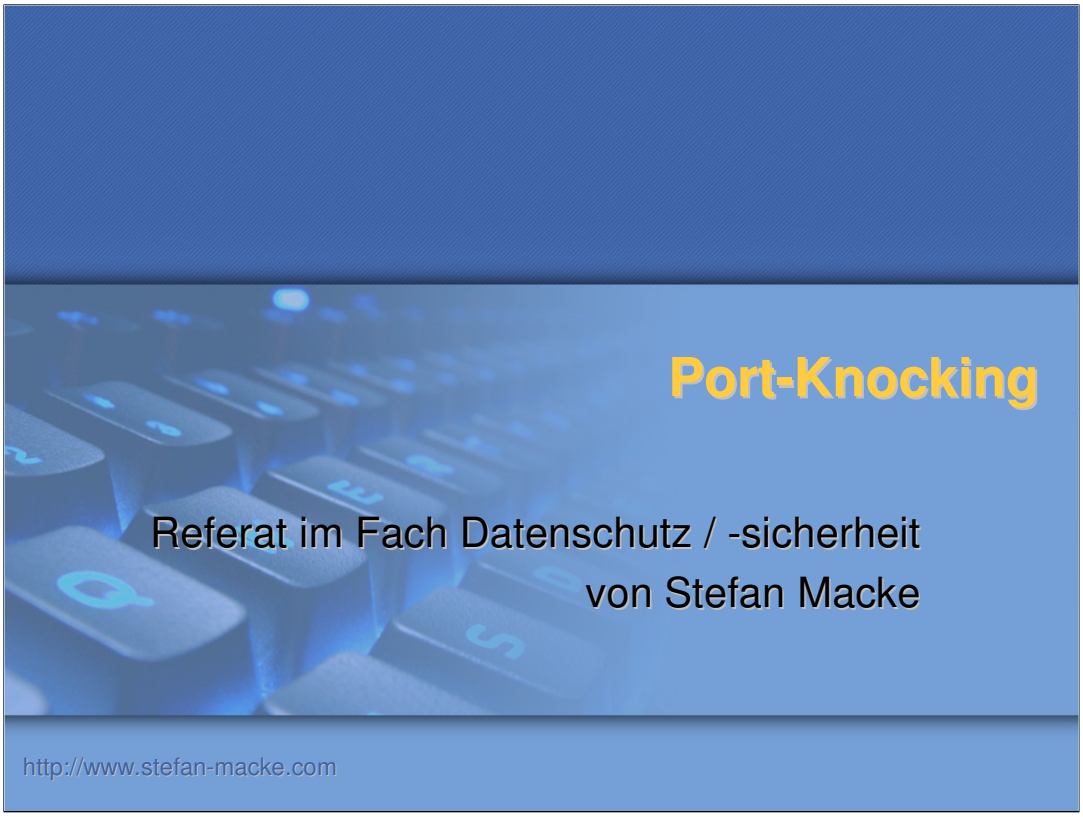
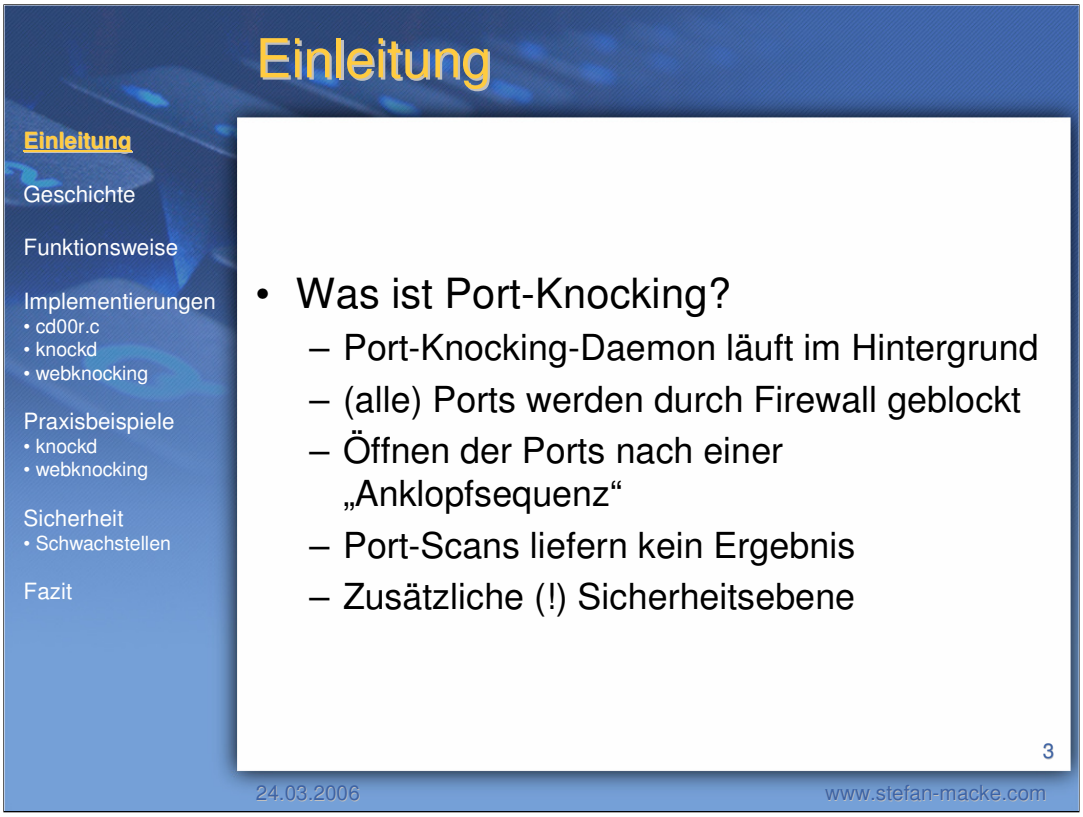

Port-Knocking
Referat im Fach Datenschutz / -sicherheit
von Stefan Macke
http://www.stefan-macke.com

Agenda

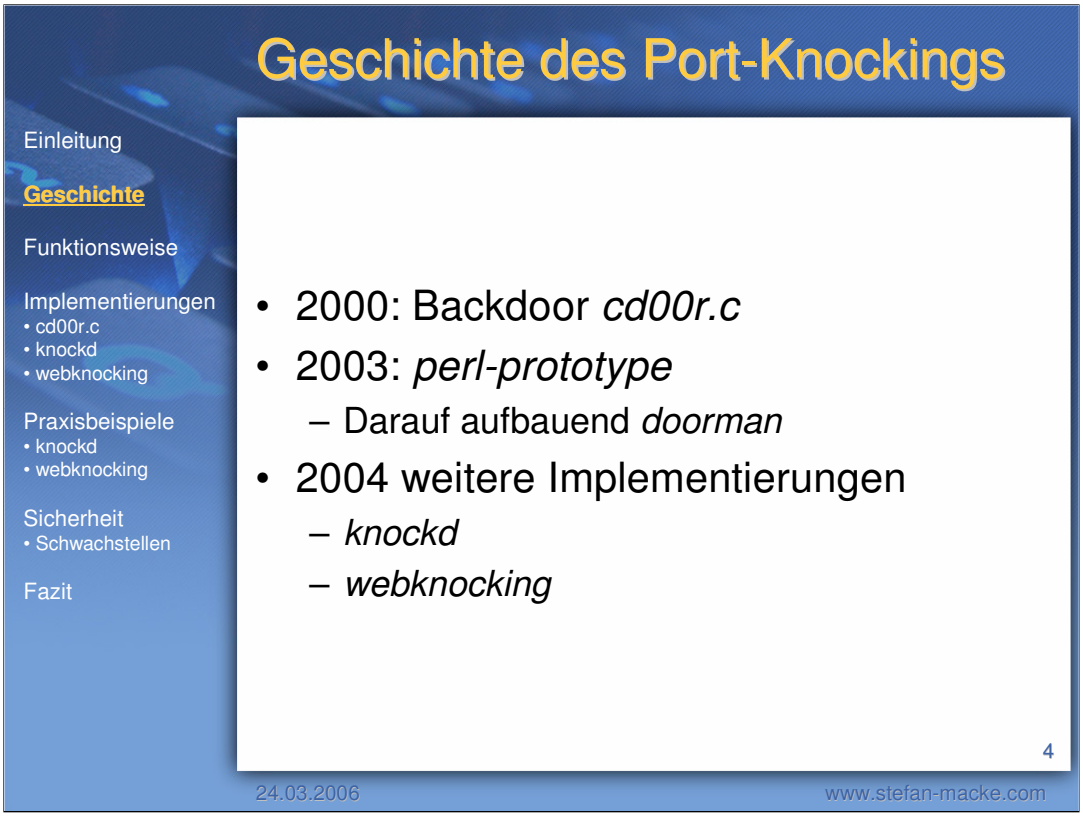
- Einleitung
- Geschichte des Port-Knockings
- Funktionsweise
- Implementierungen
- Praxisbeispiele
  - *knockd*
  - *webknocking*
- Sicherheit durch Port-Knocking?
- Fazit



Port-Knocking ist eine Methode zur Absicherung eines Servers. Hierbei werden alle oder nur die „interessanten“ Ports (wie SSH, FTP etc.) grundsätzlich von der Firewall geblockt, sodass Port-Scans auf den Server keine Auskunft über die zu schützenden Dienste geben. Die Ports werden durch die Port-Knocking-Software in der Firewall freigeschaltet, sobald ein Client eine bestimmte „Anklopfsequenz“ gegen den Server abgesetzt hat. Diese Sequenz besteht aus Verbindungsversuchen zu bestimmten geschlossenen Ports. Diese (geblockten) Verbindungsversuche werden von der Port-Knocking-Software analysiert, und im Falle der korrekten Sequenz werden die gewünschten Ports freigeschaltet.

Wichtig ist hierbei, dass die Port-Knocking-Software nicht als alleinige Sicherheitsmaßnahme gelten kann, sondern nur die Ports und Programme absichert, die häufig Angriffen ausgesetzt sind (z.B. SSH).

[Quelle: http://en.wikipedia.org/wiki/Port_knocking]

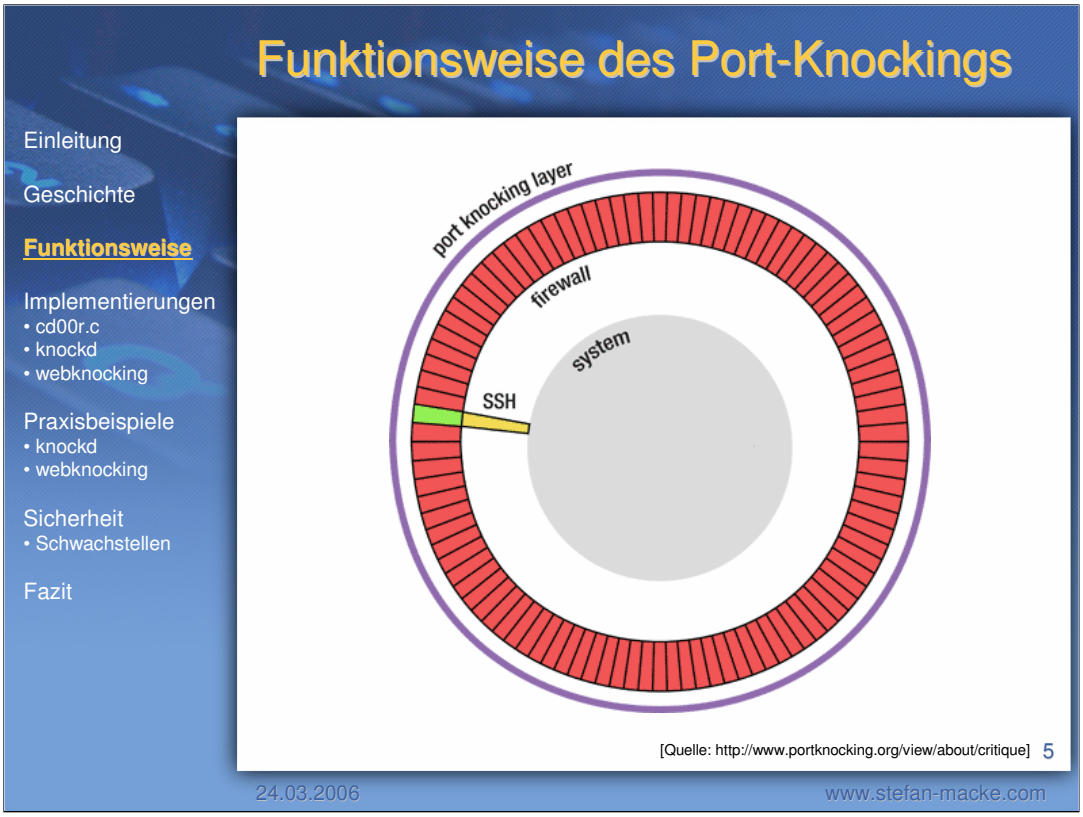


•Im Jahr 2000 wurde die Backdoor *cd00r.c* entwickelt, deren Aufgabe es war, „unsichtbar“ auf Verbindungsversuche zu bestimmten geschlossenen Ports zu lauschen und bei Einhalten einer bestimmten Reihenfolge eine Remote-Shell zu öffnen. *cd00r.c* lauschte mit Hilfe der *pcap*-Bibliothek am Netzwerktraffic, nutzte aber nicht den Promiscous Mode, wodurch ein Auffinden des Tools erschwert werden sollte.
[Quelle: http://www.phenoelit.de/stuff/cd00r.c]

•2003 entwickelte Martin Krzywinski ein kleines Script in Perl, dessen Funktion es war, in den Logs der Firewall eines Servers nach geblockten Verbindungsversuchen Ausschau zu halten und im Falle einer bestimmten Reihenfolge, einen Port in der Firewall freizuschalten. Dieses Verfahren zur „Manipulation von Firewalls mit Hilfe geschlossener Ports“ nannte er „Port-Knocking“.
[Quelle: http://www.linuxjournal.com/article/6811]

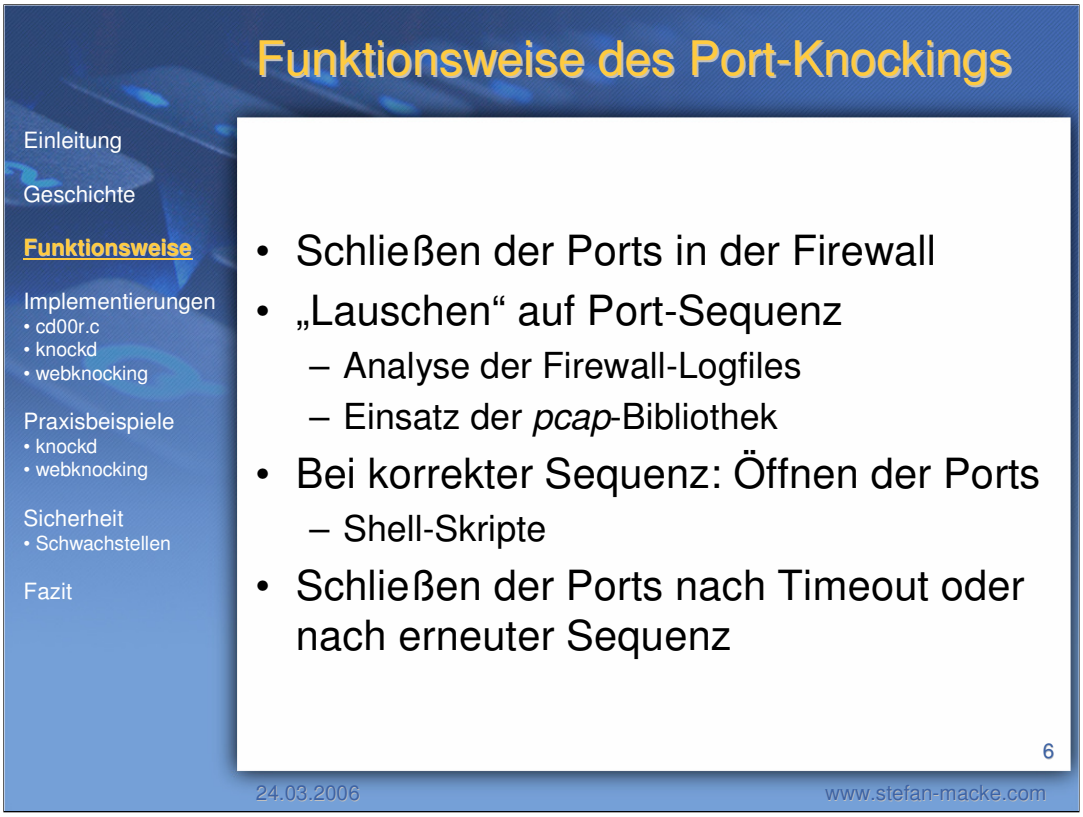
> •Auf diesem Projekt aufbauend wurde das Programm *doorman* entwickelt, dass nicht auf eine Port-Sequenz, sondern nur auf einen einzigen Port lauscht und einen darüber empfangenen verschlüsselten String als Passwort entgegennimmt.
> [Quelle: http://doorman.sourceforge.net/]

•2004 folgten dann weitere Implementierungen, unter anderem *knockd* und *webknocking*. Ersteres liest nicht die Firewall-Logs sondern arbeitet auf Basis von *pcap*, und letzteres ist eine Implementierung auf Web-Ebene, die die Log-Files des Webservers nach fehlerhaften Seitenaufrufen durchsucht.
[Quellen: http://www.zeroflux.org/cgi-bin/cvstrac/knock/wiki und http://webknocking.de/semaphor/semaphor.php?item=webknocking]



Das Modell des Systems, auf dem Port-Knocking eingesetzt wird gleicht einer Zwiebel, die mehrere Schichten hat. Die Port-Knocking-Schicht liegt vor der Firewall. Sobald die erwartete Knock-Sequenz erkannt wird, wird die Firewall zum Freischalten des gewünschten Ports veranlasst und der Benutzer kann sich normal verbinden. Dadurch wird ein Angriff auf die well-known Dienste SSH, FTP usw. verhindert, da die Ports von außen als geschlossen angezeigt werden.

[Quelle: http://www.portknocking.org/view/about/critique]



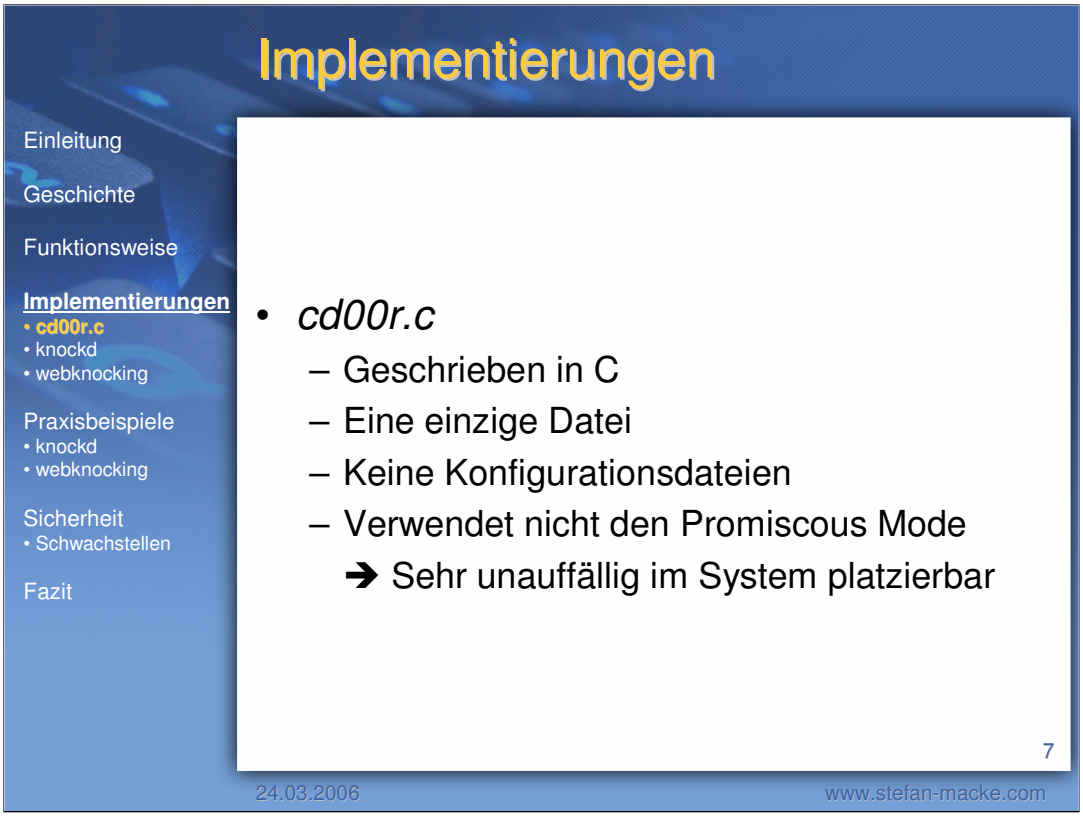
•Zunächst einmal müssen alle (oder nur die gewünschten) Ports in der Firewall geblockt werden, damit sie von außen nicht erkannt werden können.

•Um die gewünschte Port-Sequenz zu ermitteln, gibt es für die Port-Knocking-Software zwei Wege:

- •Die Analyse der Logfiles der Firewall. Hier müssen alle Verbindungsversuche zu geschlossenen Ports geloggt werden. Dies kann sehr schnell zu großen Datenmengen führen.
- •Einsatz der *pcap*-Bibliothek, um den Netzwerkverkehr zu analysieren. Hierbei werden auf dem link-layer alle Verbindungsversuche analysiert bevor die Firewall sie blockt.

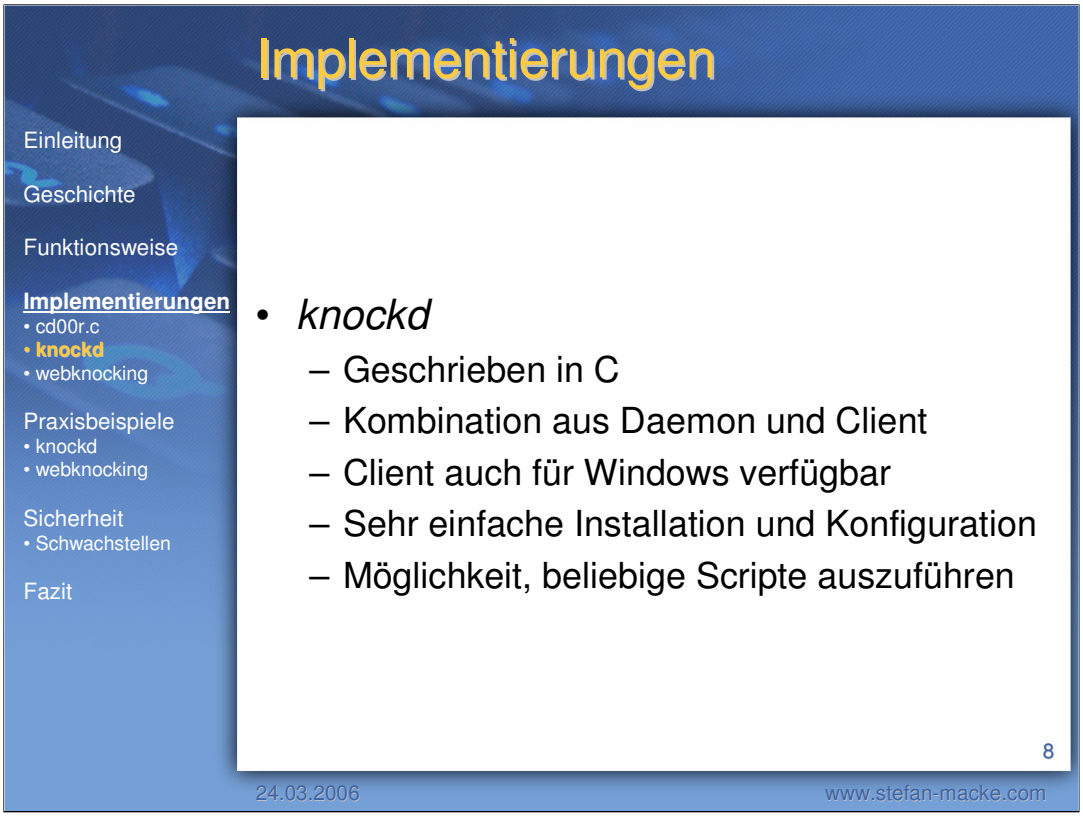
•Erkennt die Port-Knocking-Software die korrekte Knocking-Sequenz, werden z.B. durch Shell-Skripte die gewünschten Ports in der Firewall geöffnet und der Anwender kann sich normal verbinden.

•Nach einem festgelegten Timeout oder einem erneuten Port-Knocking werden die freigeschalteten Ports wieder geschlossen.



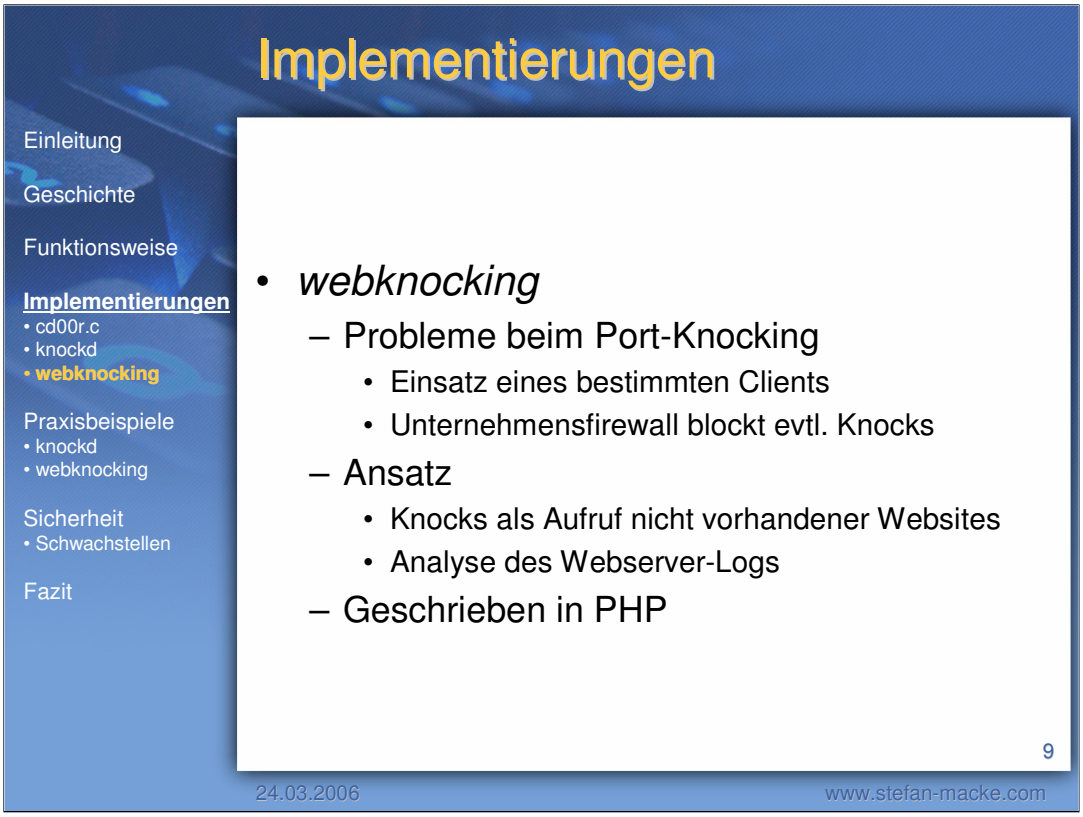
Die Backdoor *cd00r.c* ist darauf ausgelegt, möglichst unauffällig in einem System platziert werden zu können. Sie braucht wenig CPU-Zeit und besteht aus einer einzigen Datei, die Konfiguration wird direkt im Quelltext vorgenommen. Zusätzlich verwendet sie nicht den Promiscous Mode, um im Systemlog keine Spuren zu hinterlassen und somit nur äußerst schwierig nachgewiesen werden kann.

[Quelle: http://www.phenoelit.de/stuff/cd00r.c]



Die Port-Knocking-Software *knockd* umfasst einen Daemon und den zugehörigen Client. Sie ist sehr einfach zu installieren und bietet auch einen Client für Windows. Die Knock-Sequenzen können aber auch mit einem beliebigen anderen Programm (*telnet*, *nmap* etc.) erzeugt werden. Der Daemon kann auf UDP. und TCP-Verbindungen lauschen und auch nur bestimmte Pakete (SYN, FIN etc.) annehmen. Er ist über eine Konfigurationsdatei sehr einfach zu konfigurieren und kann mehrere verschiedene Sequenzen verarbeiten. Dabei ist es möglich beliebige Shell-Scripts ausführen zu lassen.

[Quelle: http://www.zeroflux.org/cgi-bin/cvstrac/knock/wiki]



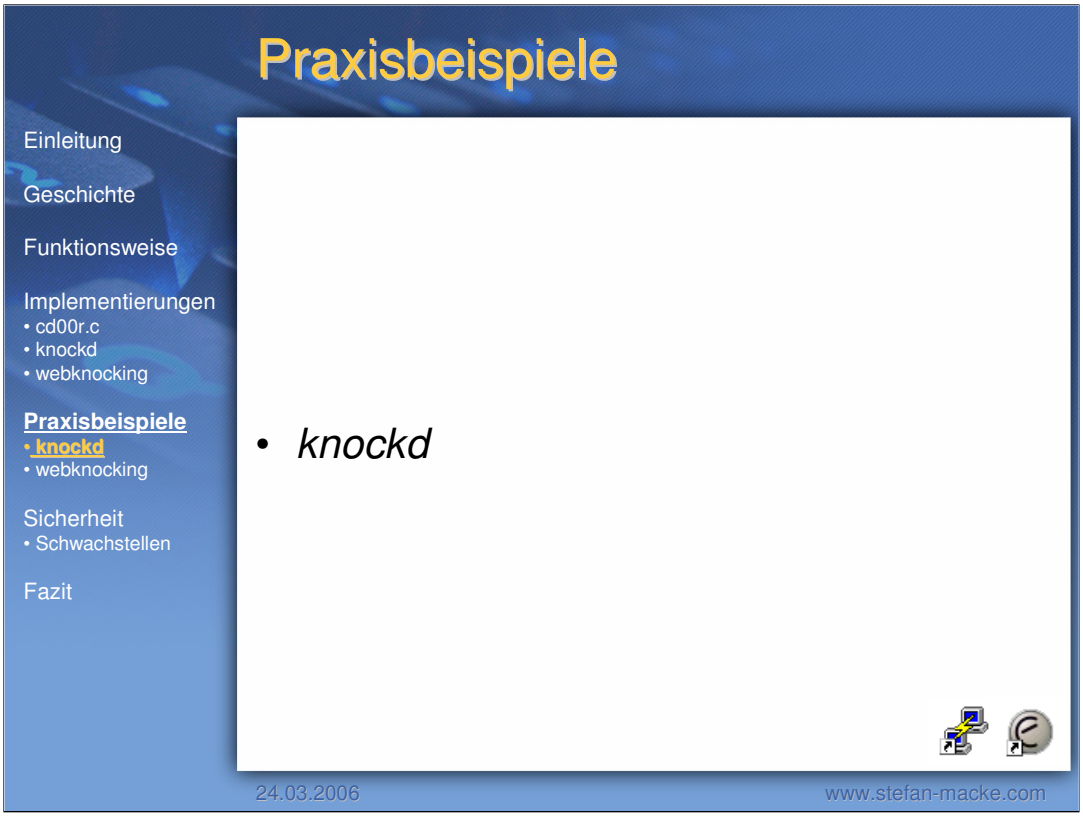
•Der Ansatz des Port-Knockings birgt zwei Probleme:

- •Der Benutzer muss evtl. einen eigenen Client verwenden, der nicht überall verfügbar ist. (Lösung: *telnet* etc.)
- •Eine Firewall blockt evtl. die Knocks, wenn z.B. nur Webverbindungen nach draußen erlaubt sind.

•Ansatz *webknocking*

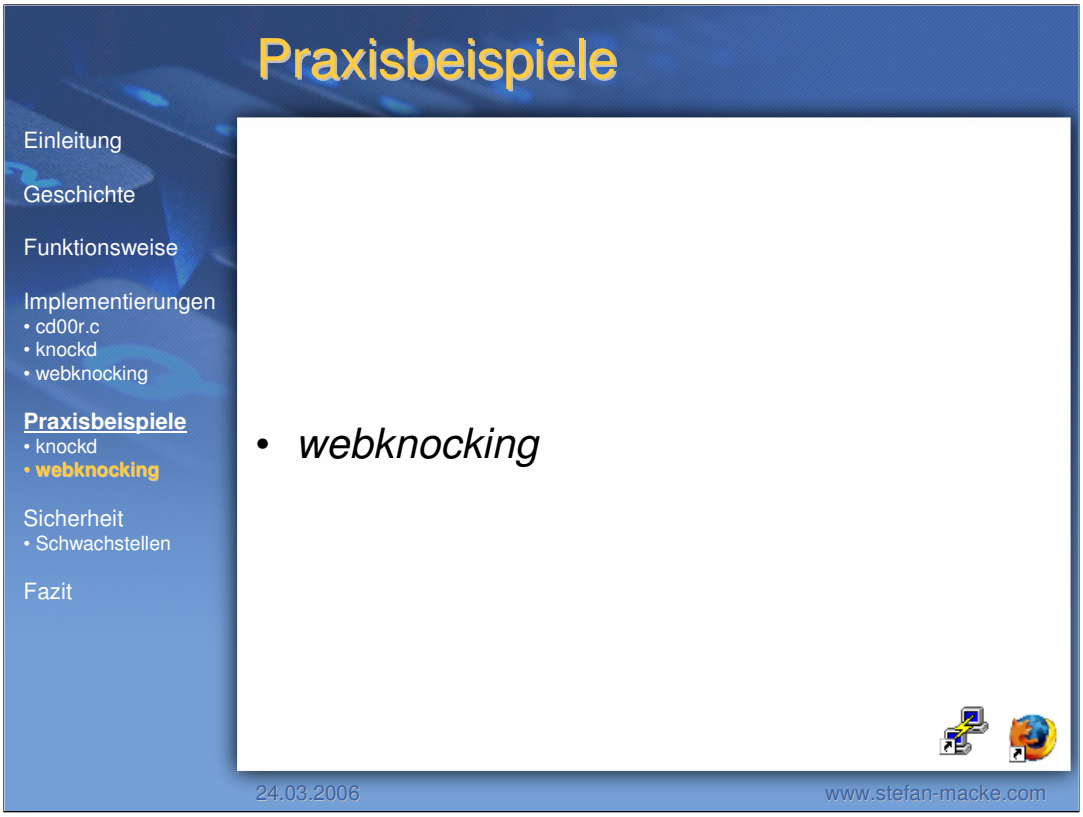
- •Analyse eines Webserver-Logs und nicht des Firewall-Logs.
- •Knock-Sequenz wird über das Aufrufen nicht vorhandener Webseiten und den abschließenden Aufruf der Knocking-Seite realisiert.

[Quelle: http://webknocking.de/semaphor/semaphor.php?item=webknocking]



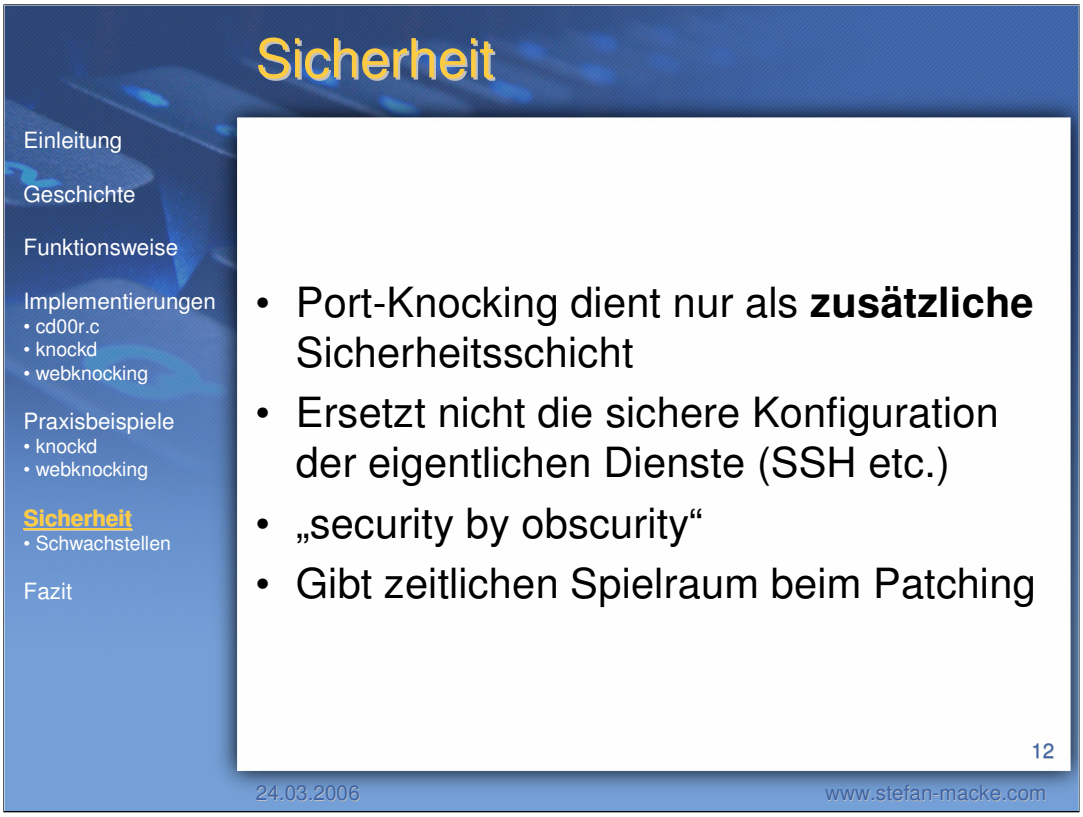
Im Praxisbeispiel wird die Installation und Konfiguration von *knockd* gezeigt. Des Weiteren wird eine Sequenz abgesetzt und beobachtet, wie die Software reagiert.

- *apt-get install knockd*
- Konfigurationsdatei */etc/knockd.conf* erstellen bzw. ändern. Eigene Sequenz erstellen und eigenes Script einbinden.
- Starten des Daemons und *knock*/*telnet*/*nmap* von anderem Rechner.
- Analyse des Netzwerkverkehrs mit *ethereal*.



Das zweite Praxisbeispiel zeigt die Implementierung von *webknocking* auf dem *Apache2*-Server und das Absetzen einer Knock-Sequenz.

- Installation und Konfiguration von *webknocking*
- Aufruf der Seiten und Ausführen des eigenen Scripts
- Analyse des *Apache2*-Logs mittels *tail -f*

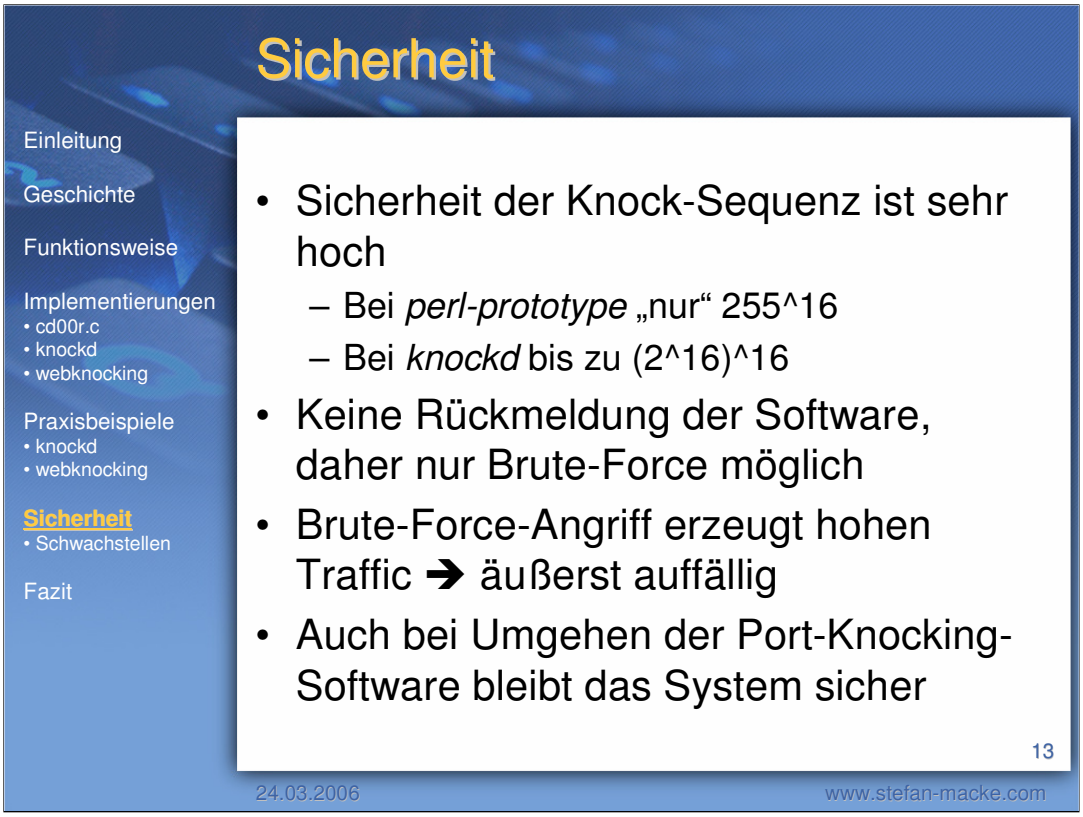


•Zur Sicherheit durch Port-Knocking ist zunächst sagen, dass es grundsätzlich nicht als alleinige Sicherheitsschicht dienen kann. Es sollte nur eingesetzt werden, um die Angriffe auf SSH, FTP usw. zu verhindern bzw. zu erschweren.

•Daher müssen die eigentlichen Dienste weiterhin sicher konfiguriert werden.

•Port-Knocking bietet „security by obscurity“, d.h. ein System, das Port-Knocking einsetzt erscheint von außen als offline bzw. die häufig angegriffenen Ports sind geblockt. Somit wird ein Angriff verhindert.

•Ein großer Vorteil des Port-Knockings ist die zusätzliche Zeit, die es einem Administrator verschafft, um evtl. Sicherheitslücken in den Serverdiensten zu patchen. Da z.B. SSH nicht öffentlich zugänglich ist, ohne die Knock-Sequenz zu kennen, wird die Chance verringert, das Exploits eingesetzt werden um den Server zu kompromittieren.

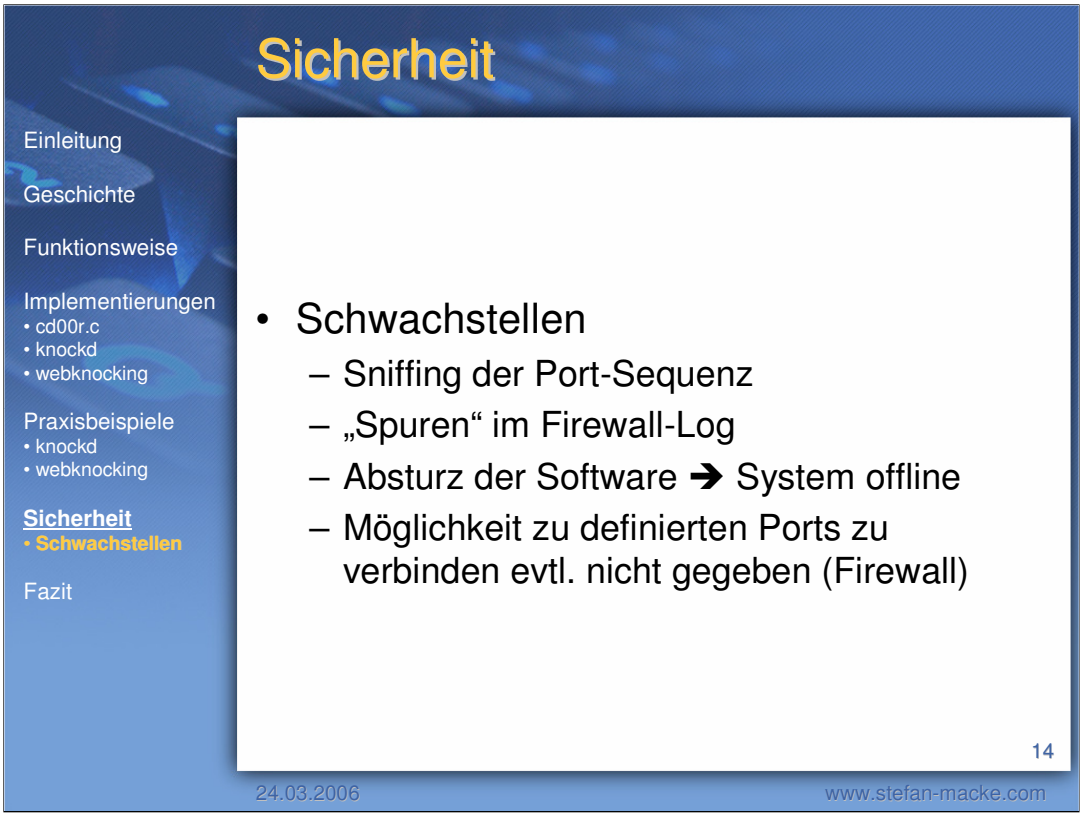


•Die Sicherheit der eigentlichen Port-Sequenz ist sehr hoch. Die ursprüngliche Implementierung *perl-prototype* konnte nur 255 Ports überwachen, wobei max. 16 hintereinander als Sequenz erlaubt waren. Somit ergibt sich eine Komplexität von 255^16 für diese Sequenz.

•Bei *knockd* und anderen Implementierungen können beliebige Ports verwendet werden, sodass die Komplexität der Sequenz auf (2^16)^16 steigt. Ein Brute-Force-Angriff ist somit praktisch ausgeschlossen.

•Brute-Force ist aber gleichzeitig die einzige Möglichkeit die Sequenz herauszufinden, da die Software keinerlei Hinweise gibt, nachdem eine Sequenz getestet wurde. Sobald eine falsche Reihenfolge erkannt wird, wird die komplette Sequenz zurückgesetzt. Der Angreifer bekommt hiervon jedoch nichts mit, da weder die Firewall noch die Software selbst irgendwelche Ausgaben erstellt.

•Zuletzt bleibt das System sicher, auch wenn die Software umgangen werden kann, da die eigentliche Absicherung immer noch auf der Ebene der geschützten Dienste selbst liegt (SSH etc.)

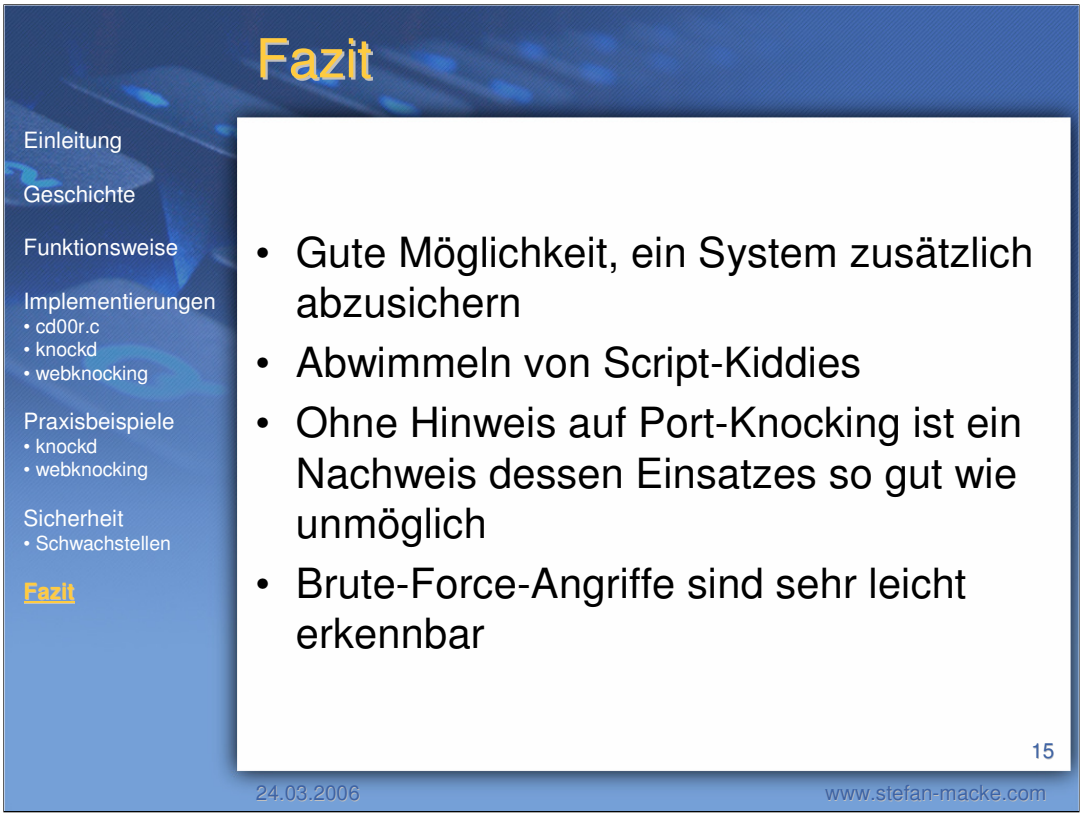


•Auch die Port-Knocking-Software hat Schwachstellen, über die die Knock-Sequenz herausgefunden werden kann.

- •Zunächst einmal kann der Netzwerk-Traffic gesnifft werden, und auf eigenartige Verbindungsversuche (etwa zu hohen Ports) untersucht werden. Dies setzt jedoch den Zugang zum Netz des Knock-Benutzer voraus. Dem entgegenwirken kann man bspw. mit *doorman* oder anderer Port-Knocking-Software, die verschlüsselte Verbindungen nutzt.
- •Hat ein Angreifer Zugriff auf das System, kann er in den Logs der Firewall nach auffälligen abgelehnten Verbindungen etwa kurz vor einer SSH-Verbindung suchen und so die richtige Sequenz herausfinden.

•Beim Absturz der Port-Knocking-Software ist evtl. das komplette System nicht mehr zu erreichen, da die Ports in der Firewall geschlossen bleiben.

•Und zuletzt kann es wie schon erwähnt sein, dass die Verbindungsversuche zu den definierten Ports von anderen Firewalls (etwa im Unternehmen des Benutzers) geblockt werden und somit ein Zugang zum Server nicht möglich ist (Lösung: z.B. *webknocking*)



Abschließend bewerte ich den Einsatz von Port-Knocking-Software als durchaus positiv, solange man sie als zusätzliche Sicherheitsschicht ansieht und die sichere Konfiguration der eigentlichen Dienste nicht außer Acht lässt. Sie dient als äußerst einfache Möglichkeit nicht nur Script-Kiddies sondern auch professionellere Hacker abzuwehren, da der Nachweis ihres Einsatzes ohne konkreten Hinweis so gut wie unmöglich ist (das System erscheint als „offline“). Und selbst wenn der Angreifer weiß, dass Port-Knocking-Software verwendet wird, ist das Knacken der Port-Sequenz mehr als schwierig und vor allem auch äußerst auffällig.

Ende der Präsentation

- Danke für Ihre Aufmerksamkeit!
- Fragen?