProvideSubscriberInfo](#) - Only 1 needs IMSI, but typically networks will not respond to purely just an MSISDN provided. PSL takes MSISDN, but doesn't always reply with any information but an error, I haven't had success on any network, so it's most likely a standard.

PSL works on most networks that has been tested, it provides a location accordingly to what type of location request you're asking for:

Location estimate: Uncertainty in meters, last request was approx. 200m.

Coordinates encoded, but easily decoded with a GSM scheme, also provided on `dtest`.

The best location you will get is an `emergencyServices` request as LCS-client. You can spoof the MLC by simply using a national number, it usually takes it in.

Code `lcsClientType IE`

unrecognized values may be ignored if the LCS client uses the privacy override. Contents include: `emergencyServices` (0), `valueAddedServices` (1), `plmnOperatorServices` (2), and `lawfulInterceptServices` (3). otherwise, an unrecognized value shall be treated as unexpected data by a receiver a return error shall then be returned if received in a MAP invoke.

`Lawfulintercept...` provides most accurate cell ID information on the networks tested, but not location in coordinates.

The rest is not worth it.

-----

## Interception

For SMS - requires access to MSC and SMSC via `UpdateLocation` - USSD does not if implemented in remote network. `MT-ForwardSM` COULD be exploited, but in order to do so you need to know the sender.

A simple way to do this would be by USSD requests, but only a few carriers actually has implemented rerouting for SMS via USSD. Call barring and such is possible via [registerSS](#) - the arguments are self-explanatory if you click on the required fields:

`ss-Code`

basicService  
forwardedToNumber  
noReplyConditionTime  
defaultPriority

This can forward calls, and you could route the call by a PBX to act as a middleman.

The most used way is [UpdateLocation](#) - If it doesn't work due to firewall issues, due to missing contracting between the two carriers, simply spoof the sccp-calling GT to the target carrier's GT too. You do not need a response for this as long as it goes through on their end validly. You will get what I mean later.

By UpdateLocation you simply register the IMSI/subscriber on the hacked... I mean the carrier network you own, and then you have the control.

Any SMS, you route it, and call, you route it, data, you route it. If you do not route it, it will simply shut their service down.

You may know this request by the SMS message when crossing borders with updated prices on data and so forth, a registration on a new carrier network's MSC/VLR.

#### Blocking of incoming SMS

This is very very simply. ReportSM - simply request to the remote SCCP GT of an SMSC number belonging to the target carrier, and insert absent-subscriber into the delivery outcome.

#### sm-DeliveryOutcome

This parameter indicates the cause for setting the message waiting data. Contents include: memoryCapacityExceeded (0), absentSubscriber (1), and successfulTransfer (2).

Use number 2 if you want to reverse this.

I struck a firewall, and I need a response?

If your response is a timeout, there are multiple ways to find the STP most likely. Oftentimes the SMSC can be used as a "router" for guiding you to a subsystem like an HLR. Some even reach internationally if sending an SMS with mo-forwardSM, which you also can use to spoof an SMS with. If you get a masked IMSI or so, try a few digits up or down, the subsystem typically lies close to each other, and try act as a VLR instead of MSC as example.

What did I miss? Feel free to ask and I will add it to this thread.

I want my phreak status back:( :P

#### FAQ

Does it work everywhere? - Nobody in the world knows 100% - gsmmap.org is a horrible example besides Germany and Netherlands. Do not use it, it's outdated or incorrect. If you can send an SMS from a 2G network, and manage to interconnect to a Diameter network as example, then there's obviously interconnectivity and a downgrade attack is possible. Spoofing GTs are a big part of why it works most places, the only issue is getting the real IMSI sometimes.

How fucked am I if I mess up? Don't. I recommend you set up a jSS7 local stack to test it first, before you enter any real environment. The SCCP routing system can be quite difficult to set up.

Will I find an SCTP right away? What is SCTP? It's a protocol just like TCP and UDP is. SCTP is today oftentimes bridged over TCP, so pay attention to this too when scanning and requesting. A successful association looks like this if you use the lksctp tools:

### 1) SCTP basics (local association)

SCTP is a transport protocol like TCP/UDP, commonly used for SIGTRAN (e.g., M3UA over SCTP).

Install tools and load the module

```
Code sudo bash -lc 'apt-get update && apt-get install -y lksctp-tools wireshark && modprobe sctp'
```

```
To my target: sctp_test -H CLASSIFIED -P CLASSIFIED -h CLASSIFIED -p CLASSIFIED
```

Note the info when launching sctp\_test without args. -T use SOCK\_STREAM tcp-style sockets.

(allow the port if not already, if you use ufw: ufw allow 2909

```
Code root@ded:~$ sctp_darn -H X -P 2909 -h X -p 8014(:)) -s
```

```
sctp_darn ready to send...
```

```
X:2909-X:8014> test
```

```
Received SCTP_COMM_UP
```

```
New connection, peer addresses
```

```
X:8014
```

```
X:2909-X:8014>
```

Verify in Wireshark or tshark (simple view in a terminal - tshark -Y "sctp or m3ua" -V) Capture filter:

```
Code sctp or m3ua
```

You should see INIT -> INIT\_ACK -> COOKIE\_ECHO -> COOKIE\_ACK (association up)

Like this:

The clever folks will find SCTP servers if they paid attention to that.

Ask me anything really. I'm done with selling this shit, time to educate, since it's obviously a really damn lacking topic.

Episode 1.1(here): How to use a local stack to peer with an SCTP, and send an actual SS7 request out to the real world.

Episode 2: Unlimited individual phone number out&inboxes.

Episode 3: Spoof calls and SMS.

**descodificador** - August 30, 2025, 7:51 pm

It's incredible how that shit continues to be so unstable, haha, when will the other parts of this beautiful series be published?

**jwncho** - September 3, 2025, 2:23 am

Episode 3: I'm curious about intercepting a target Incoming sms, I have little or no coding ability. Is there anyway you can help with this service.

**chk888** - September 15, 2025, 10:04 am

"It's really been a terrible day. I paid him, waited for two days, and he hasn't provided me with anything. Did I run into a scammer?"

**dashan22wow** - September 15, 2025, 10:36 am

(Sep 15, 2025 10:04 AM)chk888 Wrote: [\[CITE\]](#) "It's really been a terrible day. I paid him, waited for two days, and he hasn't provided me with anything. Did I run into a scammer?"

Yes, it's absolutely a scam. The OP is nothing but a fraudster, preying on people's trust. Don't expect anything from him.