Vulnerability Report – CVE-2018-17612

Certificate Management Vulnerability in Sennheiser HeadSetup

Hans-Joachim Knobloch, André Domnick
Secorvo Security Consulting GmbH

Version 1.2
Date October 31, 2018

Table of Contents

Timeline

Initial finding	July 20, 2018
Notification of the vendor about the findings and initial response from Sennheiser Support	July 23 to August 8, 2018
Initial publication in Secorvo Security News (only mentioning the addition of a root certificate, without further details)	September 3, 2018
Vendor response	September 13, 2018
Further investigation and PoC exploit	September 14 to 21, 2018
Notification of the vendor about version 1.0 of this report	October 11, 2018
Further communication with the vendor	October 26, 2018
Publication of this report	October 31, 2018
Vendor notification about upcoming remediations	October 31, 2018
Updated version of HeadSetup software and certificate removal script on Sennheiser website	announced to be released until week 48/2018

1 Introduction

The Trusted Root CA certificate store is the list of entities issuing certificates – e.g. for TLS servers or software publishers – that are trusted by the specific client system and whose certificates are generally accepted by the local system. It is in most cases maintained by the respective browser or operating system vendor (see e.g. [MS-TRCP2018]) adhering to the regulations of the CA/Browser Forum [CABF-BR2018]. Any trustworthy communication carried out by this local client – and, from a more general perspective, the backbone of trusted communication all over the Internet.

However, even though the trust system relies on the automatic management of the therein hold certificates, it remains possible for administrators and – unless administratively prohibited – users to delete or add certificates in the Trusted Root CA store of their systems. In spite of everything, the content of this store ought to be regularly inspected and controlled – something that users rarely find the time for doing so.

Upon such a rare inspection of the Trusted Root CA store, we stumbled across two unexpected root certificates. The issuer names in these two certificates indicated that they have a connection to the Sennheiser HeadSetup utility software installed on our systems in conjunction with the connected headsets of this manufacturer.

We found that – caused by a critical implementation flaw – the secret signing key of one of the clandestine planted root certificates can be easily obtained by an attacker. This allows him or her to sign and issue technically trustworthy certificates. Users affected by this implementation bug can become victim of such a certificate forgery, allowing an attacker to send e.g. trustworthy signed software or acting as an authority authorised by Sennheiser.

2 Background

The Sennheiser HeadSetup SDK supports the use of a locally connected headset by web-based softphones in a browser, loaded from a server web site via HTTPS.

According to [Senn2018], the way HeadSetup supports this application scenario is by opening a local secure web socket (WSS) through which the headset can be accessed from within the browser.

According to Sennheiser, the browser must be able to access this local web socket through a trusted HTTPS connection in order to bypass cross origin resource sharing (CORS) restrictions implemented by relevant browsers. Hence, the HeadSetup SDK needs a locally trusted TLS server certificate issued to the localhost IP address[1] (127.0.0.1) and the associated private key.

3 Vulnerability Remediation Status

Sennheiser was informed about this vulnerability in advance, is aware of the vulnerability impact and started working on an updated version of HeadSetup to resolve the issue. According to the developers, this process will take a while.

Nonetheless, we decided to publish information about the vulnerability and recommended mitigations as scheduled because an attacker looking at the right spot may find and exploit it

[1] Theoretically, the IPv6 localhost address or a valid DNS host name of the local system might be used in alternative to 127.0.0.1. However that would have no effect on the issue that we subsequently describe.

with ease. On the other hand, simple and effective mitigations for end users are available (see 7.2) and should be implemented in the meantime.

Sennheiser announced the release of an updated version of HeadSetup no later than week 48/2018 (end of November 2018). The updated software will remove critical certificates and ensure that installed certificates are deleted as part of the uninstallation process. For users who do not regularly update their HeadSetup software, a script will be made available on the Sennheiser website that uninstalls the most vulnerable certificate.

4 HeadSetup Behaviour

4.1 Older Version

The following behaviour can be observed in HeadSetup version 7.3.4903 (henceforth referenced as version 7.3 or “older version”). Potentially the same behaviour is also exhibited by several earlier and/or a few later versions.

HeadSetup 7.3 uses a single private/public key pair and associated certificate, which is identical for all installations of the software. The private key is stored in a file named `SennComCCKey.pem` and the certificate in a file named `SennComCCCert.pem`, both located in the HeadSetup installation folder, see Figure 1.

The certificate is self-signed. It uses issuer and subject name including a common name (CN) component `127.0.0.1` and is marked as a CA certificate in the standard certificate extension BasicConstraints, see Figure 2 and Figure 3. It is marked as valid until January 13, 2027.

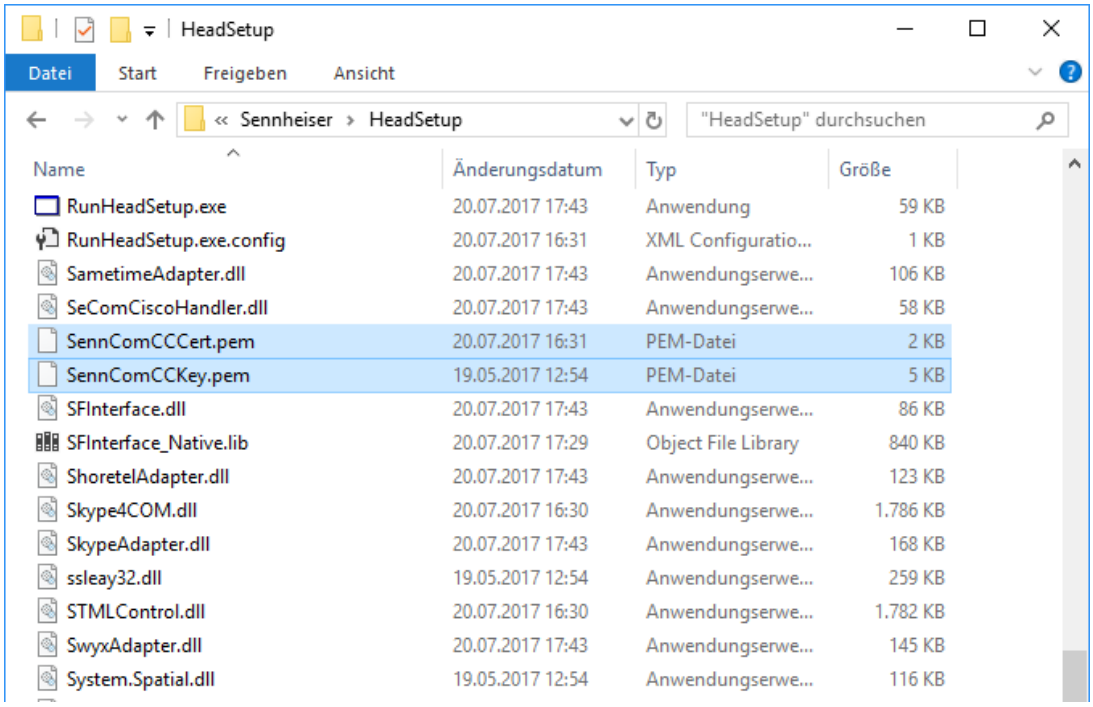

Figure 1 Certificate and key files in the older version

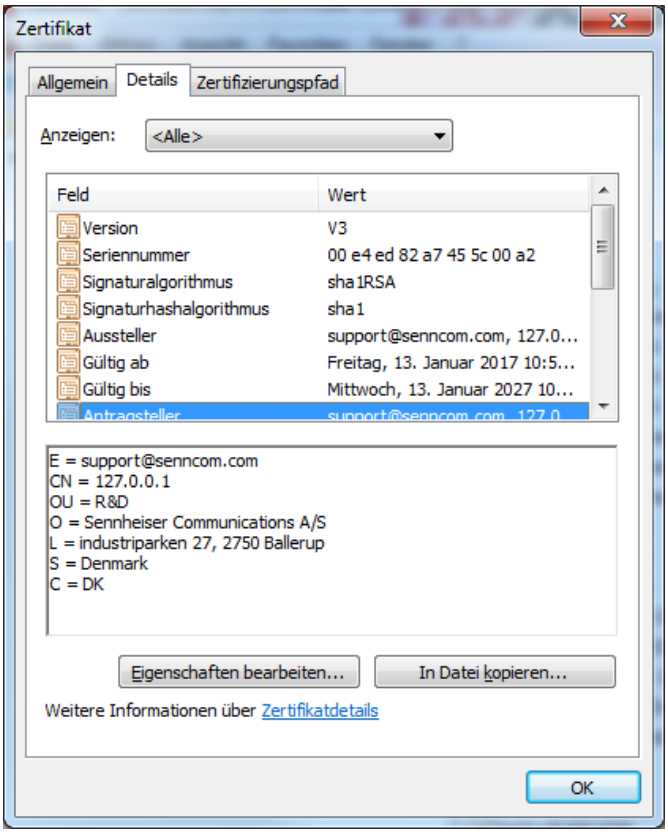

Figure 2 Subject and issuer name of the older version certificate

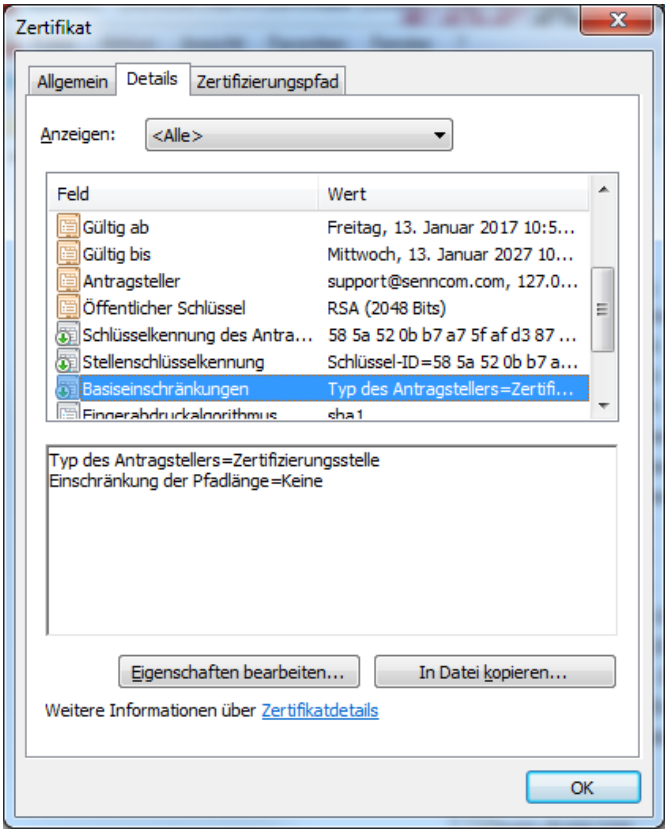

Figure 3 BasicConstraints extension of the older version certificate

Despite its designation as CA certificate, the HeadSetup software employs it as the TLS server certificate for the local secure web socket. In order to turn it into a trusted credential, the HeadSetup installer pushes the certificate into the local machine[2] trusted root certificate store of the Windows system on which it is installed.

Note that the HeadSetup installer must run with local administrator privileges. Once the installing user confirms[3] the installation of the software there is *no* further system prompt warning about the addition of the certificate to the trusted root store and displaying the certificate's fingerprint, like there would be if this root certificate were added manually.

4.2 Newer Versions

The following behaviour can be observed in HeadSetup versions 7.4.5601 and 8.0.6108 (henceforth referenced as version 7.4 respectively 8.0 or "newer versions"). Potentially the same behaviour is also exhibited by a few earlier and/or several later versions and any version in-between 7.4 and 8.0.

HeadSetup newer versions use two certificates:

a) A self-signed CA certificate using CN `SenncomRootCA`, marked as valid until July 28, 2037 and stored in a file named `SenncomRootCA.cert.pem` in the HeadSetup installation folder.
   The vendor supposedly keeps the private key associated with this certificate under custody.

b) A TLS server certificate using CN `127.0.0.1`, issued by the SenncomRootCA, valid until December 22, 2018 and stored in a file named `127.0.0.1.crt.pem` in the HeadSetup installation folder.
   The associated private key is stored in a file named `127.0.0.1.key.pem` in the HeadSetup installation folder.

Both these certificates and, of course, associated keys are identical for all installations of the software.

The HeadSetup software employs the latter certificate as the TLS server certificate for the local secure web socket. In order to turn it into a trusted credential, the HeadSetup installer pushes the SenncomRootCA CA certificate into the local machine trusted root certificate store of the Windows system on which it is installed.

Note that – like for the older versions– the HeadSetup installer must run with local administrator privileges and once the installing user confirms the installation of the software there is *no* further system prompt warning about the addition of the certificate to the trusted root store and displaying the certificate's fingerprint, like there would be if the root certificate were added manually.

---

2 Under Windows, the local machine trusted root store is also trusted by any user while she or he is logged on to that particular machine.

3 The HeadSetup installer carries a valid Authenticode signature by Sennheiser Communications, using a code-signing certificate issued by a public commercial trust center, which is displayed during the attended installation. That code-signing certificate has nothing to do with the vulnerability at hand.

4.3 Update or Removal of the Software

During an update from the old version to a newer version the certificate and key files containing the old version certificate and key in the HeadSetup installation folder are deleted and replaced by the newer version certificate and key files.

During removal (de-installation) of the HeadSetup software the complete HeadSetup installation folder – including certificate and key files – is deleted.

However, in neither case any of the CA certificates added to the local machine trusted root store during the installation or update process is removed.

5 Risk Assessment

5.1 Older Version Approach

Adding a Trusted Root CA certificate is a severe vulnerability, if a potential attacker has access to the associated private key. Such an attacker can issue forged certificates at his or her own discretion that will be automatically validated as valid and hence trusted on the affected vulnerable system.

In the case of the HeadSetup Software the root CA private key is potentially accessible to any user that has or ever had access to the HeadSetup 7.3 installer executable or an installed version, since it is stored in the `SennComCCKey.pem` key file.

Moreover, since the certificate is not removed from the trusted root certificate store during update or removal of the software, every system on which HeadSetup 7.3 was installed at any time in the past – and every user on such a system – remains vulnerable. This vulnerability persists until the CA certificate expires in 2027 or one of the mitigations recommended in section 7 becomes effective.

Possible Attacks, where such a forged/valid certificate could be employed include, but are not limited to the following two examples:

Read and modify the complete session of the victim with any seemingly secure HTTPS web server

The attacker might issue a TLS server certificate for an arbitrary server name or wildcard (e.g. CN=`*.google.com`) for a key under her or his control and install that in a man-in-the-middle system pretending to be the original secure web server which the victim on a vulnerable system wants to access.
Unless the browser implements a key/certificate pinning as an additional security measure – like some versions of Chrome do with regard to certificates for Google servers – the victim will not notice the man-in-the-middle attacker.

Send the victim malicious software or provide with a download link to malicious software seemingly coming from an arbitrary well-known software publisher

The attacker might issue a code-signing certificate for an arbitrary, well-known software publisher name (e.g. CN=`Microsoft Corporation`) for a key under her or his control and use that certificate to code-sign an installer executable containing malware.
If a victim on a vulnerable system opens that installer she or he is (in many cases, depending on local policy configuration) presented a prompt indicating the well-known software publisher as source of the executable and will more likely proceed with the installation of the software than if “Unknown Publisher” or an unexpected publisher name were indicated.

Note that in both exemplary attacks a connection the Sennheiser would hardly be noticed by an unsuspecting victim:
The victim would have to inspect the HTTPS server certificate respectively code signing certificate in a detail level that shows the root certificate to which the certificate in question is linked. Depending on the application, this requires anything from two to five additional, supposedly unnecessary clicks on buttons and options that are not commonly well-known. Hence, it is safe to assume that an overwhelming percentage of users will not perform such an inspection.

5.2 Newer Version Approach

In the newer version approach, there is no direct way for an attacker to create valid certificates under the new SennComRootCA CA certificate, since vendor supposedly keeps the private key associated with that CA under custody.

However, there is no published information about the policies according to which the SennComRootCA operates, such as a CP or CPS, nor any third party audit of its security and trustworthiness. Hence, users can only hope that Sennheiser Communications follows PKI best practices regarding the security and trustworthy operation of its CA.

In contrast to that, any CA that is pre-installed in a browser or operating system through programs such as [MS-TRCP2018] must publish a CP and CPS fulfilling the requirements of the CA/Browser Forum, in particular [CABF-BR2018], and prove their compliance with these requirements through a yearly audit report by an independent accredited auditor.

Therefore, we consider the risk that an attacker might fraudulently obtain a certificate significantly higher for the SennComRootCA than for other pre-installed Root CAs or their respective Sub CAs.

Additionally, [CABF-BR2018] strictly prohibits trust centers operating under one of the preinstalled Root CAs to issue TLS server certificates for non-routable IP addresses like 127.0.0.1. If an attacker needs a valid certificate for this address as one step of a multi-layered attack, he or she could not legally obtain it from a public trust center. However, such an attacker could directly (ab-)use the HeadSetup TLS server certificate and the associated private key, which will be valid on any system that ever had a newer version of HeadSetup installed.

Furthermore the – in comparison to the older version approach – significantly shorter validity of the TLS server certificate results in a potential availability risk.
The vendor can push out a new TLS server certificate through its update channel only. If certain users do not regularly update or do not install the respective upcoming update in time – to be precise: before the current TLS server certificate expiry on December 22, 2018[4] – their web-based softphones will no longer be able to connect to their headphones through the local HTTPS web socket.

Summarized, the vulnerability introduced by the newer version HeadSetup software is significantly less severe than in the older version. There are however some remaining risks in this approach; see section 7.1 for a recommended solution that avoids these remaining risks.

4 Incidentally, this date is the last Saturday before the Christmas holidays. So many users would probably not notice the effect of the certificate expiry before early January 2019.

5.3 CVE-2018-17612 and CVSS Score

This vulnerability is assigned the unique identifier CVE-2018-17612.

Due to the fact that any system on which the older version was ever installed remains vulnerable despite later updates or removal of the software, we rate the vulnerability according to the risk assessment of the HeadSetup 7.3 software.

The CVSS vector string is

```
CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:C/C:H/I:L/A:L/E:F/RL:W/RC:C
```

resulting in a CVSS Base Score of 7.5.

6 PoC Exploit

This section describes how we exploited the vulnerability introduced by HeadSetup 7.3 as proof-of-concept.

6.1 Extraction of the Private Key

The `SennComCCCert.pem` file contains the CA (and server) certificate of the older version software in the directly usable OpenSSL PEM format.

However, despite its name, the `SennComCCKey.pem` file is not a directly usable OpenSSL RSA key. Upon inspection, it is obviously base64-endoced. After decoding the resulting binary file starts with the characters `Salted_`. That file prefix indicates that the file was symmetrically encrypted using OpenSSL.

In order to decrypt the file we needed to know the encryption algorithm and key that the manufacturer used for encryption. Our first guess was that the vendor employed the common AES encryption algorithm with 128-bit key in CBC mode. In the HeadSetup installation directory, we found only one piece of executable code that contained the file name `SennComCCKey.pem`, a DLL file named `WBCCListener.dll`. We searched for “AES” in the strings contained in this DLL. The result is shown in Figure 4: there is indeed the algorithm identifier `aes-128.cbc`. We found the key that the vendor used in close proximity to that algorithm identifier, stored in clear in the code.

After AES decryption, the `SennComCCKey.pem` file becomes a standard OpenSSL PEM format RSA private key file, where the key is protected by an additional passphrase. We found that the passphrase was specified in the configuration file `WBCCServer.properties` (see Figure 5) and additionally in the strings within `WBCCListener.dll`.

Knowing the AES key to decrypt the OpenSSL private key file and passphrase to access that private key, we could use the standard OpenSSL command line to extract the CA certificate and private key into a PKCS#12 file.

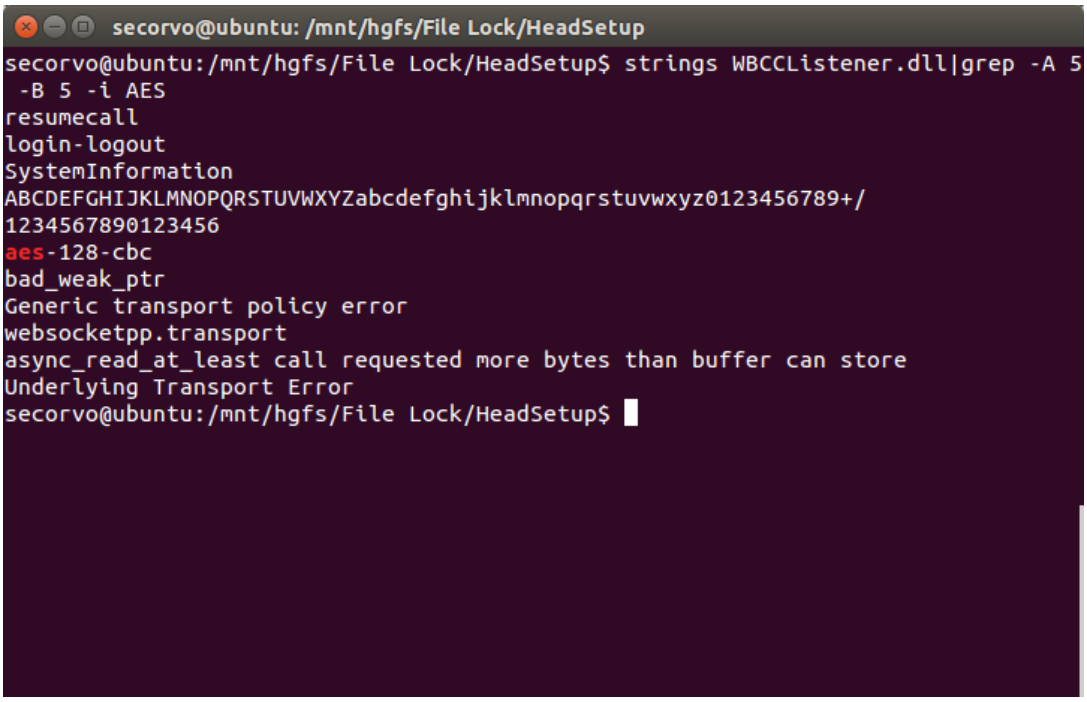

Figure 4 Strings within WBCCListener.dll

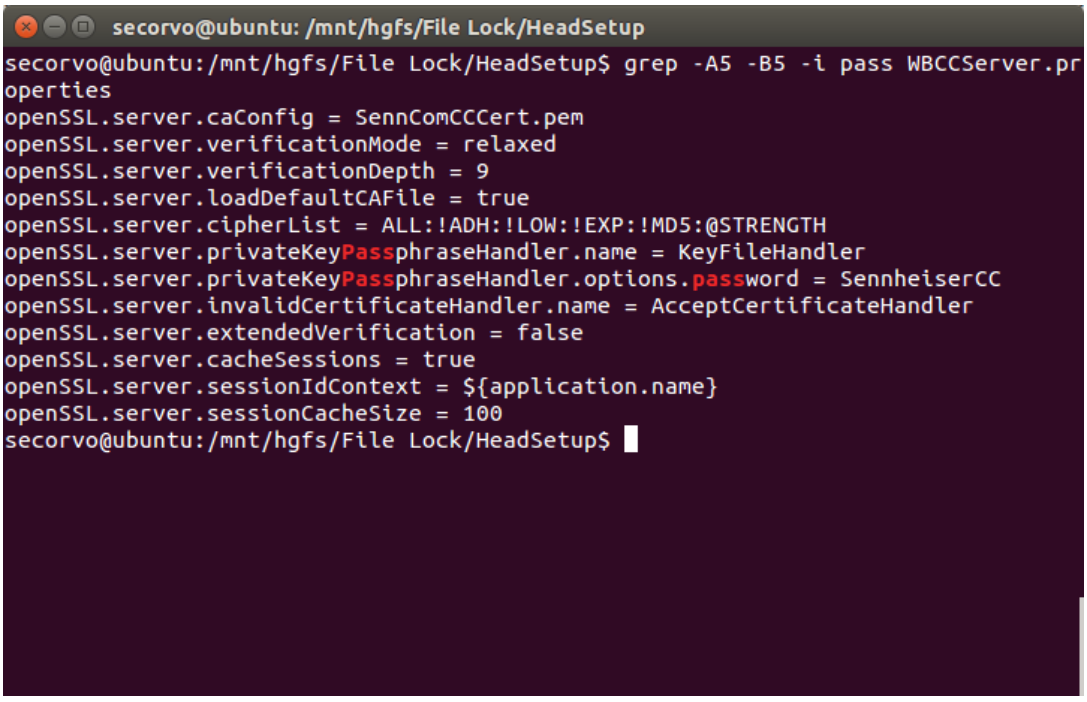

Figure 5 Configuration settings within WBCCServer.properties

6.2 Creation of a Fraudulent Trusted Server Certificate

We imported the PKCS#12 file with the older version HeadSetup key and certificate into a CA software (in our case XCA, however almost any other PKI CA software would work). Then we created a new key pair and used our fraudulent CA to issue a wildcard TLS server certificate for hosts in the domains of Google, Sennheiser and some of Sennheiser's competitors, see Figure 6.

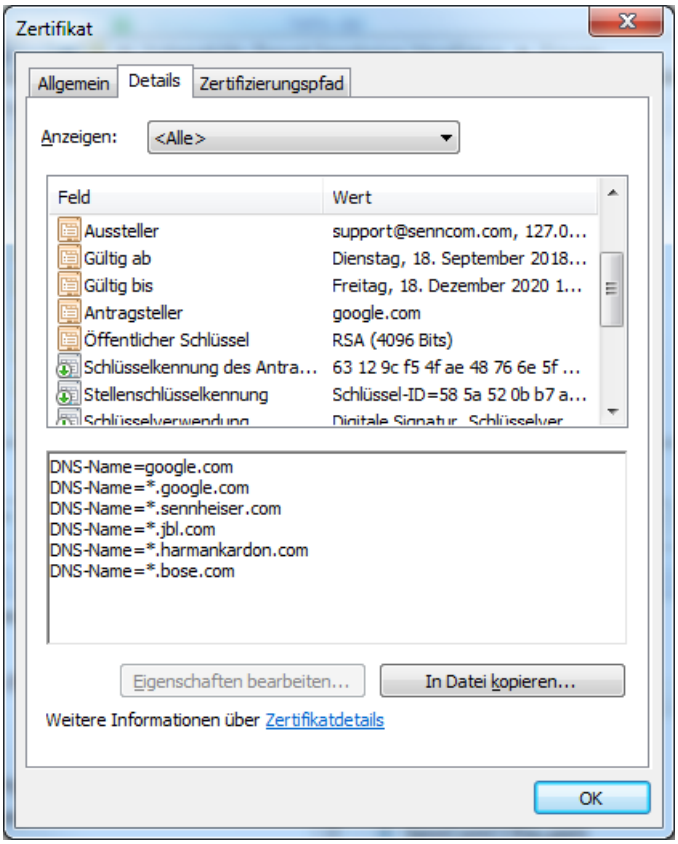

Figure 6 Fraudulently issued certificate

As mentioned above several times, every system that ever had HeadSetup 7.3 installed will validate this certificate as trusted until the year 2027[5].

6.3 Tools for Potential Abuse

To complete the exploit PoC, we imported the CA certificate and its associated private key into a readily available man-in-middle attack tool, in our case mitmproxy[6] and employed an ARP spoofing attack to redirect the victim traffic to our attacker proxy.

When visiting a TLS secured site the victim browser reported the connection as secure (see Figure 7) and the attacker was able to sniff and manipulate the whole connection.

5 In our PoC, we deliberately chose to limit the server certificate validity to 2020; we could of course issue it with a validity date equal to the older HeadSetup CA certificate.

6 See https://mitmproxy.org/

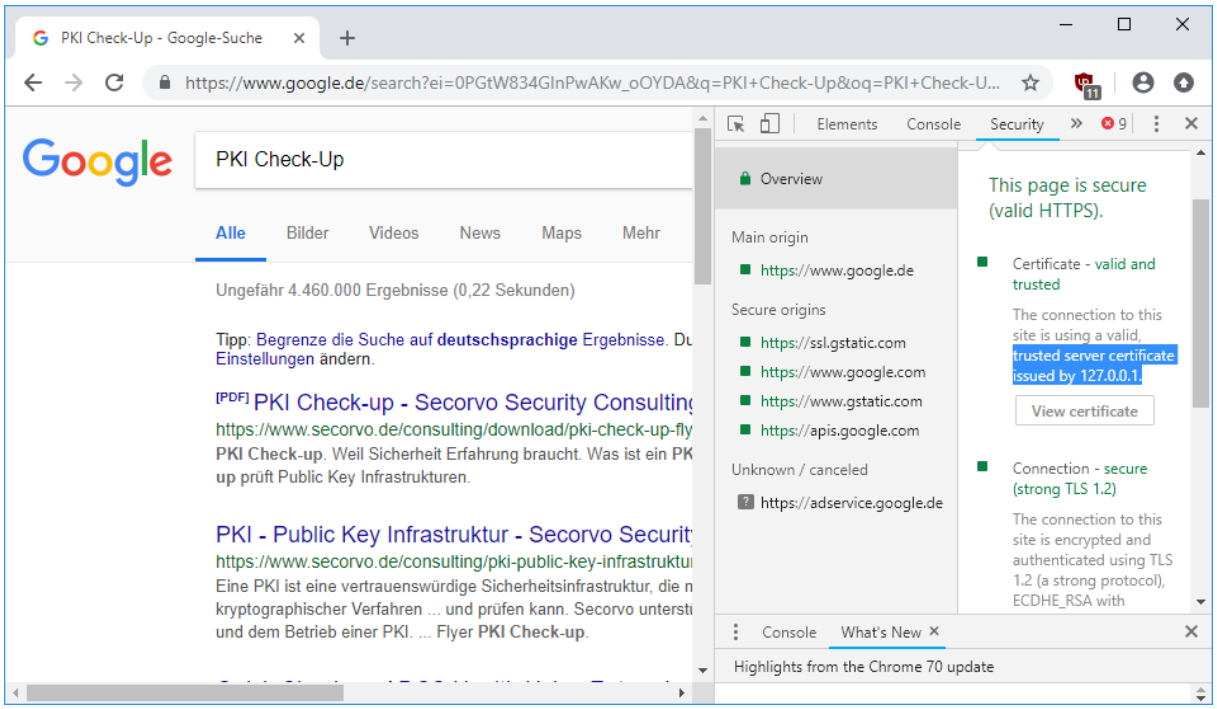

Figure 7 Browser accepts the attacker certificate[7]

7 Recommendations

7.1 Recommended Solution the Original Issue

Our recommended solution to procure a locally trusted TLS server certificate for the local secure web socket that eliminates all the described vulnerabilities is:

During installation of the software:

- Create a new public/private key pair that is individual for each installed instance of the software.
- Create an associated self-signed certificate for CN `127.0.0.1` which is marked as TLS server certificate, but *not* as CA certificate, using the respective standard certificate extensions.
- Push that certificate to the local machine trusted people certificate store (*not* the trusted root store).

Then the certificate will be trusted on the local machine, but cannot be used to issue other valid (but fraudulent) certificates.

Depending on whether the software regularly runs with local administrative privileges, the self-signed certificate can either be regularly renewed and the renewed certificate pushed to the trusted people store. Or it can be issued for a validity period of multiple years during installation. In both ways, the certificate renewal availability risk of the HeadSetup newer version approach is avoided.

7 Note that the right half of the browser window, where “issued by 127.0.0.1” is indicated, is the browser’s developer console that will not be displayed during regular use.

As an additional practical benefit, this solution would eliminate the effort for secure and trustworthy operation of the SennComRootCA.

7.2 Risk Mitigation by Users

7.2.1 Individual Users

Individual users need local Windows administrator privileges to remove the CA certificates added by the older and/or newer HeadSetup versions from the trusted root certificate store of their machine. The certificates can either be deleted using the certificate management MMC (on Windows 10: `certlm.msc`), by using PowerShell to access the certificate store `cert:LocalMachine\Root`, or by issuing the following commands on the command-line:

```
certutil -delstore root "127.0.0.1"
certutil -delstore root "SennComRootCA"
```

Users who need the local web socket functionality for web-based softphones should update HeadSetup to the newest version and only remove the older CA certificate (CN=`127.0.0.1`). For all other users we recommend removing both certificates.

7.2.2 Active Directory Administration

In an Active Directory, the AD administrator can use a Group Policy to add one or both to the “Untrusted Certificates” store of all AD-joined systems. The respective Group Policy setting can be found in the `Computer Configuration` section of a suitable GPO under `Policies\Windows Settings\Security Settings\Public Key Policies\Untrusted Certificates`, see Figure 8. One or both certificates must be imported to that GPO branch.

After Group Policy renewal, these certificates will be listed in the local machine untrusted certificates store of each system. This listing in the untrusted store has precedence, even if the certificates remain in the trusted root certificate store.

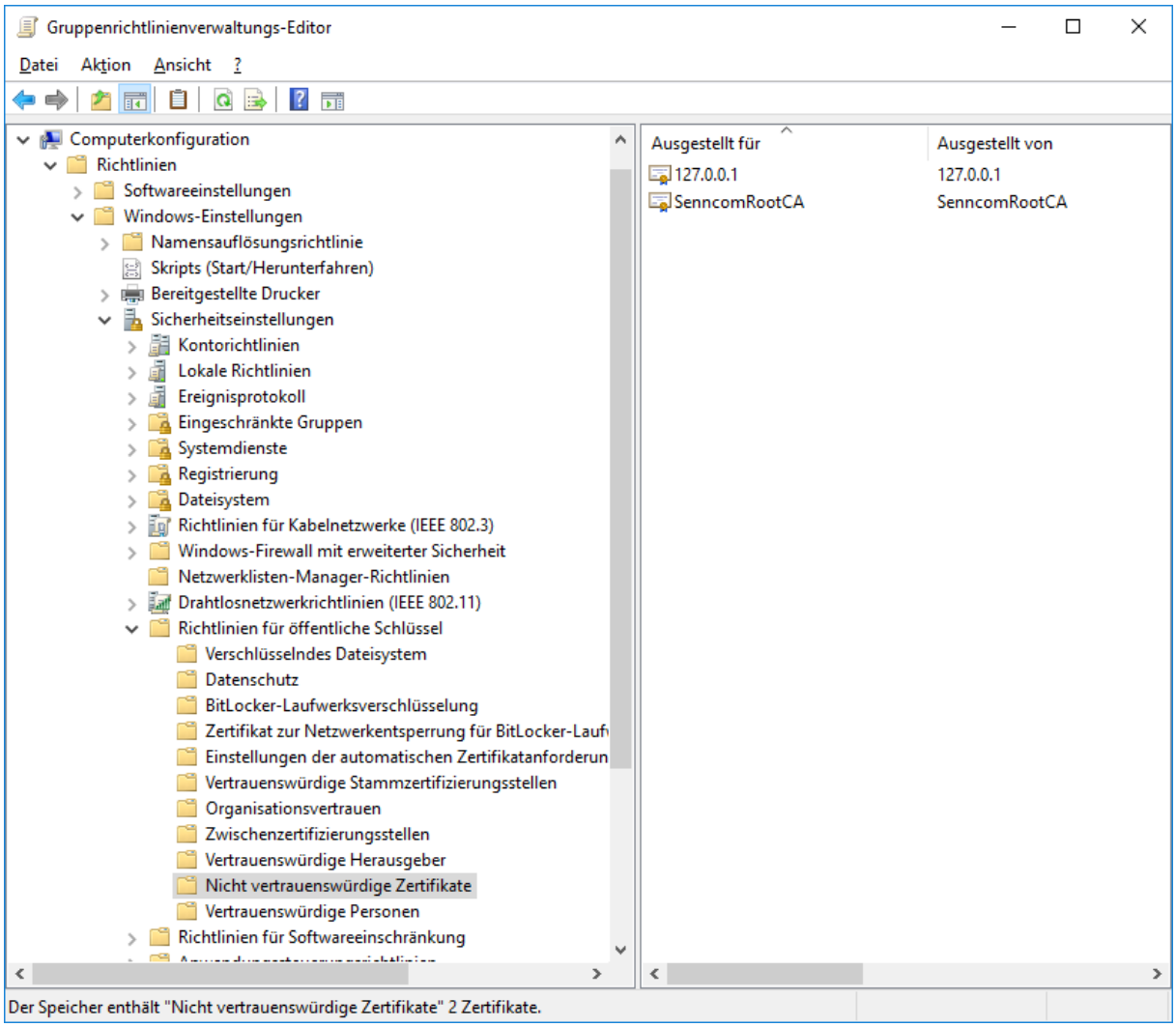

Figure 8 Group Policy setting to distrust certificates

We recommend organizations who need the local web socket functionality for web-based softphones to update HeadSetup to the newest version and distrust the older CA certificate only. All other users should distrust both certificates.

7.3 Risk Mitigation by the Vendor

7.3.1 Warning and User Decision

Unless the vendor implements the recommended solution in section 7.1, any new version of the HeadSetup software should warn the user during installation about the addition of a Trusted Root CA and permit users who do not need the local web socket to skip this step.

7.3.2 Trusted Root Certificate Removal

Any future version of HeadSetup should remove the old version CA certificate from the trusted root store, if present, during installation or update.

If the vendor implements the recommended solution in section 7.1, any future version of the HeadSetup software should also remove the newer version CA certificate from the trusted root store, if present.

In all cases, any future version of HeadSetup should remove both CA certificates from the trusted root store during de-installation.

7.3.3 Propagation as Untrusted via Microsoft Update

Since affected systems stay vulnerable even after removal of the HeadSetup software, any mitigations provided by future versions of the HeadSetup software cannot mitigate the risk for all affected users, in particular not for those users who will not install future versions.

Therefore, Sennheiser should contact Microsoft and request that the older version CA certificate (CN=`127.0.0.1`) is pushed to the untrusted certificates store (see also section 7.2.2) of all Windows systems by the Windows Update mechanism.

Microsoft has done so before for certificates that were fraudulently obtained or CA certificates of compromised CAs. In Windows 7, certificates in the untrusted store can be examined by local users; in newer versions of Windows only the fingerprints of untrusted certificates are stored.

7.4 Prevention of this Class of Vulnerabilities

The certificate management vulnerability described in this report was introduced during the design phase of the HeadSetup software. In order to prevent this class of vulnerabilities, we recommend the following generic steps:

- Documentation of a security concept for the software, independently reviewed before starting the implementation.
- In an application software, middleware, or utility (i.e. everything except system software) making changes to system-wide security settings such as fiddling with AD administrative credentials or – in the case at hand – adding trusted Root CAs is to be avoided.
- If doing so is deemed absolutely necessary, an expert in the field should be consulted before concluding the software design.

References

[CABF-BR2018] CA/Browser Forum: Baseline Requirements, https://cabforum.org/baseline-requirements-documents/, retrieved September 26, 2018

[MS-TRCP2018] Microsoft: Trusted Root Certificate Program Requirements, https://technet.microsoft.com/en-us/library/cc751157.aspx, retrieved September 26, 2018

[Senn2018] Sennheiser: Cross origin resource sharing (CORS) and Self-Signed Certificates Explained, 2018, obtained through direct communication

Acronyms

AD	Active Directory
AES	Advanced Encryption Standard
ARP	Address Resolution Protocol
CA	Certificate Authority
CBC	Cipher Block Chaining
CN	Common Name
CORS	Cross Origin Resource Sharing
CP	Certificate Policy
CPS	Certification Practice Statement
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DNS	Domain Name Service
GPO	Group Policy Object
HTTPS	HyperText Transfer Protocol via SSL/TLS
IP	Internet Protocol (version 4)
IPv6	Internet Protocol version 6
MITMf	Man-In-The-Middle framework
PKCS#12	Public Key Cryptography Standard no. 12
PEM	Privacy-Enhanced Mail
PKI	Public Key Infrastructure
PoC	Proof-of-Concept
RSA	Rivest, Shamir, Adleman crypto algorithm
SDK	Software Development Kit
SSL	Secure Socket Layer
TLS	Transport Layer Security
WSS	Secure Web Socket