



## Covert Channels: Connectivity

### *Circumventing network borders*

Jürgen Pabel, CISSP

Covert channels are a serious and distinct threat. However, they differ in significance and impact, depending on the environment and the role of the covert channel. Although covert channels often integrate both elements of circumvention and secrecy, one can distinguish between covert channels primarily used for circumvention of controls on the one hand and those used for secret communications on the other. Smuggling of electronic documents over covert channels is a significant threat, but this essay evaluates the role of covert channels in circumventing network controls to establish unauthorized network connectivity. Look to the second installment, "Covert Channels: Data Theft", as to how else covert channels endanger your environment.

There are essentially only three types of data flows in networks: incoming, outgoing and local. At the same time, a distinction must be made between the flow of communication and the actual traffic it incurs. VPN connections are usually the only connections that corporate networks allow inbound, but the traffic associated with those connections flows in both directions. Someone accessing the World Wide Web from within the internal network, also requires both incoming and outgoing traffic - yet, the direction of the communication is considered outbound. But why do I care so much about this distinction? Here is why: the paradigms of incoming and outgoing connections differ radically. Inbound connections provide usually network level connectivity, essentially the integration of an external entity onto the internal network, while outbound connections only provide access to specific resources outside the internal network. The security implications are dramatic: since incoming VPN connections are authenticated, the encapsulated traffic is conceptually deemed acceptable, although there may be additional controls in place to control the flow and to verify its content. Outgoing connections are just the opposite, they are in most corporate networks implemented through an intermediary process; from a security standpoint, it is generally seen as undesirable to provide direct access to external networks - in most cases the Internet. We now have a functional idiosyncrasy: incoming traffic is evidently unrestricted - within the perimeters of VPN security - while outgoing connectivity is limited. A covert channel within an incoming VPN connection is therefore of no apparent value, but a covert channel which facilitates outbound connectivity may be sapid to some.

It should be obvious that covert channels must be prevented. However, the nature of



covert channels is that anything can facilitate one. However, there are practical limits on the type of covert channels that are of interest in this context: to provide practical access to otherwise restricted resources, covert channels must be transport efficient. It would be of no practical use to an attacker if the throughput or latency of the covert channel exhibits values well beyond any practical limits. An attacker is unlikely to choose the Email system as the transport mechanism – the inherent delay would not allow a responsive transport channel.

Now we can finally get to the technical aspects of current covert channel technologies, found in increasing numbers in corporate networks. I am sure by now you are eager to ask (remember, these questions are in the context of network security):

- What is possible?
- How is it possible?
- What countermeasures are available?

The first question is the easiest to answer: anything is possible. Ready-to-deploy tools exist that allow virtually everyone on the internal network to establish a full blown IPv4 communication layer to the outside, around all the neat technologies meant to secure your networks: firewalls, proxy systems, content filters, antivirus systems and network monitors. But how is that possible? Well, one reason may be the architecture of the IT environment: if your internal systems rely on DNS servers that allow querying of records external to your network, then a specific tool and privileged access to a single system on your internal network is all that is needed to open the gates to everything the Internet has to offer - inbound access included. The tool requires privileged access because it installs a virtual network interface on the system that transfers all its traffic via DNS requests through the DNS servers your organization so graciously provide. Fortunately, many corporate networks are in fact set up in a different manner, such that internal DNS servers have in fact no way of delegating requests to the Internet. There is, however, one specific type of network that almost always exhibits this configuration: commercial wireless hotspots for Internet access. What are the implications? Free wireless access to those who have no scruples in using said software; alleviating this specific problem is fairly simple on a technical level, but imagine the deployment nightmare. Wireless hotspot providers usually operate thousands of hotspots in geographically distributed locations; the organizational process needed to reconfigure all hotspots would be very involved.

Does your internal network allow web access? If so, this item will interest you: there are numerous tools available that provide arbitrary transport layer connectivity using TCP to the Internet. Unlike the previous scenario, no privileged system access is



required - only access to the World Wide Web. One specific category of tools relies on a peculiarity of the protocol known as HTTP, in combination with communication security, its technical name being SSL/TLS. The problem arises when a client must connect through a proxy server to connect to the Internet: the proxy usually retrieves the requested pages on behalf of the client, but with SSL protected connections (HTTPS), it can't retrieve the pages for the client because the client demands to communicate with the server in private. How was HTTP adapted to resolve this conflict? It was extended in a manner to allow a client to advise the proxy that it needs to communicate with a specific server over a cryptographically secured channel. The proxy reverts from its active role of retrieving pages to a passive entity that does little more than forward all data between the client and the server. Presume now that the client is not a Web browser, but rather a SSH client connecting to a SSH server - et voila, an outbound path is created. But: it gets worse; SSH has a function which allows the establishment of an inbound TCP tunnel. Now it becomes possible to connect to your internal network from the Internet. Again, without your security gadgets ever noticing anything. Resolving this threat is not easy; it's not an implementation issue, it's a protocol design issue. I would not hold my breath to wait for someone to "fix" the HTTPS interface specifications for proxies, too many implementations and installations exist. There are essentially two options: disable or restrict HTTPS access through the proxy. If this isn't possible, there is another solution, but it is a very complex system with implications on user privacy, these products are called SSL-Proxies.

Now, let's presume you have resolved the DNS problem and the HTTPS issue is also addressed. You think you have done everything you can to make it as hard as possible for anyone to create a covert channel? -Yes, I can assure you, you have done all you can. Bad news, though: a problem exists for which there is no practical solution. HTTP was designed to facilitate high throughput and low latency - perfect conditions for a covert communication channel. Ready to deploy software is again widely available to establish outbound and inbound connections, and there is essentially no technical defense against it. When you provide web access to employees, you are not only trusting their discipline regarding surfing habits - you are entrusting the security of your network in them.

I would like to mention one last aspect: the presumably "good" guys don't refrain from abusing HTTP neither. Is your organization deploying SOAP? SOAP was specifically designed on top of HTTP, because the organization that designed SOAP realized that HTTP access would not be easily controllable. Think about that the next time you have to deal with issues about controlling the flow of data on your networks.