All your Calls Are Still Belong to Us

Authors: Daniel Mende, Enno Rey, Christopher Werny

ERNW
providing security.

Table of Contents

1 INTRODUCTION

The present paper provides an overview of security problems frequently found in current VoIP environments. There's a common misconception that encryption alone can provide sufficient protection from attacks in such settings. This might fail – as we'll lay out – for two main reasons. First encryption is only one part in an overall strategy to protect infrastructures processing sensitive data or offering critical services, so neglecting other elements of an overall infrastructure security approach will inherently lead to vulnerabilities. Second, as always, sufficient protection by cryptographic means, quite obviously depends on the quality of the crypto implementation. We will show that of one major vendor (Cisco) disposes of several architecture level flaws that, when exploited, will render major parts of a VoIP deployment vulnerable.

The paper is organized into three main sections. We start with a short overview of elements of an infrastructure security strategy and their meaning for VoIP environments. We then present an overview how Cisco CUCM handles encryption. In the last part we discuss a potential attack against the Cisco VoIP crypto framework.

2 THE SEVEN SISTERS OF INFRASTRUCTURE SECURITY

These are a number of fundamental security principles which can be applied to any complex infrastructure, be that a network, a building, an airport or the like. This section provides a short overview as for their meaning and security benefits.

These seven controls are:

- Access Control
- Isolation
- Restriction
- Encryption
- Entity Protection
- Secure Management
- Visibility

2.1 Access Control: "try to keep the threats out of the environment containing the assets to be protected at all".

This should pretty much always be an early consideration as limiting access to "some complex infrastructure" obviously provides a first layer of defense and does so in a preventative[1] way. Usually authentication plays a major role here. Please note that in computer networks the access control principle does not only encompass "access to the network [link]" (where unfortunately the most prevalent technology – Ethernet – does not include easy-to-use access control mechanisms. And, yes, we are aware of 802.1X...) but can be applied to any kind of ("sub-level") communication environment or exchange. Taking a "passive-interface" approach for routing protocols is a nice example here as this usually serves to prevent untrusted entities ("the access layer") from participating in some critical protocol [exchange][2] at all.

In a VoIP scenario limiting who can participate in the various layers and communication exchanges, be it by authentication, be it by configuration of static communication peers for certain exchanges[3] (this might not scale and usually has a bad *operational feasibility*) would be an implementation of the *access control* principle.

---

[1] *In general preventative controls have a better cost/benefit ratio than detective or reactive ones. And this is still true in the "you'll get owned anyway that's why you should spend lots of resources on detective/reactive controls" marketing hype age…*

[2] *To provide another example from the routing protocol space: the "inter-operator trust and TCP-" based nature of BGP (as opposed to the "multicast and UDP-"based nature of other routing protocols) certainly is one of the most fundamental stability contributing properties of the current Internet.*

[3] *Another simple example here. If the two VoIP gateways in the incident described here [http://www.ernw.de/content/e15/e26/e1342/download1344/ERNW_Newsletter_26_VoIP_Sec_ger.pdf] had used a host route for each other instead of their default route (which wasn't needed given their only function was to talk to each other), presumably the whole thing wouldn't have happened.*

2.2 Isolation: "separate some elements of the environment from others, based on attributes like protection need, threat potential or trust(worthiness)".

In computer networks this one is usually implemented by network segmentation (with different technologies like VLANs or VRFs and many others) and it's *still* one of the most important infrastructure security principles. We mean, can one imagine an airport or corporate headquarters without areas of differing protection needs, different threat exposure or separate layers and means of access?

Again, it should be noted that "traditional network segmentation" is only one variant. Using RFC 1918 (or ULA, for that matter) addresses is some parts of a network without NATing them at some point, or *refraining* from route distribution at some demarcation point constitute other examples.

In the VoIP world the main realization of the isolation principle is the commonly found approach of "voice vs. data VLAN[s]".

2.3 Restriction: "once [as of the above principle] isolated parts get connected try to limit the interaction between those parts at the intersection point".

This is the one most people think of when it comes to network security as this is what the most widely deployed network security control, that is firewalls, is supposed to do.

Two points should be noted here, from our perspective:

In some network security architecture documents phrases going like "the different segments are [to be] separated by firewalls" can be found. Which, well, is a misconception: usually a firewall *connects* networks (which would be isolated otherwise), it does *not* separate them. It may (try to) limit the traffic passing the intersection point but it still is a *connection element*.

And it should be noted that the restriction it applies (by *filtering* traffic) always has an operational price tag. Which is the one of the reasons why firewalls nowadays tend to fail so miserably when it comes to their actual security benefit...

In VoIP networks taking the restriction approach of is considerably hard (and hence quite often simply doesn't happen) given a number of protocols' volatility when it comes to the (UDP/TCP) ports they use.

2.4 Encryption: "while in transit encrypt some asset to protect it from threats on its [transit] way."

Again, this is a very common infrastructure security control (alas, at times the only one people think of) and probably does not need further explanation here.

Still it should be noted – again – that it has an operational price tag (key management and the like). Which – again – is the very reason why it sometimes fails so miserably when it comes to providing actual security...

In the VoIP world (as this one is very much about "assets in transit") it's (nowadays) a quite common one, even though still a number of environments refrains from using it, mainly due to the mentioned "operational price tag".

2.5 Entity protection: "take care of the security exposure of the individual elements within the environment containing the assets to be protected".

This encompasses all measures intended to increase the security of individual elements. It's not limited to simple hardening though, but includes all other "security [posture] quality assurance" things like pentesting or code reviews (when the element looked at is an application).

Adding a comment again I'd like to state that, in times of virtualization and vaporizing security layers (deploying shiny apps pretty much directly connecting customers to your ERP systems, by means of fancy webservices) this one might become more and more important. In the past many security architectures relied on layers of isolation & restriction and thereby skipped the hardening/quality assurance step ("we don't have to harden this Solaris box as there's a firewall in front of it").

As we'll lay out in this paper and the next chapter's case studies this one is a fundamental (and overlooked one) in many VoIP deployments.

2.6 Secure management: "manage the [infrastructure] elements in a secure way".

Secure management usually can be broken down to

- Restrict the endpoints allowed to establish management connections.
- Either use a trusted environment (network link) or use secure variants of mgmt. protocols instead of their less secure counterparts (SSH vs. Telnet, HTTPS vs. HTTP, SNMPv3 vs. community-based SNMP and the like).
- Require sufficient authentication (as for methods, authenticator [e.g. password] quality, personalized accounts etc.).
- Logging of security related events and potentially all management actions performed.

While this is (should be) an obvious security principle, daily assessment experience shows that failures/weaknesses in this space account for the majority of critical vulnerabilities when it comes to infrastructure security.

This applies in particular to VoIP implementations (see below for examples).

2.7 Visibility: "be able to assess the current security posture of your infrastructure and its elements with reasonable effort".

This is where logging (+ analysis), monitoring etc. come into play. We'd like to note that while this is a valid infrastructure security principle, its actual security benefit is often overestimated given the "detection/reaction" nature of this principle and its subsequent bad operational feasibility[4].

Furthermore it is a particularly interesting (and neglected) one in many VoIP environments. Usually the data generated in this space (for VoIP) can not be easily processed (by $SIEM one acquired two years ago, for a six-figure € number and which still has only a handful of use cases defined…), while on the other hand being heavily useful (or even required for legal follow-up) in one of those numerous billing fraud incidents.

[4] *As it requires the usually most scarce resource of an organization, that is humans and their brains. The part that can not be easily substituted by technology…*

2.8 How to Apply those Principles in a Generic Way

Looking at these fundamental security principles allows for tackling any type of "securing assets within a complex overall setting" by going through a simple (checklist-type) set of questions derived from them. These questions could look like

- Can we limit who's taking part in some network, protocol, technology, communication act?
- Any need to isolate stuff due to different protection need, (threat) exposure or trust(worthiness)?
- What can be done, filtering-wise, on intersection points?
- Where to apply encryption in an operationally reasonable way?
- What about the security of the overall system's main elements?
- How to manage the infrastructure elements in a secure way?
- How to provide visibility as for security-related stuff, with *reasonable effort*?

2.9 Some Case Studies

The talk presents a number of case studies from pentests we recently performed. These show that, even though good crypto was used in some VoIP implementations, the lack of consideration of other infrastructure controls pretty much always led to severe business risks. Please see the presentation's slides for the case studies.

3 The Cisco Unified Communications Manager Crypto System

This section describes the basic functionality of the Cisco Unified Communications Manager Crypto System, which components are involved and how they interact between each other.

Before we go into details how the different components interact, a brief overview which components will be used for the Cisco Unified Communications Manager (CUCM from here on) crypto approach will be helpful.

3.1 Mode of Operation of the CUCM

The CUCM can be operated in two different modes. The first mode is called "non-secure" mode. As the name implies, it does not offer any authentication or encryption services for the environment. This is the default mode when one installs the CUCM. The second mode is the so called "Mixed Mode". In this mode the CUCM offers authentication and encryption capabilities for IP phones, trunks, CTI et al. In order to activate this mode, one has to acquire two Cisco security tokens (which are just relabeled Aladin Tokens). These tokens are portable security modules that contain a private key and an X.509v3 certificate which is signed by the Cisco certificate authority.

3.2 Certificate Trust List

The Certificate Trust List (CTL) is the root of the whole trust chain used in the crypto system of the CUCM. The CTL contains a server certificate, public key, serial number, signature, issuer name, subject name, server function, DNS name and the IP address for each server in the Unified Communication environment. The CTL contains entries for the following servers and security tokens:

- Security Token of the systems administrator
- CUCM and TFTP services
- Certificate Authority Proxy Function (CAPF)
- TFTP Servers
- ASA Firewalls

The CTL file must be created by an administrator in order to activate the "Mixed Mode" of the CUCM. The CTL file itself is signed by the private key stored on the security token. In order to generate this CTL file, one has to install the Cisco CTL Client on an MS Windows system and add all the necessary certificates. After successful creation of the file, one can configure all the whistles and knobs to provide authentication and encryption for your Unified Communications environment.

3.3 Certificate Authority Proxy Function

The CAPF functionality is installed on the CUCM in the default installation, but is deactivated. The Certificate Authority Proxy Function is a process, by which a phone can request a LSC (locally significant certificate) from the CUCM CAPF functionality. The certifcate will be later used for secure signaling (via TLS) and for encrypting the payload of the voice calls. After activation of the service, CAPF automatically generates a key pair and a certificate that is specific to the CAPF function. This certificate is integrated in the CTL File so that the IP Phones trust this certificate. The CAPF function supports the integration external CAs to issue company certificated to the IP Phones. Furthermore the CAPF function performs the following tasks:

- It can authenticate the IP Phones via an existing Manufactured Installed Certificate (MIC), Locally significant Certificate (LSC), randomly generated authentication string or a "null" authentication.
- As mentioned before, CAPF can issue LSCs to IP phones.
- Upgrading existing LSCs.

3.4 Transport Layer Security

In a Cisco Unified Communications environment, TLS provides secure data transfer between systems or devices. TLS secures the connections among CUCM managed devices and processes, to prevent access to the voice domain. TLS can be used in a UC environment to secure SCCP calls to phones and secures SIP calls to IP Phones or SIP Trunks.

3.5 Phone Certificate Types

Cisco IP Phones are using two different certificate types :

- Manufacture-installed certificate (MIC): This certificate is installed on the phone during the manufacturing process. These certificates will be used to authenticate against the CAPF Service in order to install a LSC. This certificate can not be overwritten or erased.
- Locally significant certificate (LSC): This type of certificate is installed on the phones via the CAPF function. After installation, the LSC secures the connection between the CUCM and the IP Phone after the security mode for the IP Phone is configured.

3.6 Secure Phone Profile

CUCM groups all security-related settings for a phone type and protocol into security profiles. This allows to configure the profile once, and apply it to multiple phones. These settings include the device security mode, digest authentication, and some CAPF settings. These configuration parameters include, but are not limited to:

- Signaling encryption
- Authentication
- Voice Payload encryption

•

3.7 Interaction between main components

Before one can configure anything security related, one has to activate the security mode for the whole CUCM Cluster or the standalone server if only one CUCM is present. In order to so, one has to buy two of the aforementioned tokens. In the CUCM Administration GUI, one must install the the Cisco CTL Client on an MS Windows machine. The CTL Client is used to generate the CTL File which the phones will use to "know" which certificates they can trust. When the CTL is initialized, one has to insert the security token in a local USB port and add the existing certificate to the CTL file. This process has to be repeated for the second token as well. After the CTL File is created, the CUCM operates now in mixed mode, which means one can now configure the security related steps. In order to apply the security settings to the phone, one has to configure a secure phone profile. In this profile, all the relevant security configuration takes place. When the configuration is done, the phone resets and requests the CTL File on bootup from the CUCM. After the CTL file is downloaded it connects to the CAPF service to get a LSC. When the IP Phone does not have a LSC, the phones authenticates to the CAPF service via the MIC. The Phone generates a public and private key, and forwards the public key to the CAPF service. The CAPF service forwards a PKCS#10 Certificate Signing Request to the external CA (if applicable). The CAPF function signs the phone certificate and sends it back to the phone in a signed message. If this process is completed, all configured security parameters in the phone secure profile can be used. The phone registers to the CUCM over SIP TLS and encrypts the Payload of the voice calls. In addition, the configuration and firmware files of the phone are also encrypted (if configured in the profile)

4 EXPLOITING THE TRUST RELATIONSHIP IN CISCO UNIFIED COMMUNICATIONS MANAGER

The Achilles heel of the whole CUCM Security Model is the Certificate Trust List, which can potentially be subverted. The following outlines the details of such an attack.

4.1 The Role of the CTL in more Detail and What Can Go Wrong

In the beginning of the boot sequence the VoIP Phone – the Cisco Unified IP Phone as well as the Cisco IP Communicator – requests a root of trust from the selected CUCM, the Certificate Trust List. This file is requested at the first boot up to get an initial state of trust and on each following boot process to check for updates in the Trust List.

If this process is intercepted, by a Man-in-the-Middle attack, it is possible to replace the original Certificate Trust List with a modified one. The VoIP Phone should recognize the modification, at the initial boot up it should verify the validity of the signing certificate in the Certificate Trust List against the MIC's signing Certificate, at every other boot process the validity of the new Certificates Trust List's signing certificate should be checked against the old installed CTL, but the VoIP Phone fails to do so in both cases.

In the first case, the CTL enrollment at the initial boot up, it seems as there is no signer validation at all, the Certificate Trust List is just accepted by the phone. In the second case, the CTL update at each other boot process, the modified CTL is rejected at the first try, but accepted at the second try and also the old and valid Certificate Trust List is discarded.

Once this root of trust is compromised, all further authentication and encryption based on the CTL can become invalid. This includes the signed and encrypted firmware and configuration update as well as the TLS secured CAPF and SIP communication.

The following screenshots are taken from a sample attack on the trust relationship. The setup for the attack includes/requires:

- Attacker in Man in the Middle possition between the Softphone and the CUCM by ARP spoofing
- All traffic was forwarded, except for TFTP and SIP-TLS, which both were terminated locally.
- A tool called 'ctl_proxy' was used maipulate the CTL and all signed files on-the-fly

As seen on the next two screenshots, the softphone accepts the modified CTL without any warning or error, the logfile shows that the Softphone doesn't even recognize the CTL beeing changed:

Softphone accepts modified CTL without any complaint

```
(  1408) tftpDownload : server 10.10.20.1, srcFile CTLSEPSP95548010.tlv, destFile C:\Documents and Se
(  1408) tftpDownload : attempting TFTP download of file <tftp://10.10.20.1/CTLSEPSP95548010.tlv> to
(  1408) tftpDownload : Successful TFTP download of file <CTLSEPSP95548010.tlv>
(  1408) tftpDownload : return:1 with status=0
(  1408) downloadFile : return:1 with status=0
(  1408) tftpRead : return: 0
(  1408) authenticating ctl file
(  1408) parseHdr(): start of pad ('T' 0x0d) at TLV 15
(  1408) parseHdr(): hdr ver 1.2 (knows upto 2.0)
(  1408) parseHdr(): skipping 3 trail bytes (pad and/or unknown TLVs)
(  1408) parseHdr(): start of pad ('T' 0x0d) at TLV 15
(  1408) parseHdr(): hdr ver 1.2 (knows upto 2.0)
(  1408) parseHdr(): skipping 3 trail bytes (pad and/or unknown TLVs)
(  1408) validate_file_envelope: File sign verify SUCCESS; header length <304>
(  1408) CTL_ValidateSignedCTL: new CTL matches old, not updating
(  1408) finished CTL update
(  1408) got CTL, no updates
(  1408) setting CTLstatus=2
(  1408) exiting SECUpdateCTL() - succeeded (no changes), rc=<0>
(  2964) -VM| main|cip.cfg.ConfigManager:? - ConfigManager updateCTL() rc=0 retryCount=0
(  2964) main|cip.cfg.ConfigManager:? - ConfigManager updateCTL() rc=0 retryCount=0
(  2964) -VM| main|cip.cfg.ConfigManager:? - CTL and/or ITL updated
(  2964) main|cip.cfg.ConfigManager:? - CTL and/or ITL updated
(  1408) cip_sec_NativeSecurity - getCTLInfo()
(  1408) entering SECGetCTLInfo()
(  1408) trying to get CTL info
(  1408) phone has CTL
(  1408) parseHdr(): start of pad ('T' 0x0d) at TLV 15
(  1408) parseHdr(): hdr ver 1.2 (knows upto 2.0)
(  1408) parseHdr(): skipping 3 trail bytes (pad and/or unknown TLVs)
(  1408) exiting SECGetCTLInfo() - setCtlItem(IIIILjava/lang/String;)V method lookup success
(  1408) trying to get CTL item
(  1408) ***** CTL item *****
(  1408) index    = 0
(  1408) hasItem  = 1
(  1408) itemRole = 0
(  1408) ipAddr   =
(  1408) dnsName   =
(  1408) hasCert  = 1
(  1408) ***** CTL item *****
(  1408) trying to get CTL item
(  1408) ***** CTL item *****
(  1408) index    = 1
(  1408) hasItem  = 1
```

Softphone logfile shows the modified CTL being successfully validated

Once the phone accepts the modified CTL-File, the root of trust is compromised, meaning a Man-in-the-Middle attacker can do all kind of nasty things like:

- Deliver own configuration files.
- Modify original configuration files.
- Akt as CAPF and mess with the Phones certificat (e.g. delete the installed one, replace the installed one).
- Deliver manipulated .jar files for code execution on the phone.

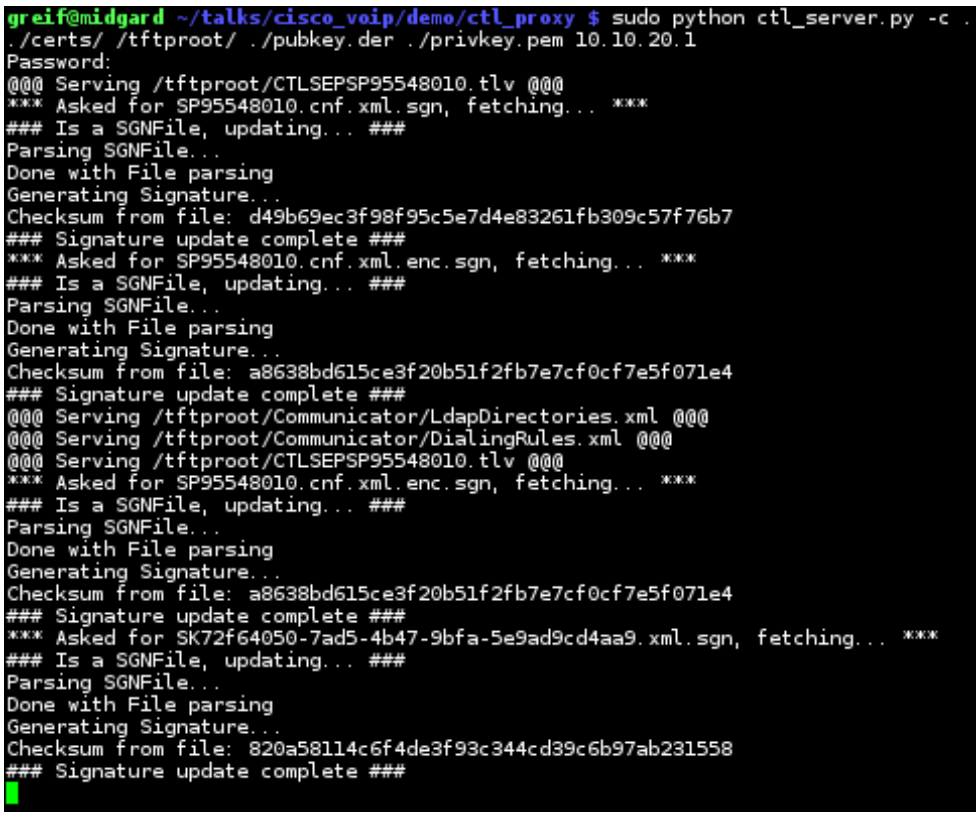
4.2 The ctl_proxy tool

This small python script implements an TFTP server to serve modified CTL files and signed files to cisco VoIP Hard- and Softphones. The tool is transparent from an TFTP view, so if a phone requests a special file, this file is fetched from the CUCM TFTP server and served to the phone afterwards. Only if the requested file is either a CTL-File or a signed .sgn file, the tool manipulates the file and serves the manipulated copy to the phone. On CTL-Files, the public keys of the CUCM Certificate, the CAPF Certificate and the Signing Certificate are exchanged by a given public key. The modified CTL-File is then signed again with the given private key and delivered to the phone. The steps are in detail:

- Exchange the public key of the signing certificate
- Exchange the public key of the CAPF and CUCM certificates
- Remove the signature TLV
- Generate the checksum of the modified CTL
- Encrypt the checksum with the private key
- Insert the new signature TLV

If the phone requests a signed .sgn file, the tool again fetches this file from the CUCM TFTP server and replace the signature with a signature that the phone can validate with its modified CTL, which means the signature of the .sgn file is re created with the given private key.

The following screenshot below shows the ctl_proxy tool in action:



5 CONCLUSIONS

Comprehensive security in VoIP deployment requires more than just using encryption. Furthermore one should be aware that vendors' crypto implementations might have weaknesses leading to exploitable conditions. Special care must hence be taken of the overall management and provisioning processes in VoIP deployments.