Prof. Dr. Steffen Wendzel

Modul 63515

Information Hiding

Lehrveranstaltung

Network Steganography

LESEPROBE

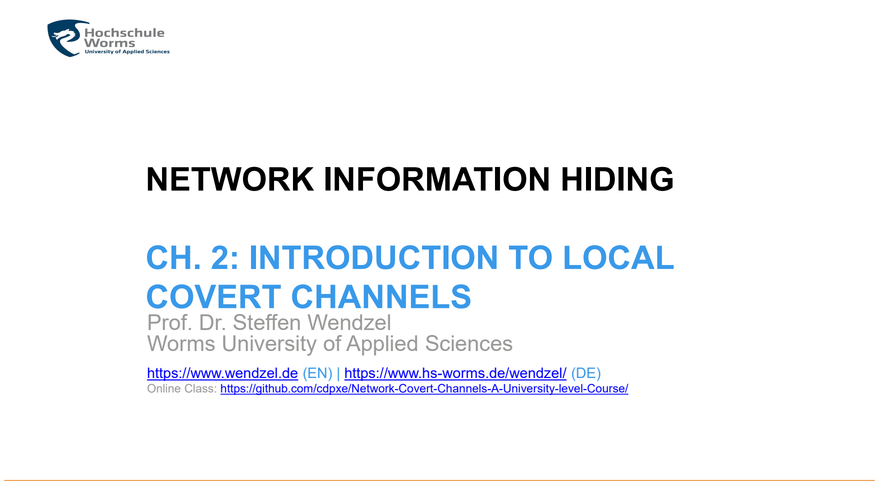
Fakultät für
Mathematik und Informatik

FernUniversität in Hagen

Leseprobe Kurs 01731 Network Steganography

Bei diesem Kurs werden aufgezeichnete Vorlesungen benutzt.

Als Leseprobe finden Sie die transkribierten Folien der einführenden Vorlesung.



Hochschule Worms University of Applied Sciences

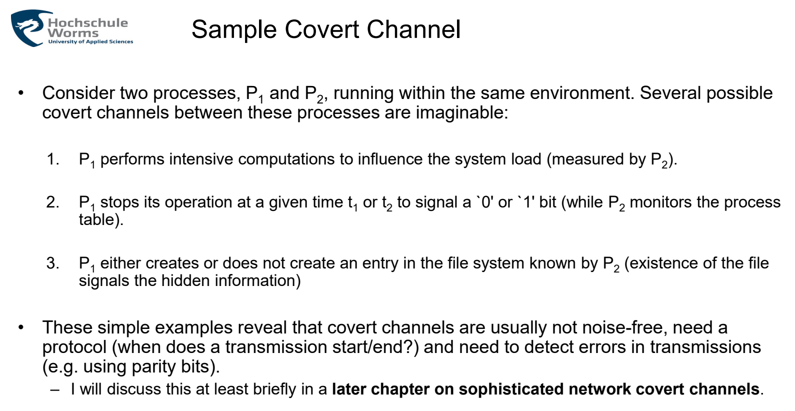
NETWORK INFORMATION HIDING

CH. 2: INTRODUCTION TO LOCAL COVERT CHANNELS

Prof. Dr. Steffen Wendzel
Worms University of Applied Sciences

https://www.wendzel.de (EN) | https://www.hs-worms.de/wendzel/ (DE)
Online Class: https://github.com/cdpxe/Network-Covert-Channels-A-University-level-Course/

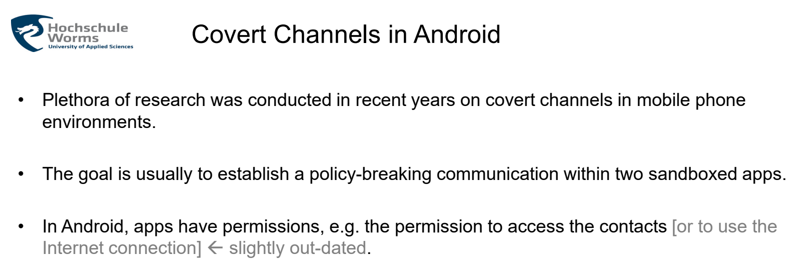
Hello and welcome back to the second part of network information hiding class. This is a brief session. I will discuss local host covert channels that do not exploit network capabilities. So this is a minor topic in this course but I wanted to at least cover it a bit for completion so that you have a full picture so that you have a good overview of this whole information hiding and covert channel topic.



Sample Covert Channel

- Consider two processes, P_1 and P_2, running within the same environment. Several possible covert channels between these processes are imaginable:
  1. P_1 performs intensive computations to influence the system load (measured by P_2).
  2. P_1 stops its operation at a given time t_1 or t_2 to signal a `0' or `1' bit (while P_2 monitors the process table).
  3. P_1 either creates or does not create an entry in the file system known by P_2 (existence of the file signals the hidden information)
- These simple examples reveal that covert channels are usually not noise-free, need a protocol (when does a transmission start/end?) and need to detect errors in transmissions (e.g. using parity bits).
  - I will discuss this at least briefly in a **later chapter on sophisticated network covert channels**.

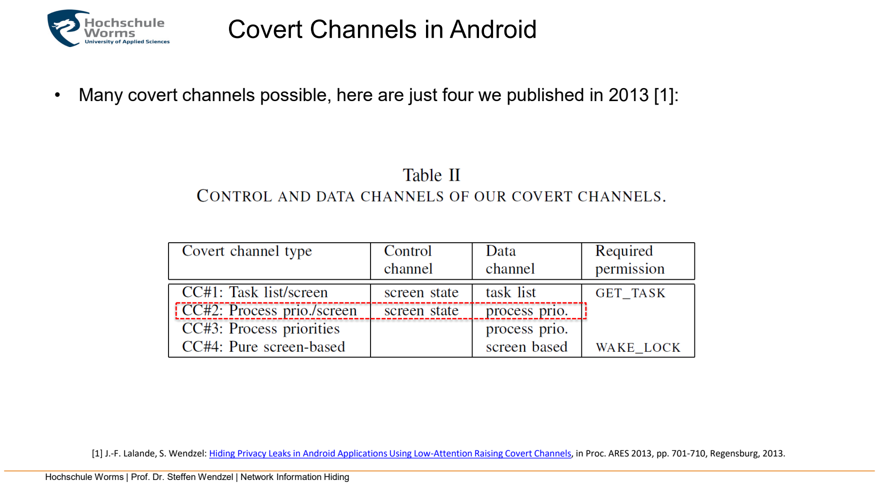
Such covert channels that are locally realized can exploit several capabilities of the system, usually hardware capabilities or operating system related meta-data. If we consider two processes P_1 and P_2, and let's say they run within the same environment. That could be the same operating system, two processes on the same operating system, and several covert channels between these processes are imaginable. For instance, P_1 could perform some intensive computations or not to signal a 1 or a 0 bit to the other process P_2, who has to measure the CPU load constantly. Another example would be if P_1 stops its operation at a given time t_1 or at a given time t_2 to signal a secret 0 or 1 bit, respectively, while again P_2 has to monitor the process table the whole time. And another option would be if P_1 either creates or does not create an entry in the file system so one file that is known by P_1 and P_2. They have to agree in advance on this file name and P_1 constantly checks for existence of this file name and if the file is present then let's say it's a 0 and if it's not present then it's a 1 bit, and P_1 and P_2 performed this operation let's say every second or something like that so they could signal one secret bit per second. And these simple examples reveal that covert channels are usually not noise-free because other processes, for instance, could also create the same file or there could be delay due to scheduling, and in the first example using the CPU load other processes could also influence the CPU load. Also, it's not clear when a transmission starts or ends so it would be good if there would be a protocol that signals such meta information. It would also be good to detect errors, for instance using parity bits. I will discuss network-based solutions for part of these issues in the chapter on sophisticated network covert channels but we will first have a few chapters in between and then I will introduce you to these sophisticated topics.



Covert Channels in Android

- Plethora of research was conducted in recent years on covert channels in mobile phone environments.
- The goal is usually to establish a policy-breaking communication within two sandboxed apps.
- In Android, apps have permissions, e.g. the permission to access the contacts [or to use the Internet connection] ← slightly out-dated.

Another example of such covert channels on local hosts are covert channels for the Android system. So Android was subject of a plethora of covert channel research papers. We also wrote one to cover a fraction of the covert channels that we proposed for Android. The goal usually for Android covert channels is to establish some policy-breaking communication between sandbox apps.



Covert Channels in Android

- Many covert channels possible, here are just four we published in 2013 [1]:

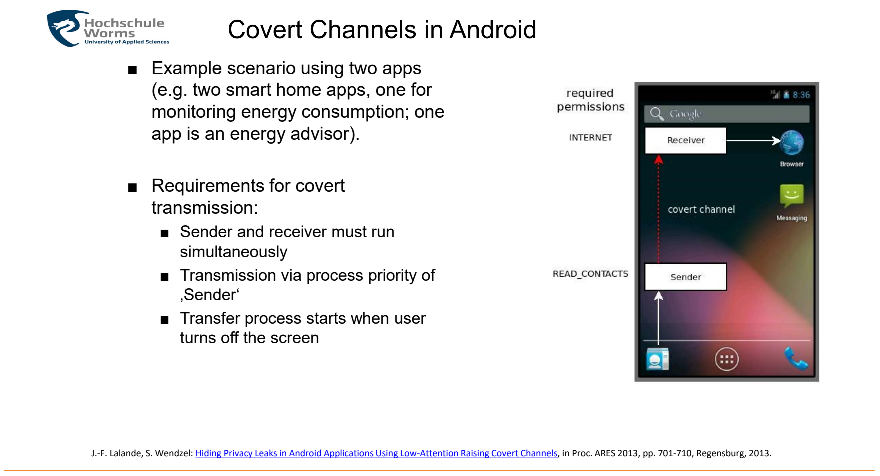
Table II
CONTROL AND DATA CHANNELS OF OUR COVERT CHANNELS.

Covert channel type	Control channel	Data channel	Required permission
CC#1: Task list/screen	screen state	task list	GET_TASK
CC#2: Process prio./screen	screen state	process prio.	
CC#3: Process priorities		process prio.	
CC#4: Pure screen-based		screen based	WAKE_LOCK

[1] J.-F. Lalande, S. Wendzel: Hiding Privacy Leaks in Android Applications Using Low-Attention Raising Covert Channels, in Proc. ARES 2013, pp. 701-710, Regensburg, 2013.

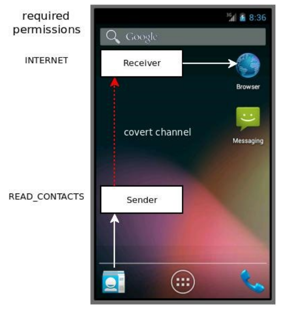
Hochschule Worms | Prof. Dr. Steffen Wendzel | Network Information Hiding

So as you can see here we propose different cover channels for Android and they exploit different characteristics of the OS. These covert channels transfer information from one process to another while the communication wouldn't be allowed. For instance, the screen state, so whether the screen is turned on or off can be used to signal hidden information, also the task list meaning that we can check whether a process is present or not. And, however, this is work from 2013 so I think some updates will have occurred for the Android OS here, also in terms of required permissions, and also process priority. Processes can have different priorities on Android and by using a syscall to decrease the priority another process can notice this and also this way you can signal hidden information. And I will show you covert channel number two that uses process priority in screen state to signal hidden information.



Covert Channels in Android

- Example scenario using two apps (e.g. two smart home apps, one for monitoring energy consumption; one app is an energy advisor).
- Requirements for covert transmission:
  - Sender and receiver must run simultaneously
  - Transmission via process priority of ‚Sender'
  - Transfer process starts when user turns off the screen



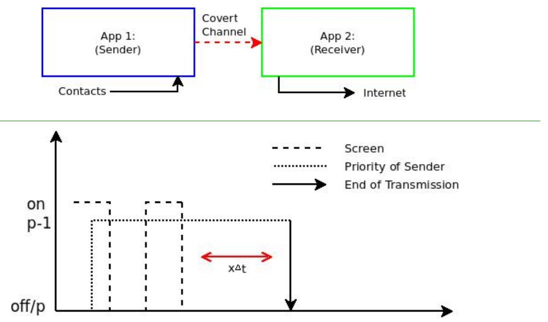
J.-F. Lalande, S. Wendzel: Hiding Privacy Leaks in Android Applications Using Low-Attention Raising Covert Channels, in Proc. ARES 2013, pp. 701-710, Regensburg, 2013.

So the scenario is as follows: this was joint work with Jean-Francois Lalande, and he did most of the work for this paper so credit goes mostly to him, and he also drew these figures. So the example scenario that we have here is that we have two applications, and the user installed both of them, and the first app is one that has the permission to contact remote servers. It can use the internet but it has no access to the contacts. The user did not allow the app to use the contacts, and the second app has the permission to read the contacts but it has no permission to access the internet. I think today all Android applications can access the Internet but I'm not sure. And so the user would assume that his contacts cannot be leaked to the internet because the app that has access to the Internet has no access to the contacts, and the app that has access to the contacts has no access to the Internet. We have shown that there can be a covert channel between the first and second app and this way one can leak the contacts to the other app, and that app leaks the data to the Internet. An example could be that a user downloads to smart home apps or to fitness apps or something, for instance in the case of the smart home apps one could be used for monitoring the energy consumption, and one could be used as an energy adviser or something like that. There are some requirements for a transmission. First of all, the sender and the receiver must run simultaneously. The transmission by a process priority of the sender must be feasible and the transfer process starts when the user turns off the screen.

Hochschule Worms University of Applied Sciences

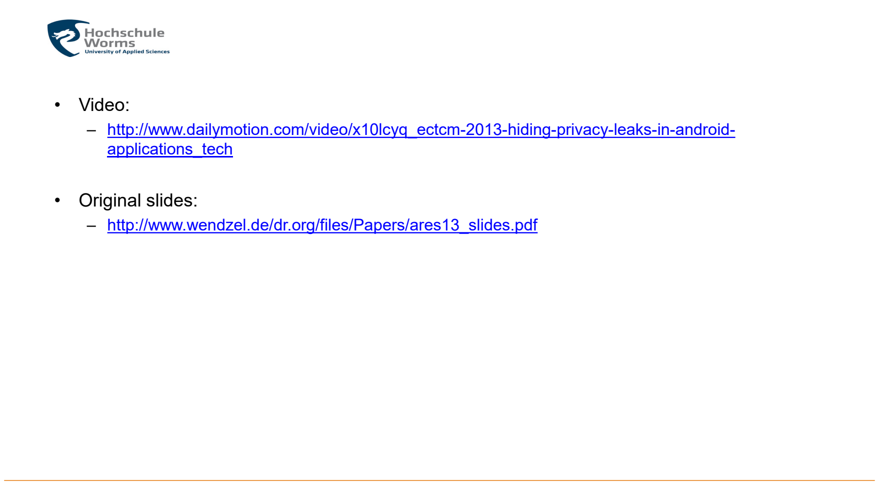
Covert Channels in Android

- How bits are transmitted:



J.-F. Lalande, S. Wendzel: Hiding Privacy Leaks in Android Applications Using Low-Attention Raising Covert Channels, in Proc. ARES 2013, pp. 701-710, Regensburg, 2013.

So how are secret bits transferred? Remember we want to read in the contacts somehow, and we assume that app 1 (the sender app) read the context already, and now we want to transfer these contacts bitwise to the covert channel receiver (app 2) that leaks it to the internet. And we have two channels, a data channel and a control channel for which we use the screen state and the priority of the sender. So the screen state, that's the dashed line here, can be on, the screen can be on or off, and the priority of the sender can be p or $p-1$. Now, let's assume the user turns on the screen and now it turns off the screen. When a screen is turned off by the user it's a signal that we exploit to indicate the start of a hidden transmission but only if the process priority was set to $p-1$. If the process priority is p, which is the standard priority for the process, then we indicate that there's no data available. Now data is available because the sender set the priority to $p-1$, and the screen was turned off so the sending would begin. However, let's assume the user quickly turns on the screen again what leaves not enough time to read the hidden information, to signal the hidden information. The transmission process would then be canceled because the end of the transmission is indicated by the fact that the process priority goes back to p as shown by this arrow here. So the user turns on the screen again and then turns off the screen again or it gets turned on and off automatically because some message was received or a telephone call was received or something like that. And now, the screen goes off again and now the covert receiver measures the time between the screen being turned off and the process priority being set back to p. And this time slot is divided by Δt, and the resulting value represents the hidden symbol. So this is a typical way to indicate a hidden information because you have to somehow have some coding setup, and here we have, for instance, all the Latin letters of the alphabet, and in the simplest case one could say that if Δt is one second then an 'A' would be the transfer of a message for one second, and a 'B' would be two seconds, and 'C' would be three seconds, and so on. And of course you can also perform this bitwise.



- Video:
  - http://www.dailymotion.com/video/x10lcyq_ectcm-2013-hiding-privacy-leaks-in-android-applications_tech
- Original slides:
  - http://www.wendzel.de/dr.org/files/Papers/ares13_slides.pdf

There is a video available that shows the secret transmission if you click on this link, and you can also find the original slides of the presentation during ARES 2013 that I gave in Regensburg. So this is all for this very short local covert channel session, and next time we will discuss fundamental countermeasures that are not specific to the network but that are also working, or especially working for local covert channels and also on the source code level.

Definition

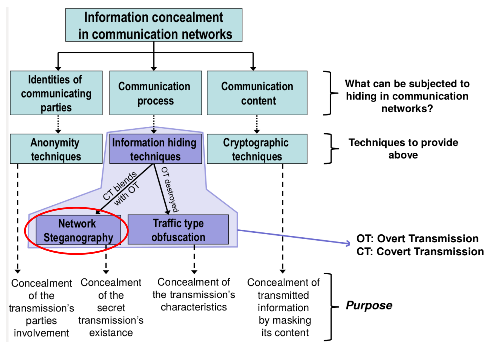
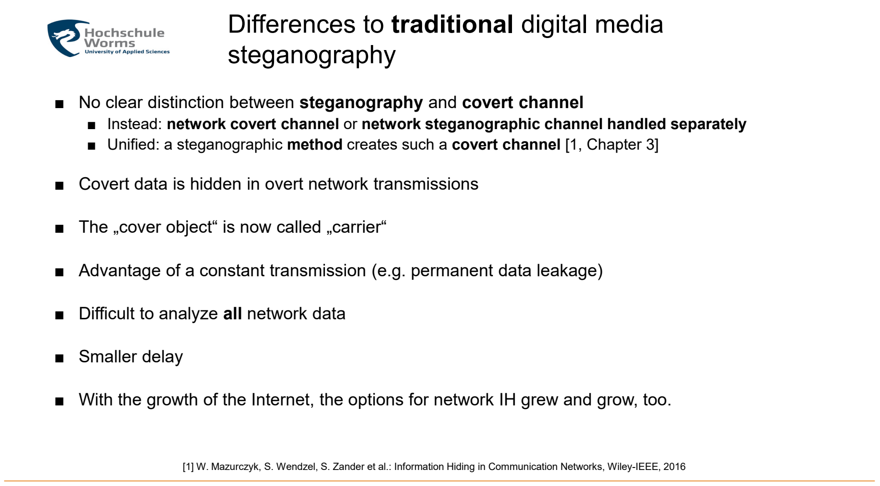

Information concealment in communication networks
Identities of communicating parties
Communication process
Communication content
What can be subjected to hiding in communication networks?
Anonymity techniques
Information hiding techniques
Cryptographic techniques
Techniques to provide above
CT blends with OT
OT destroyed
Network Steganography
Traffic type obfuscation
OT: Overt Transmission
CT: Covert Transmission
Concealment of the transmission's parties involvement
Concealment of the secret transmission's existance
Concealment of the transmission's characteristics
Concealment of transmitted information by masking its content
Purpose

Fig.: W. Mazurczyk, S. Wendzel, S. Zander et al.: Information Hiding in Communication Networks, Wiley-IEEE, 2016

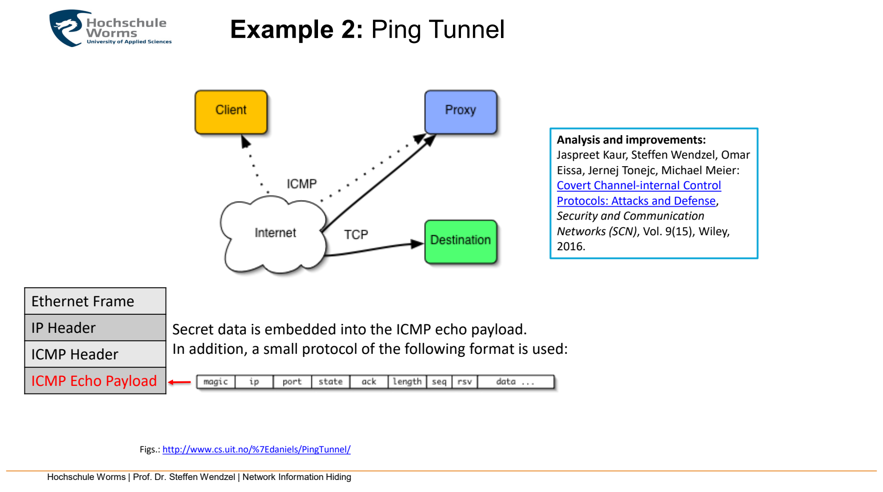
If we look at the following figure from our book, it shows us different concealment methods in communication networks. One can conceal the identity of communicating parties. That would be anonymity. One can also conceal the content. cryptographic techniques can do this. However, we want to hide the communication process. Because we deal with steganographic issues here. This can be done in two different ways: first of all, one can let the covert transmission blend with the overt transmission. That's network steganography. And the other one is that we destroy the overt transmission, so the actual transmission by obfuscating the traffic type. Which leads to concealment of traffics characteristics. However, we will focus only on network steganography, so our goal is to let the secret traffic blend with the overt transmission, so the covert transmission is hidden and the overt transmission and we want to hide the existence of the communication. Nobody should notice that there is such a secret communication going on.


Differences to **traditional** digital media steganography

- No clear distinction between **steganography** and **covert channel**
  - Instead: **network covert channel** or **network steganographic channel handled separately**
  - Unified: a steganographic **method** creates such a **covert channel** [1, Chapter 3]
- Covert data is hidden in overt network transmissions
- The „cover object“ is now called „carrier“
- Advantage of a constant transmission (e.g. permanent data leakage)
- Difficult to analyze **all** network data
- Smaller delay
- With the growth of the Internet, the options for network IH grew and grow, too.

[1] W. Mazurczyk, S. Wendzel, S. Zander et al.: Information Hiding in Communication Networks, Wiley-IEEE, 2016

In the past and sometimes still there is no clear distinction between network steganography and network covert channels. So, network steganography and network covert channel researcher handled separately and back when I was doing my phd this was quite frustrating for me. Because, it led to similar ideas published on a different terminology. So, this is like a wild forest of terminology. So, we cleaned up the terminology, we unified the understanding of both communities, published in the book that I cited many times already and a steganographic method creates such a covert channel. That's the definition that we proposed. So, there is no network steganography without network covert channels. Because, these network steganographic methods create network covert channels. And what we want to have in the end is a networkt covert channel. Covered data is hidden in the overt network transmissions and the cover object in network transmissions is the carrier. So, the network packets, the frames, the datagrams. However, you call it depending on the layer. And the advantage is a constant transmission in comparison to for instance hiding data in one image, transfer the image and then the transmission is done. But, if you want to send more data you need a new image. Here, you just have a continuous flow of data if you want and this allows permanent data leakage, for instance. Which is also resulting in a higher threat in terms of its durability. It's difficult to analyze all network data because network traffic contains so many informations that allow to hide hidden data in it that it's almost impossible to detect and search for all forms of network steganography or network covert channels in such a traffic. Another advantage is the smaller delay. So, you can immediately get a reply and you can immediately send data. If you first have to wait for someone to send back an image then that's taking longer. Of course, one could also send image over network steganography but then in the end you use network as a carrier at least. With the growth of the internet options for network information hiding grew and they still grow a lot.



Example 2: Ping Tunnel

Analysis and improvements:
Jaspreet Kaur, Steffen Wendzel, Omar Eissa, Jernej Tonejc, Michael Meier: Covert Channel-internal Control Protocols: Attacks and Defense, *Security and Communication Networks (SCN)*, Vol. 9(15), Wiley, 2016.

Secret data is embedded into the ICMP echo payload.
In addition, a small protocol of the following format is used:

Figs.: http://www.cs.uit.no/%7Edaniels/PingTunnel/

Hochschule Worms | Prof. Dr. Steffen Wendzel | Network Information Hiding

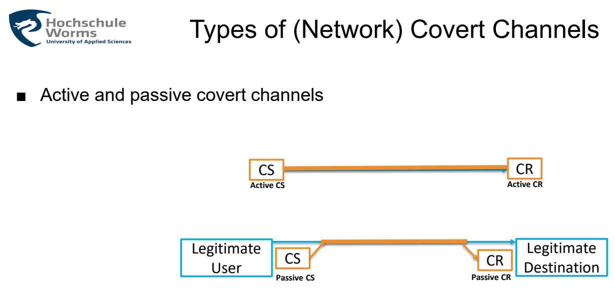
Another example is ping tunnel. Ping tunnel is a little bit more sophisticated. So, the ground setup is as follows: a client wants to communicate with a destination over the internet. But the client is permitted to directly communicate via tcp to the destination, for instance, let's assume that the port is blocked by a firewall. Now, what an attacker can do is setting up some proxy server on some reachable web server or wherever some virtualized host or anywhere in the internet some server and pack the tcp information into icmp, send this to the proxy and the proxy forwards the packets to the destination. When a reply comes, the destination answers the proxy and the proxy packs the reply packet into the icmp packet again and sends it to the client. So, in the end it's as follows: we have our ip packet here with icmp attached and what they send here is an icmp echo reply, I think. And in the echo packets in the echo reply packet you have some payload and in the payload you pack the hidden data, which you can see here plus some meta information like information required for the proxy to see how long is the data packet that is hidden here. Is it an acknowledgement? What's the state? Who's actually communicating? Which IP and port? And so this is a simple version of a covert channel internal control protocol. I will discuss such protocols in detail in the chapter on sophisticated hiding methods. So, this is a good and working example ping tunnel is now even developed further I think there's a github repository and we analyzed this protocol and proposed also improvements in this paper. It's open access so you need no widely access to read it because there are some things that you can improve and still might be things that can be improved, but if you look for a tool that is good for understanding covert channels then it's ping tunnel. Also, there are rules available for typical intrusion detection systems like snort that can help you to detect ping tunnels. So, it's still easy detectable because this magic number is always the same. So, some tools just search for the magic number that indicates the presence of ping tunnel packets and that is used to distinguish between ping tunnel and legitimate icmp echo replies.

Types of (Network) Covert Channels

Fundamental:

- Local and network covert channels
- Storage and timing channels
- Noisy and noise-free covert channels

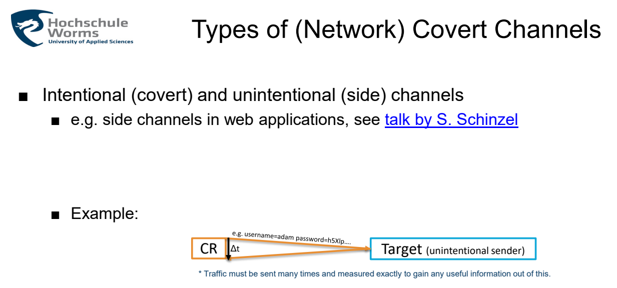
I will now go through some very fundamental categories of covert channels in the network. So, first of all we talk about network covert channels and not about local covert channels that i already introduced a few chapters ago, like those exploiting cpu load between two processes that want to communicate. But, there are also storage and timing channels. Storage channels write your storage location in some network data for instance to the reserved bit in the IP Version 4 header or the icmp payload that we just saw, while timing channels modulate the timing behavior of network traffic to signal hidden information. A simple example is to modulate inter packet gaps, so the timing between succeeding packets for instance you can send packets with a delay of one second or two seconds to indicate a '0' or '1' bit. There are also noisy and noise-free channels. Noisy channels are those that well are not noise-free and noise is a disturbance in that sense and usually randomized at least in many cases. So, for instance, if you use a network timing channel that modulates the timings between succeeding packets and you use the timings of 0.001 and 0.002 seconds or something like that, then other network traffic may might influence the timings. Because hops in between might be busy with processing other packets. So, your own packets might be delayed and then you have noise. It's always nice for the covert channel user to have a noise free channel but in many cases that's just not possible. So, these are the very fundamental categorizations. Now, I will go into some network-specific more sophisticated differentiation.


Types of (Network) Covert Channels

- Active and passive covert channels

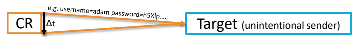
Hochschule Worms University of Applied Sciences
CS
Active CS
CR
Active CR
Legitimate User
CS
Passive CS
CR
Passive CR
Legitimate Destination

So, there are first of all actually it's not network specific in any case. So, active and passive covert channels for instance are not network specific. But, here they play a big role. So, an active covert channel is one where someone actively sends traffic. So, the covert channel sender which I abbreviated with cs and the covert channel receiver which I abbreviated with cr. So, the cover channel sender generates its own traffic to hide information in this traffic. The overt traffic should appear as legitimate traffic, but the truth is there's some hidden traffic inside it. And so the passive way would be to piggyback traffic generated by a legitimate user basically as a man in the middle that can be on the host of the legitimate user or in between. So, it can be integrated in the TCP/IP stack of the sender like done by Joanna Rodkovska during some CCC talk several years ago. But, it can also be done on a hop in between and so the traffic passes by and the covert channel sender modifies the traffic, but does not generate own traffic. And the covert receiver would remove read the message also here from the traffic however the receiver is less important for the active and passive part as long as the receiver can read the traffic it does not matter so much. So, that the clear that the main differentiation is between the sender the active and passive sender. There are also reversibility methods, where the passive receiver would restore the original traffic and that's a capability that is specific to passive covert channels. So, the covert receiver would try to restore the original traffic and then forward it to the legitimate destination. So, that the legitimate destination does not see the original the covert channel content any more for instance if the covert sender would modify the reserved IP bit IP reserved bit then whatever message the cover channel receiver receives, it could always clear the reserved bit. So, that the legitimate destination always gets a zero bit.



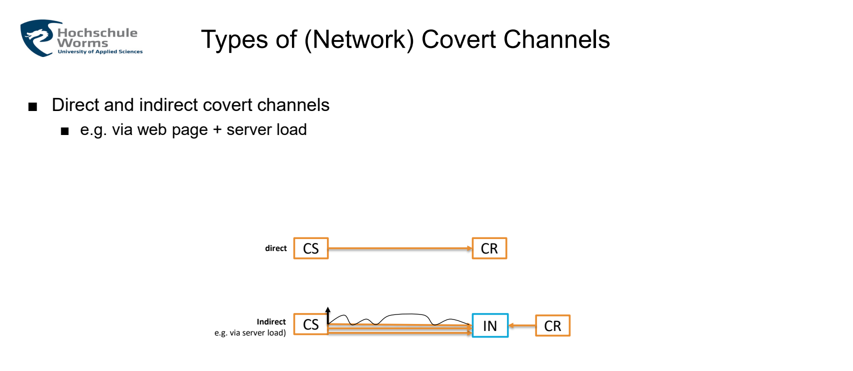
Types of (Network) Covert Channels

- Intentional (covert) and unintentional (side) channels
  - e.g. side channels in web applications, see talk by S. Schinzel
  - Example:


e.g. username=adam password=h5Xlp....
CR
Δt
Target (unintentional sender)

* Traffic must be sent many times and measured exactly to gain any useful information out of this.

The intentional versus unintentional or side channel is ... So, the side channels are a sub type of covert channels. They are covert channels without that does that send without having the intention to send information. There's interesting talk by Sebastian Schinzel from a few years ago and I linked his talk here. I just want to show some central idea here. So, instead of having a covert sender, we have a side channel sender here or a target an unintentional sender and a covert receiver that measures something, some timing information, for instance. And so it could work as follows: let's say the target reacts differently to depending on which information the covert receiver sends to the packet to receive the actual information. If the covert receiver is a web client and this is a database or a wiki, then the covert receiver could send lots of login credentials many many times and measure how long it takes for the reply to arrive. And when the reply arrives the time delta is computed and the thing here is let's say there's a sql database at the target and let's assume there's a credential sent by the covert receivert that contains an invalid username. So, the target would say okay username does not exist and some error message would be returned. But if the username exists and also a password would be sent in the same packet, then also the password would need to be checked and maybe some additional permission. So, the target system needs to perform more reactions or in other words needs to invest more cpu time and then the reply will take slightly longer. So, delta t will be higher and the covert receiver can try to measure this. So, this is an unintentional information sending process going on here and that can be exploited to data mine for instance which usernames exist and which do not exist.



Types of (Network) Covert Channels

- Direct and indirect covert channels
  - e.g. via web page + server load

Another type of covert channel differentiation is between direct and indirect covert channels. In a direct way the sender well directly sends to the receiver, but in the indirect way some intermediate node is exploited to signal hidden information. For instance, let us assume this is a webserver the IN is a web server and the covert sender covert sender sends lots of traffic to the web server or none. So, over time it send lots or not so much more little and then a lot of traffic for some longer time and let's say that this is a signal that can be understood as '1' '0' '1' '0' '1' '1' '1' '0' '1' or something like that. The idea here is to influence the load on the web server and while the covert receiver tries to probe and tries to also get requests fulfilled from the webserver the response time of the web server will take longer, if the covert receiver requests a website during a high load from the covert sender and it will take less time, if the covert receiver requests a website, while there is little or no load from the covert sender. So, this way we have again a side channel that is exploited here, but intentionally and so it's not as intentionally influenced by the sender. I thought this is a unintentional sending process here from the web server and it's an indirect '1', because the intermediate node is like a proxy or a cache for the information. I will in a few chapters deepen the knowledge of you in the sense that I will speak about such network caches. We published work also on network protocol exploitation to store data in network protocol caches and I will also speak about reversible data hiding methods at least about those that are already published and about many many more details of sophisticated hiding methods, but first we need to understand more about how data can be hidden. So, which hiding methods do actually exist these are more models and general difference differentiations between types of covert channels but in the next chapter we will learn what hiding methods do actually exist, because so far we saw very little only the reserved bit exploitation and the icmp payload exploitation but these are very simple methods and there are many more.