Bachelor's Thesis Computing Science

Radboud University Nijmegen

Attacking the power grid by minimizing the IPv6 search space

Author:
Pepijn van Grunsven
s1020173

First supervisor/assessor:
drs. ing. Pol Van Aubel

Second assessor:
dr. Bram Westerbaan

June 29, 2023

Abstract

The power grid is an integral part of modern society. Any disruption in the accessibility of electricity could have far-reaching consequences. This thesis examines the potential risks posed by solar power inverters which control how the generated electricity from solar panels transfers to the power grid. Since solar power inverters are increasingly often connected to a network their security implementations are essential to prevent attacks on the power grid. Our research focuses on the feasibility of exploiting vulnerabilities in the implementation of IPv6 in these inverters. Our findings show that these devices do not always follow best practices and do not implement known solutions for vulnerabilities.

Contents

Chapter 1

Introduction

Electricity is an essential part of the functioning of society. Anything that consumes electricity (not powered by mobile power sources) is connected to the power grid. If a larger outage was to occur in the power grid it would affect large parts of a country, state, or even continent. An occasional outage is to be expected, no system can be 100% reliable.

1.1 Problem Description

Generally speaking, power grids need to be balanced [18]. This means that at all times there must not be a significant difference in power generation and usage [18]. Whenever such significant differences do occur, they lead to spikes in voltage which could lead to the breaking of equipment or consumer electronics. This difference in power generation and usage also trips safeguards of the power grid, to prevent damage to it, which also leads to blackouts. A German report by the Office of Technology Assessment concludes that the consequences of a prolonged power outage could lead to the collapse of society [2]. From this, we can safely say that balancing the power grid is very important.

Ideally there is no difference in power consumption and generation. If any imbalance occurs it will not immediately cause problems. If the difference becomes larger than 3-5GW [20] the network operator cannot manage this imbalance anymore. If anybody is able to cause the appearance or disappearance of 3GW on the power grid they can effectively trigger a blackout.

This number of 3-5 GW is well known and studied for conventional methods of generating power. With the large subsidies governments are offering to anyone willing to install solar panels on their roof there is a large increase in the amount of these systems. The power the solar panels provide is first consumed locally. If the local power requirement is lower than the solar panels generate the surplus power gets transferred to the power grid. The solar panel installation works by measuring if it should be turned on or off. There are 2 scenarios when such a system will shut off:

- The system will shut off when it no longer detects a 50Hz pulse on the power network.
- They shut off when the voltage of the power network exceeds 252 volts.

The first scenario is to prevent danger to maintenance workers who are working on the network. The latter is to prevent damage to electronics. This is all handled by solar power inverters installed at the same location as the solar panels.

The companies behind these solar power inverters often make the decision to develop an app or web interface for their systems. A nice touch for the end consumer, after all now they can see how much they are saving on their electricity bill. This usually means however that these solar power inverters are now connected to an IP network. In some, but not all, cases this network might be accessible via the internet. It might be enticing for an attacker to target these systems, even if these single-home systems may not generate that much electricity when compared to 3 GW. If the attacker somehow gets enough of these systems in their control, it might lead to catastrophic consequences [1] [12]. For example, the 2015 power outage in Ukraine was caused by attackers who gained access to the systems of three energy companies, resulting in approximately 225.000 people losing power [13].

A question arises: How many solar power inverters does one need to control to attack a power grid successfully? As said before, these single-home installations will not generate huge amounts of electricity, they may only be in the order of a few thousand watts. If we conservatively assume a generating power of 1000 watts, we "only" need 3 million solar panel installations to control a power grid. This might seem like much, but what if an attacker with a large budget and lots of time wants to do this? There are certainly countries that have the capability to perform large-scale attacks. In this thesis, we will find out and discuss if this assumption of 1000 watts is big enough for comfort.

In 2017 Willem Westerhof found many devices by the manufacturer SMA connected to the internet via IPv4 [4]. SMA commented on all CVEs outlined in Westerhof's research and claimed that none were an active threat to power grids [19]. A recent report by the Dutch government contradicts SMA's analysis. The report states that if certain solar power inverters are directly reachable through the internet, they form a security risk [14].

This gives rise to the question: are these systems directly reachable through the Internet? As will be explained in section 2.1 IPv4 usually uses NAT. Therefore IPv4 is often seen as secure [17]. This conclusion comes from the way NAT makes it tougher to directly reach systems via IPv4. This could be interpreted as IPv6 being less secure since it does not use

NAT. This is the case unless a properly configured firewall is in place. It is not reasonable to just assume a firewall is in place. Given the depletion of the IPv4 address space which increases the likelihood of new devices using IPv6. The question arises, if there is a vulnerability in a solar power inverter and these are connected through IPv6 is it feasible to attack them en masse to take down a power grid?

1.2 Scanning for Vulnerable Devices

If an attacker knows a certain device has a vulnerability, they want to find these devices to exploit. The common way to find these devices is IP scanning, which we will focus on in this thesis. The basis of IP scanning lies in performing some probe to a specific IP address and waiting for the device listening on this IP to reply. If a reply is formatted like expected, it means that on that specific IP, such a device is actively listening.

Standardized tools exist to perform these scans. One such tool is Nmap. Nmap can be used for many applications when it comes to scanning, the Host Discovery feature might be the most interesting for us. This feature performs a very basic reconnaissance scan on networks to see if there are any interesting hosts [9]. It took a long time to singlehandedly scan the entire IPv4 address space. Therefore Zmap was developed. Zmap is specifically made to perform comprehensive Internet-wide research scans, a scan using Zmap can be completed 1300 times faster compared to the same scan with Nmap [6]. Nowadays it is not even necessary to perform these scans manually. Shodan, a search engine for the Internet of Things, maintains a database of all hosts on IPv4. Shodan probes IPv4 addresses every few hours, publishing its results online, meaning anyone can access this information at will.

Now what happens when we want to find devices that we know are connected to the internet through IPv6? Scanning all IPv4 addresses with Zmap can be done in 45 minutes [6]. Since IPv6 has 128-bit addresses compared to the 32-bit addresses of IPv4, IPv6 has 2^{96} times more theoretical addresses than IPv4. This would seem to make scanning the entire IPv6 address space infeasible [3]. This will be further explored in chapter 2.

1.3 Approach

If we want to make sure devices controlling large amounts of power are not susceptible to attacks, we must know how these devices behave. We therefore wish to cover the following question: Do solar power inverters utilize IPv6 in such a way that in the absence of correctly configured firewalls they could feasibly be attacked?

We divided this question into subquestions:

1. Are there techniques that make scanning the IPv6 address space feasible?
2. Do solar power inverters utilize IPv6?
3. If they do, are they susceptible to the aforementioned techniques?

Combining the answers to these subquestions gives us enough information to answer the main research question.

1.4 Attacker Model

In this thesis, we want to give a hypothetical attacker the most chance of succeeding in their attack. The attack we are exploring concerns direct reachability via IPv6, the attacker is therefore not allowed to for example attempt to control the solar power inverters by breaching the manufacturer's cloud environment. One advantage we do allow our attackers to have is that they know the type of solar power inverters for which known vulnerabilities exist.

We were unable to find numbers to reliably give an average output on home installations of solar panels. We based our average output of solar power inverters on the ones we tested, the one with the lowest peak wattage output of 2000 watts and the one with the highest peak wattage output was 8000 watts. In this thesis, we used 1000 watts as a conservative lower bound because no solar panel will give the maximum output all the time.

Chapter 2

Scanning the IPv6 Address Space

In this chapter we will answer subquestion 1: Are there techniques that make scanning the IPv6 address space feasible?

2.1 IP Addresses

The internet works by transferring packets of data between hosts. For these packets to “know” where to go addresses are needed. Historically, on the internet, IPv4 was used to accomplish this. IPv4 uses 32-bit addresses. This gives IPv4 the theoretical limit of only 2^{32} or $4,294,967,296$ addresses, to be used by machines worldwide. The exhaustion of IPv4 addresses has been upon us for quite some time now. To combat this network address translation (NAT) was developed. NAT maps multiple private addresses inside a local network to one public address. Devices behind NAT are therefore not (reliably) directly reachable from the broader internet. To put it simply, an outsider cannot start a connection to a machine behind NAT, while the device itself can make connections to the outside which can then communicate back to the device. Most people use NAT daily for their home network as their ISP usually only provides their home with one public IPv4 address. In 1991 the IETF decided to design a successor to IPv4, which would become IPv6. Fundamentally, IPv4 and IPv6 perform the same task: a way for hosts connected to the internet to transfer packets to each other through unique addresses. IPv4 had some issues which IPv6 solved. We will not review every difference since most of them are irrelevant to this thesis. The relevant differences for us are:

- The length of the addresses in IPv6 compared to IPv4. IPv4 uses 32-bit addresses while IPv6 uses 128-bit addresses, which means IPv6 has a theoretical limit of 2^{128} addresses. When written out this is incomprehensible.

- Because IPv6 has such a vast address space the use of NAT is not necessary. This means that devices using IPv6 can be directly connected to, unless such connections are explicitly blocked by a firewall. For end users of ISPs these firewalls need to be installed by the ISP, from personal experience we can say this does not always go well on the first try [15]. Our worry is an attacker from the outside, which would be stopped by NAT installed in IPv4 networks, would not be stopped here.

As mentioned in section 1.2, attackers can scan for vulnerable devices. To determine how long it would take to scan the IPv6 address space in the best-case scenario we assume we use a tool as fast as Zmap (which scans the IPv4 address space in 45 minutes [6]). We need to calculate how much bigger the IPv6 address space is compared to the IPv4 address space. This can be done by calculating $\frac{2^{128}}{2^{32}} = 2^{96}$. So just scanning the IPv6 address space would take about $\frac{2^{96} \cdot 45}{60 \cdot 24 \cdot 365} \approx 6.783 \cdot 10^{24}$ years. For context, it will only take about $5 \cdot 10^9$ more years for our sun to explode [10]. At that point, it will not make sense to scan for vulnerable solar power inverters to attack a power grid since, among other things, solar panels will not generate any more electricity.

However, by making some reasonable assumptions about the IPv6 search space we may be able to bring this number down.

2.2 How Many IPv6 Addresses Are in Use?

To start reducing the number of IPv6 addresses we will need to scan, we will look at the IPv6 addresses the Internet Assigned Numbers Authority (IANA) allocated to be used. If IANA has not yet allocated a range of internet addresses, they simply cannot be assigned to end users by ISPs. So this is a very simple way to significantly reduce the number of addresses we need to scan.

IANA has published a list with the ranges they have allocated [11]. Before we start doing any calculations we can already dismiss most of these records. First of all, the ranges that have the status “RESERVED” do not need to be looked at, as the name implies, these ranges are reserved but not actively used. Because we are conducting this research in The Netherlands, and an attack on the power grid needs to be geographically localized, we simply only have to take the RIPE NCC ranges into account. RIPE NCC is the regional Internet registry (RIR) for Europe, the Middle East, and parts of Central Asia. Listing all the ranges stated under RIPE NCC we end up with the following list:

1. 2001:0600::/23
2. 2001:0800::/22
3. 2001:1400::/22
4. 2001:1a00::/23
5. 2001:1c00::/22
6. 2001:2000::/19
7. 2001:4000::/23
8. 2001:4600::/23
9. 2001:4a00::/23
10. 2001:4c00::/23
11. 2001:5000::/20
12. 2003:0000::/18
13. 2a00:0000::/12
14. 2a10:0000::/12

Given these ranges we can calculate how many IPv6 addresses are allocated in the RIPE NCC area: $2^{(128-23)}+2^{(128-22)}+2^{(128-22)}+2^{(128-23)}+2^{(128-22)}+2^{(128-19)}+2^{(128-23)}+2^{(128-23)}+2^{(128-23)}+2^{(128-23)}+2^{(128-20)}+2^{(128-18)}+2^{(128-12)}+2^{(128-12)} \approx 2^{117}$ IPv6 addresses.

2.3 Further IPv6 Search Space Reduction

As found in section 2.2 there are only 2^{117} IPv6 addresses our attacker needs to scan. Now we have to see how much bigger 2^{117} is compared to the IPv4 address space: $\frac{2^{117}}{2^{32}} = 2^{85}$. Assuming this is all the attacker knows, this would now "only" take $\frac{2^{85} \cdot 45}{60 \cdot 24 \cdot 365} \approx 3.312 \cdot 10^{21}$ years. The attacker must decrease this even further to even remotely be able to scan for the device they want to attack. This can be done by taking advantage of the way the IPv6 address is selected by endpoints.

2.3.1 EUI-64 Algorithm

A machine that connects to an IPv6-enabled network usually does not just have a static IP address. The network assigns an address dynamically. IPv6 uses two automatic ways for machines to obtain an address configuration: Stateless Address Autoconfiguration (SLAAC) and Dynamic Host Configuration Protocol for IPv6. Implementations must support SLAAC while DHCPv6 is optional [8]. We will focus on SLAAC in this thesis as this is likely to be the most common home setup.

The "prefix", i.e. the firts 64 bits of the address, of the network the device wants to join is the same for all devices in that subnet. This is simply sent to the device by the default gateway of the network. The device then calculates its own EUI-64 address to append to the prefix to get its final IPv6 address. The calculation of the EUI-64 address is based on the MAC address of the network interface of the machine, which is a unique 48-bit number assigned to it by the manufacturer (although it can be changed by the device). The EUI-64 address is calculated as follows:

1. Take the first 24 bits of the MAC address.

2. Append it with 'FFFE'.

3. Append the last 24 bits of the MAC address [20].

4. Flip the 7^{th} bit, to classify the address as globally unique [5].

MAC addresses are made up of two parts:

1. The Organizationally Unique Identifier (OUI).
2. A device identifier, effectively random/unpredictable.

The OUI is a 3-byte sequence that specifies the manufacturer of that specific network card.

Risks of EUI-64

Risks of calculating the final IPv6 address this way have been known for some time [7]. Some of them are:

- It makes tracking users across networks easier because the last 64 bits are always the same. Furthermore, the MAC address can be determined from an IPv6 address by reversing the calculation of the EUI-64 address on the last 64 bits of the IPv6 address.
- The use of EUI-64 also makes scanning networks for specific devices from specific manufacturers easier because of the way MACs are constructed [21]. This is most relevant for our attacker and is further explained in section 2.3.2

Because of these risks a new method was proposed in RFC7217 that uses totally random bits instead of the MAC address [7]. In an ideal world, all our devices would use these techniques to provide their users with more security and privacy. So it is unfortunate that the adoption of RFC7217 has taken a long time. Linux has been supporting it for some time now, but Windows only implemented it recently [16]. No analysis on whether IoT devices use RFC7217 has been done. We can expect that if they run modern Linux they do support it, but it might not be enabled. If devices use real-time OS's the option may not even be available. For these devices it is likely to take some time to be implemented.

2.3.2 Using the OUI to Gain 40 Bits

Because we assumed our attacker knows which device they want to find on the internet, they can simply buy one and find which network card is used in this model. They can then find the OUI that corresponds to this manufacturer, in other words, just look at the first 3 bytes of the MAC address of the device they bought. Because we know the first half of the MAC address (24 bits) and also the static 'FFFE' part of the EUI-64 address (16 bits), the attacker still figured out 40 bits of the final IPv6 address. This reduces the search space to 2^{77}, because we can subtract 40 bits from the original 117. 2^{77} is $\frac{2^{77}}{2^{32}} = 2^{45}$ times bigger than the total amount of addresses in IPv4. So with this extra reduction it would take $\frac{2^{45} \cdot 45}{60 \cdot 24 \cdot 365} \approx 3.012 \cdot 10^{9}$ years.

With this reduction of the IPv6 search space, finding a specific model of device would "only" take a little longer than half the time from now up until the sun explodes. Clearly, this is still not a feasible attack.

2.3.3 Feasible Attack

The only way to make the attack feasible is if the attacker knows the prefix of a network with a device that has a flaw the attacker wants to exploit. This way the attacker only has to scan the local subnet. In concrete terms, this means the attacker only has to scan 2^{24} addresses since we already know the 64 bits of the prefix and 40 bits of the device-specific address. This could be done in about $\frac{2^{24}}{2^{32}} \cdot 45 \cdot 60 \approx 10.5$ seconds.

Even an attacker that does not know the OUI but does know the prefix would still only need to scan 2^{48} addresses, because we still assume the targeted devices use EUI-64 to calculate their IPv6 address. We know that 'FFFE' is part of the address and thus reduces 2^{64} addresses to 2^{48}. This would take this attacker $\frac{\frac{2^{48}}{2^{32}} \cdot 45}{60 \cdot 24 \cdot 365} \approx 5.6$ years, which might still be interesting to a state-level attacker. For example, a nation-state could have started this scan in 2015 and would have scanned enough addresses by now. But as explained in section 2.3.2 our attacker has an easy way to obtain the OUI and thus has a viable way to find devices much faster.

2.4 Attacker Model Revisited

This chapter explained the theoretical limits of scanning the IPv6 address space. Therefore we can adjust our attacker model introduced in section 1.4. As seen in section 2.3.3, the attacker must know the prefix of a network. This is not an unreasonable assumption. There are practical ways for an attacker to gather such data. For example by phishing: If a manufacturer's database of customers for solar power inverters were to leak, an attacker could simply send them all an email with a malicious link to their own website. The server this website is hosted on can then store all IPv6 prefixes that visit the website.

2.5 Attack

Assuming the attacker model of sections 1.4 and 2.4 combined, we can conclude that finding enough solar power inverters to generate 3GW takes at most 364 days. We determined this number by considering the following:

- The amount of solar power inverters needed to amass 3GW or 3,000,000,000 watts. As we assumed in section 1.4 each solar power inverter gives an average output of 1,000 watts, which means we need $\frac{3{,}000{,}000{,}000}{1{,}000} = 3,000,000$ solar power inverters.

- The search time for each prefix before our attacker finds the solar power inverter, which is at most 10.5 seconds as calculated in section 2.3.3. Which comes out to: $\frac{3{,}000{,}000 \cdot 10.5}{60 \cdot 60 \cdot 24} \approx 364$ days.

- On average our attacker finds a solar power inverter in a prefix each 5.25 seconds. Because our attacker sometimes finds the device immediately and sometimes not until the last address. Taking this into account we see it now takes $\frac{3{,}000{,}000 \cdot 5.25}{60 \cdot 60 \cdot 24} \approx 182$ days.

- And if we then consider an attacker is probably capable of performing the attack in a parallelized form it is almost certain they could perform the attack much faster. Assuming the attacker has enough bandwidth and multiple computers, the attack could possibly be done in less than one day.

If all our assumptions are correct it means that any attacker could cause a power outage in much less than half a year. As said in section 1.1 a total failure of the power grid can be catastrophic. The ability to black out a power grid is also a useful military capability.

2.6 Hypothesis

Given our revised attacker model, it is clear that attackers could find vulnerable devices if certain conditions are met. There are three distinct scenarios possible that give insight into the answer to the research question:

1. The solar power inverters do not utilize IPv6 yet.

2. The solar power inverters use IPv6 but do not use EUI-64 to calculate their IPv6 address.

3. The solar power inverters use EUI-64 to calculate their IPv6 address.

We believe that in scenario 3 the biggest threat lies, as the calculation of EUI-64 can be used to minimize the search space an attacker needs to scan to find the device.

Chapter 3

Determining IPv6 Use of Inverters

With the first subquestion answered we can now look at subquestions 2 and 3: “Do solar power inverters utilize IPv6?” and “If they do, are they susceptible to the aforementioned techniques?”. Our plan was to scan from within Radboud University. We would have asked people we know who had solar power inverters to turn off their firewall for our specific IP to be able to scan their network remotely. However, due to the university not having implemented IPv6 yet, this was not possible. So we had to physically go to these people’s homes, get on their network, and perform the scan locally. Luckily, since we already assumed the attacker knows the prefix of the solar power inverter this way of gathering data would have resulted in the same outcome as if the scan was performed remotely.

3.1 Experimental Setup

To test our theory we pinged known solar power inverters from within the network to see if they respond and are functioning as we think, then proceeded to Nmap on IPv4 and IPv6 to gain more insight.

To get their IPv6 address we wrote a script, included in appendix A.1, that asks the user for the known IPv4 address or the MAC of the device we want to test. If an IPv4 address is given the script determines the MAC address associated by running a directed Nmap scan to the IPv4 target. Once the MAC is obtained or input by the user, the EUI-64 address is determined via the algorithm from section 2.3.1. Then the network’s IPv6 prefix is obtained by checking the IP configuration of the machine running the script.[1] Finally, the prefix and EUI-64 address are combined to obtain the full IPv6 address. After getting the IPv6 address, this address is pinged.

[1] Current version does not support multiple different IPv6 prefixes on the scanning machine.

If the device responds we know for sure that the device uses IPv6 (and that it actively responds to pings) with EUI-64.

3.1.1 Testing the Script

To test our script we ran it against a Linux target device where we made sure it used EUI-64 to calculate its own IPv6 address. This was done by adding the lines:

```
slaac hwaddr
ipv6only=off
```

to the end of the file

```
/etc/dhcpcd.conf
```

on the target and restarting the networking service:

```
sudo service networking restart
```

When we ran the script and gave the IPv4 address of this host, it resulted in the same IPv6 address as the IPv6 address visible in the IP configuration of the host. Thus confirming that the script does give the desired outcome.

3.2 Results

We tested our target devices on multiple different aspects:

- **IPv6 utilization**: To know if the solar power inverter uses IPv6 we ran our script on a local computer with access to the network. The script then told us if the device uses IPv6 and EUI-64.
- **IPv4 web server**: We found the IPv4 address of the inverter by looking at the local router DHCP page and determining which it was by trial and error. We wanted to see if a web server is running on the device. To test this we simply went to the IPv4 address of the device in a web browser.
- **IPv6 web server**: If the device uses IPv6 we also wanted to check if a web server is running on this address. We checked this by going to the IPv6 address in a web browser.
- **IPv4 TCP port scan**: To see if any other ports, besides perhaps the web server, were listening we used Nmap on the IPv4 address of the device. The command looked like this:

```
sudo nmap -sV -p1-65535 -O XXX:XXX:XXX:XXX
```

- **IPv6 TCP port scan**: The same scan was performed on the IPv6 address of the device by running this command:

```
sudo nmap -sV -p1-65535 -O -6 XXXX::XXXX
```

We also tried running UDP port scans on some of the devices but due to the time these scans took and the results that were often not interesting (all ports are blocked), we omitted the outcome of these scans from our results.

In total, we had access to 5 different types of solar power inverters:

1. Enphase Envoy, installed in 2019
2. SolarEdge SE3000, installed in 2014
3. SolarEdge SE4000, installed in May 2023
4. Solar Frontier Turbo 1P, installed in 2015
5. Zeversolar Zeverlution 2000S, installed in 2016

For each of these, we had a computer with access to the local network to perform our tests. We will go over each inverter separately but we first summarize our results in table 3.1.

	Enphase Envoy	SolarEdge SE3000	SolarEdge SE4000	Solar Frontier Turbo 1P	Zeversolar Zeverlution 2000S
IPv6 utilization	Yes, with EUI-64	No	Yes, with EUI-64	No	No
IPv4 web server	Yes, with authentication on control functions	No	No	Yes, without authentication	Yes, without authentication
IPv6 web server	No	No	No	No	No
IPv4 TCP port scan	Only port 80, no SSH	No open ports	Old kernel version found	Only port 80	Only port 80
IPv6 TCP port scan	SSH service accessible	No open ports	No open ports	No open ports	No open ports

Table 3.1: Summary of Results.

3.2.1 Enphase Envoy

- **IPv6 utilization**: Running our script resulted in an IPv6 address which was subsequently pinged and the device replied to the ping. This means this device implements IPv6 with EUI-64.
- **IPv4 web server**: We tried going to a web page on the IPv4 address of the inverter. To our surprise, we immediately saw a status page without it asking us for a password. When we tried going further in this web interface it does prompt us for a password.
- **IPv6 web server**: When we tried to access the device via a web page with its IPv6 address nothing came up.
- **IPv4 TCP port scan**: When we do a port scan on the IPv4 address our script calculated, the only open port is port 80 which is used by the web server.
- **IPv6 TCP port scan**: One open port was found by Nmap, but it was port 22, not port 80. On this port an SSH service was listening. The version of OpenSSH was 6.6, a version released in March 2014, on a device with a software build date of June 2021. As remote access ability goes, we are happy it uses SSH but it is unclear to us why a manufacturer would ship a device with such old software.

Furthermore, this device might not have an IPv6 firewall since the SSH service is just open on IPv6, while on IPv4 the service is filtered, implying a firewall on IPv4. This might simply be an oversight from the manufacturer as it is weird that there is a mismatch between open ports on IPv4 and IPv6.

3.2.2 SolarEdge SE3000

- **IPv6 utilization**: This device did not reply to the EUI-64 address for this device. This might imply a correct implementation of RFC7217. What is more likely, considering RFC7217 was introduced in the same year this device was manufactured, is that this device just does not use IPv6.
- **IPv4 web server**: When we tried to access the device via a web page with its IPv4 address nothing came up.
- **IPv6 web server**: When we tried to access the device via a web page with its IPv6 address nothing came up.
- **IPv4 TCP port scan**: When we do a port scan on the IPv4 address of the device, there are no open ports to be found.

- **IPv6 TCP port scan**: When we do a port scan on the IPv6 address our script calculated, there are no open ports to be found.

This device does not seem to run any servers. All data this device collects must be pushed to the cloud environment of the manufacturer via an API as there is a working app for this device. The security of the API is beyond the scope of this thesis.

Under our attacker model this device is considered secure because it does not support IPv6. The security of the network stack and/or IPv4 implementation is not applicable to us.

3.2.3 SolarEdge SE4000

- **IPv6 utilization**: A newer successor of the SE3000, our script confirmed that the SolarEdge SE4000 responds to the ping on its IPv6 address using EUI-64.
- **IPv4 web server**: When we tried to access the device via a web page with its IPv4 address nothing came up.
- **IPv6 web server**: When we tried to access the device via a web page with its IPv6 address nothing came up.
- **IPv4 TCP port scan**: When we ran Nmap on this device, the OS detection reported the Linux kernel to be either version 2.6 or 3. Now we can not say for sure if this is correct. But if it is, and giving it the benefit of the doubt and assuming it is version 3, this device uses an OS kernel released in 2011. For a device being installed in 2023 this might be something for the manufacturer to look at.
- **IPv6 TCP port scan**: When we do a port scan on the IPv6 address our script calculated, there are no open ports to be found.

This device gives us valuable insight into the configuration of the software. Since there are no open ports on either IPv4 or IPv6, not even a simple web page, this device must either have no servers or a correctly configured firewall for IPv6. Considering this, the device must also use some sort of API to push data to a cloud, as all data is available in an app for consumers to use, just like the SE3000.

Under our attacker model this device is insecure as it uses IPv6 while also utilizing EUI-64.

3.2.4 Solar Frontier Turbo 1P

- **IPv6 utilization**: This device did not reply to the EUI-64 address for this device. It is highly likely that the device does not utilize IPv6.

- **IPv4 web server**: Trying to access a web page on the device's IPv4 address gives us a status page. All pages this web server provides do not prompt the user for a password. We did not have time to delve into all settings in the web interface, but it seems that the home page for the device gives the same functionality as the physical screen and buttons on the device. So if an attacker could get access to the network this device is on, they could theoretically mess with the settings of the device.

  However, there does not seem to be a simple on/off button in the web interface. This was confirmed by the owner, who once had to shut the device off because of an outage and could only do so by physically unplugging the device. So an attacker in the network would likely not be able to directly influence the power grid via the web interface.

- **IPv6 web server**: When we tried to access the device via a web page with its IPv6 address nothing came up.

- **IPv4 TCP port scan**: When we do a port scan on the IPv4 address our script calculated, the only open port is port 80 which is used by the web server.

- **IPv6 TCP port scan**: When we do a port scan on the IPv6 address our script calculated, there are no open ports to be found.

3.2.5 Zeversolar Zeverlution 2000S

- **IPv6 utilization**: This device did not reply to the EUI-64 address for this device. It is highly likely that the device does not utilize IPv6.

- **IPv4 web server**: The device hosts a web page on its IPv4 address. Anyone in the network can access this web interface without any form of authorization. Furthermore this device has an option to turn it off via the web interface. When we tested this, the 1.6kW generating solar panel installation simply turned off without asking for any type of authorization.

- **IPv6 web server**: When we tried to access the device via a web page with its IPv6 address nothing came up.

- **IPv4 TCP port scan**: When we do a port scan on the IPv4 address our script calculated, the only open port is port 80 which is used by the web server.

- **IPv6 TCP port scan**: When we do a port scan on the IPv6 address our script calculated, there are no open ports to be found.

Since the web server of this device has an option to completely turn off the device an attacker with access to this network, can influence the power grid.

3.3 Interpreting These Results

We now know that 2 out of 5 of our solar power inverters utilize IPv6 and use EUI-64 to calculate their IPv6 address. Even though most of the models we tested did not use IPv6 we still consider this as answering subquestions 2 and 3 in the affirmative.

Although we found the Zeversolar Zeverlution 2000S of section 3.2.5 had security issues. It is behind NAT so it is not as big a risk considering outside attackers. But it does demonstrate ill-considered security.

Chapter 4

Conclusions & Discussion

We will discuss our findings below.

4.1 Answers to Research Subquestions

We have answered all three research questions:

1. **Are there techniques that make scanning the IPv6 address space feasible?**

   As was determined in chapter 2, it is not feasible to brute force a scan on IPv6 even with a maximized reduction of the number of addresses we have to scan. But with realistic assumptions about EUI-64 and prefix, a feasible attack exists.

2. **Do solar power inverters utilize IPv6?**

   From our tests as described in chapter 3, there certainly are solar power inverters that utilize IPv6, even in our small sample size. The ones we saw that do, also utilize EUI-64 to compute their IPv6 address.

3. **If they do, are they susceptible to the aforementioned techniques?**

   Two of the tested solar power inverters utilize EUI-64. The devices also respond to the ping sent to them. So yes, if an attacker found a vulnerability in these specific types of solar power inverters the inverter is susceptible to the attack techniques discussed in chapter 2. So in general we can conclude this issue does exist

4.2 Conclusions

Now we have enough information to answer the main research question: Do solar power inverters utilize IPv6 in such a way that in the absence of correctly configured firewalls they could feasibly be attacked?

The answer is yes, solar power inverters that utilize IPv6 do exist and, in the absence of firewalls on the network edge, are feasible to attack by our hypothetical attacker. This also confirms our belief that scenario 3 of section 2.6 is indeed the biggest threat.

We must also not forget the security issue found in section 3.2.5 where an inverter could be turned off remotely without any authentication. Clearly the security of these devices needs more consideration.

We acknowledge that it is unlikely that all preconditions for our attack are met. Less likely is that we can find 3 million of these devices to attack the power grid. However this does not mean we can ignore the issues we found. Individual devices can still be attacked this way and the solutions are relatively easy.

The attack we used in this thesis could also be applicable to other IoT devices. Since there are often reports being published of new vulnerabilities being discovered in all kinds of IoT devices, they might not cause a blackout but it is something we may need to worry about.

4.3 Recommendations

We have seen that devices that utilize IPv6 must be protected by a well-configured firewall to prevent attacks. If not done so a device is susceptible to attacks. The same goes for the use of EUI-64: no modern operating system should use this method of obtaining an IPv6 address [7]. As explained in section 2.3.1 there are more modern and better methods available to implement that do not carry the same risks as the use of EUI-64.

Devices that do not (yet) utilize IPv6 must also implement firewalls, to make sure that when IPv6 is introduced in the device there is no oversight and the firewall is forgotten.

Furthermore, all web servers and/or SSH services running on a device should be protected by strong passwords. There should be no device that can be turned on and/or off remotely without any sort of authorization.

Lastly, all services and kernel packages on devices should be up-to-date and updateable especially when these services are exposed to the network. It is unacceptable that an old version of SSH service is accessible.

4.4 Future Work

During this research, several interesting questions remain unexplored:

- Our conclusions are based on 5 different types of solar power inverters, this is a small sample size. If this experiment is repeated it is valuable to test more individual solar power inverters with our methods to see if there are more vulnerabilities in their implementation. Furthermore it is also of interest to perform these scans remotely to see if firewalls are correctly configured on networks with solar power inverters.

- Given the fact that Radboud University currently has no IPv6 implementation, we were limited in what we could do in terms of experiments on IPv6. To test our hypothesis we now had to travel to people that had solar power inverters and a working IPv6 connection. If the university had an IPv6 network, we could have actually performed more scans remotely.

- In section 2.2 we concluded that we needed to scan about 2^{117} addresses. This might be further reduced by looking at the allocation of IPv6 addresses of large ISPs in a single country. We do not expect this to reduce the search time significantly if the attacker does not know the prefix, but it is worth checking to see if this is a viable way of reducing the search space.

- We conducted research to provide experimental data to see if our theory was at all applicable. During our research we learned that some inverters do not work at all unless they are connected to the manufacturer's cloud environment and may be turned on/off through the cloud environment. If this is a common practice implemented by manufacturers, hacking these cloud environments is interesting as one cloud network possibly controls thousands or more of these inverters. This would be an interesting starting point for future research.

- We also found websites that, like Shodan, publish port scan results for IPv6 networks. We suspect these sites simply traverse DNS records to find AAAA records and only scan the addresses found. While we did not further investigate these sites, it could be interesting to see how these actually work and if they are applicable to sites that do not specify AAAA records in DNS.

- In our testing of the solar power inverters we found some interesting opportunities to further attack these systems that were out of scope for this thesis. For example, some of the pages accessible via either their IPv4 or IPv6 address do not utilize HTTPS. Also, can the system somehow be tricked into changing the default password by e.g. sending a POST request without first authenticating or the SSH service user/password combo?

Bibliography

[1] Sridhar Adepu, Nandha Kumar Kandasamy, Jianying Zhou, and Aditya Mathur. "Attacks on smart grid: power supply interruption and malicious power generation". en. In: *International Journal of Information Security* 19.2 (2020-04), pp. 189–211. ISSN: 1615-5270. DOI: 10.1007/s10207-019-00452-z. (Visited on 2023-03-08).

[2] *Bericht des Ausschusses für Bildung, Forschung und Technikfolgenabschätzung (18. Ausschuss) gemäß § 56a der Geschäftsordnung Technikfolgenabschätzung (TA) TA-Projekt: Gefährdung und Verletzbarkeit moderner Gesellschaften - am Beispiel eines großräumigen und langandauernden Ausfalls der Stromversorgung.* German. Tech. rep. 17/5672. 2011-04, p. 136. URL: https://dip.bundestag.de/vorgang/bericht-des-ausschusses-f%C3%BCr-bildung-forschung-und-technikfolgenabsch%C3%A4tzung-18-ausschuss/35375 (visited on 2023-06-08).

[3] Kevin Borgolte, Shuang Hao, Tobias Fiebig, and Giovanni Vigna. "Enumerating Active IPv6 Hosts for Large-Scale Security Scans via DNSSEC-Signed Reverse Zones". In: *2018 IEEE Symposium on Security and Privacy (SP)*. ISSN: 2375-1207. 2018-05, pp. 770–784. DOI: 10.1109/SP.2018.00027.

[4] *CVE-Information – Horus Scenario.* URL: https://horusscenario.com/cve-information/ (visited on 2023-03-15).

[5] Steve E. Deering and Bob Hinden. *IP Version 6 Addressing Architecture.* Request for Comments RFC 4291. Num Pages: 25. Internet Engineering Task Force, 2006-02. DOI: 10.17487/RFC4291. (Visited on 2023-04-20).

[6] Zakir Durumeric, Eric Wustrow, and J. Alex Halderman. "{ZMap}: Fast Internet-wide Scanning and Its Security Applications". en. In: *22nd USENIX Security Symposium (USENIX Security 13)*. 2013-08, pp. 605–620. ISBN: 978-1-931971-03-4. URL: https://www.usenix.org/conference/usenixsecurity13/technical-sessions/paper/durumeric (visited on 2023-03-08).

[7] Fernando Gont. *A Method for Generating Semantically Opaque Interface Identifiers with IPv6 Stateless Address Autoconfiguration (SLAAC).* Request for Comments RFC 7217. Num Pages: 19. Internet Engineering Task Force, 2014-04. DOI: 10.17487/RFC7217. (Visited on 2023-04-13).

[8] Fernando Gont and Tim Chown. *Network Reconnaissance in IPv6 Networks.* Request for Comments RFC 7707. Num Pages: 38. Internet Engineering Task Force, 2016-03. DOI: 10.17487/RFC7707. (Visited on 2023-04-13).

[9] *Host Discovery | Nmap Network Scanning.* URL: https://nmap.org/book/man-host-discovery.html (visited on 2023-05-31).

[10] *In Depth | Sun.* URL: https://solarsystem.nasa.gov/solar-system/sun/in-depth (visited on 2023-05-31).

[11] *IPv6 Global Unicast Address Assignments.* URL: https://www.iana.org/assignments/ipv6-unicast-address-assignments/ipv6-unicast-address-assignments.xhtml (visited on 2023-05-31).

[12] Jay Johnson, Louis Jencka, Timothy Ortiz, Christian Birk Jones, Adrian Chavez, Brian Wright, and Adam Summers. *Design Considerations for Distributed Energy Resource Honeypots and Canaries.* English. Tech. rep. SAND2021-11609. Sandia National Lab. (SNL-NM), Albuquerque, NM (United States), 2021-09. DOI: 10.2172/1821540. (Visited on 2023-03-03).

[13] Robert M. Lee, Michael J. Assante, and Tim Conway. *Analysis of the Cyber Attack on the Ukrainian Power Grid.* Tech. rep. 2016-03. URL: https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2016/05/20081514/E-ISAC_SANS_Ukraine_DUC_5.pdf (visited on 2023-03-15).

[14] *Onderzoek storingsproblematiek en cyberveiligheid omvormers voor zonnepanelen.* nl. Tech. rep. Rijksinspectie Digitale Infrastructuur, 2023-05, p. 26. URL: https://www.rdi.nl/binaries/agentschap-telecom/documenten/rapporten/2023/05/30/onderzoek-storingsproblematiek-en-cyberveiligheid-omvormers-voor-zonnepanelen/Onderzoek+storingsproblematiek+en+cyberveiligheid+omvormers+voor+zonnepanelen.pdf (visited on 2023-05-30).

[15] Pol Van Aubel. *Missing firewall on IPv6 on consumer connection.* nl. Tweet. Archive: https://web.archive.org/web/20230621103014/https://twitter.com/polvanaubel/status/1248367604370194432. 2020-04. URL: https://twitter.com/polvanaubel/status/1248367604370194432 (visited on 2023-06-05).

[16] *RIPE Forum.* URL: https://www.ripe.net/participate/mail/ripe-forum (visited on 2023-04-13).

[17] M. Smith and R. Hunt. “Network security using NAT and NAPT”. In: *Proceedings 10th IEEE International Conference on Networks (ICON 2002). Towards Network Superiority (Cat. No.02EX588).* 2002-08, pp. 355–360. DOI: 10.1109/ICON.2002.1033337.

[18] Ingo Stadler. “Power grid balancing of energy systems with high renewable energy penetration by demand response”. en. In: *Utilities Policy.* Sustainable Energy and Transportation Systems 16.2 (2008-06), pp. 90–98. ISSN: 0957-1787. DOI: 10.1016/j.jup.2007.11.006. (Visited on 2023-03-08).

[19] *Statement on Cyber Security | SMA Solar.* URL: https://www.sma.de/en/statement-on-cyber-security (visited on 2023-03-15).

[20] *Use of EUI-64 for New Designs.* URL: https://www.ieee802.org/secmail/msg00396.html (visited on 2023-04-20).

[21] Ali Zohaib and Amir Houmansadr. “Automated Detection of IPv6 Privacy Leakage in Home Networks”. In: *Free and Open Communications on the Internet* (2023). URL: https://www.petsymposium.org/foci/2023/foci-2023-0005.php (visited on 2023-05-23).

Appendix A

MAC/IPv4 to EUI-64 Conversion Script

```python
import subprocess
import re
import socket
import netifaces
import ipaddress

# Prompt the user to enter a MAC or IPv4 address
address = input("Enter a MAC or IPv4 address: ")

# Check if the input is a valid IPv4 address
try:
    socket.inet_pton(socket.AF_INET, address)
    is_ipv4 = True
except socket.error:
    is_ipv4 = False

# If the input is an IPv4 address, get the MAC address associated with it
if is_ipv4:
    print("Finding MAC address associated with the given IPv4 address")
    nmap_output = subprocess.check_output(["sudo", "nmap", "-sP", address]).decode("utf-8")
    mac_address_search = re.search(r"(([0-9A-Fa-f]{2}:){5}[0-9A-Fa-f]{2})", nmap_output)
    if mac_address_search:
        mac_address = mac_address_search.group(0)
        print(f"MAC address: {mac_address} found")
    else:
        print("Could not find the MAC address associated with the given IPv4 address")
        exit()
else:
    mac_address = address
```

```python

# Calculate the EUI-64 IPv6 address
print("Calculating the EUI-64 IPv6 address")
# split MAC address into octets
mac_octets = mac_address.split(':')
# convert 7th bit to 1 to indicate a locally administered address
eui64_octet = int(mac_octets[0], 16) ^ 2
# construct EUI-64 address
ipv6_eui64 = f"{eui64_octet:02x}{mac_octets[1]}:{mac_octets[2]}ff:fe{mac_octets[3]}:{mac_octets[4]}{mac_octets[5]}"

# Get the IPv6 prefix of the network interface
print("Getting the IPv6 prefix of the network interface")
for interface in netifaces.interfaces():
    addresses = netifaces.ifaddresses(interface)
    if netifaces.AF_INET6 in addresses:
        for ipv6_addr in addresses[netifaces.AF_INET6]:
            if "addr" in ipv6_addr and not ipv6_addr["addr"].startswith("fe80"):
                # Check if the address has a scope and is global
                if "scope" in ipv6_addr and ipv6_addr["scope"] == netifaces.scope["global"]:
                    prefix = ipaddress.IPv6Address(ipv6_addr["addr"]).exploded.split("::")[0]
                else:
                    prefix = ipaddress.IPv6Address(ipv6_addr["addr"]).exploded.split(":")[:-4]
                break
prefix = ':'.join(prefix)
print(f"IPv6 prefix: {prefix} found")

# Form the complete IPv6 address by combining the prefix and EUI-64 address
ipv6_complete = f"{prefix}:{ipv6_eui64}"

# Ping the resulting IPv6 address
print(f"Pinging the resulting IPv6 address: {ipv6_complete}")
response = subprocess.call(["ping6", "-c", "1", ipv6_complete])

# Check the response status
if response == 0:
    print("The device with MAC address", mac_address, "responded at", ipv6_complete)
else:
    print("No response from the device with MAC address", mac_address, "at", ipv6_complete)
```

Listing A.1: Finding and pinging IPv6 devices.