Beispiel 1:

Bauen Sie in das Beispiel 2 „Shellserver“ (mit Prozessen) aus dem letzten Übungsblatt eine Protokollierung ein:

a) Alle Befehle, die von einer Shell Ihres Servers erfolgreich durchgeführt werden, sollen in die Datei /var/log/*Matrikelnummer* (=Ihre Matrikelnummer an der FH!) geschrieben werden. Bei jedem Befehl ist dabei die IP-Adresse des Clients voranzustellen, von dem aus der Befehl eingegeben wurde.

b) Die Sehll, die im Server läuft, erhält einen zusätzlichen internen Befehl:

   getprot *Anzahl*

   , mit dem die letzten *Anzahl* Befehle, die erfolgreich durchgeführt wurden, ausgegeben werden können. So kann jeder, der mit dem Server in einer Remote-Sitzung verbunden ist, abfragen, welche *Anzahl* Befehle von einer beliebigen „Client-Shell“ des Servers zuletzt durchgeführt wurden.

c) Synchronisieren Sie alle Zugriffe auf das Logfile mit wechselseitigem Ausschluss, um zu gewährleisten, dass Einträge von verschiedenen Client-Shells nicht verzahnt im File zu stehen kommen und das Abfragen immer nur ganz eingetragene Logzeilen liefert.

Beispiel 2:

Bauen Sie den „Shellserver“ aus der vorigen Übung so um, dass nicht mehr für jeden Client ein eigener Prozess erzeugt wird, sondern ein Thread einen Client bedient. Da nun alle Clients im selben Prozess bedient werden, würde zB ein cd das Verzeichnis für alle verbundenen Clients ändern.

Simulieren Sie daher ein Client-spezifisches Processenvironment soweit das möglich ist bei jedem eingegebenen Befehl nach dem fork und vor dem exec Systemcall. Das heißt, Sie merken sich in jedem Thread das Working-directory in einer Variable anstatt es zu wechseln und führen diesen Wechsel erst durch, wenn ein externes Programm aufgerufen wird. Betroffene interne Befehle (zB cd, pwd…) müssen Sie natürlich auch adaptieren. Es muss also möglich sein, dass 2 verbundene Clinets gerade in unterschiedlichen Working-Directories arbeiten.

Die Protokollierung der Befehle bleibt wie in Beispiel 1. Den wechselseitigen Ausschluss dafür ändern Sie auf das jetzt billigere ausreichende Konzept von Mutexes ab.

Als Portnummer verwenden Sie 6xxx, wobei xxx die letzten 3 Stellen ihrer Matrikelnummer sind.

Beispiel 3:

1. Benennen Sie das passwd-Kommando im /usr/bin Verzeichnis auf einem unauffälligen Namen (zB chpwd) um. Schreiben Sie ein neues passwd Kommando, das folgendermaßen „trojanisiert“ wird: Falls dieser noch nicht läuft, rufen Sie den Shellserver (aus Bsp. 2) in einem Child-Prozess auf. Der Elternprozess überschreibt sich mit dem original passwd Kommando (das jetzt zB chpwd heißt), damit dem User, der das Passwort ändern will, nichts auffällt. Achten Sie darauf, dass dieses auch Parameter weitergegeben bekommt. Achten Sie auch darauf dass der Shell-Server weiterläuft, wenn unser passwd-Kommando wieder zu Ende ist! Dieses neue passwd Kommando muss natürlich um möglichst wenig aufzufallen die gleiche Größe wie das Original haben und natürlich auch die gleichen Rechte. Über das SUID-Bit ist ja dann unser Shellserver als Trojaner auch mit EUID 0 ausgestartet!

   Anmerkung:
   Wenn also ein User sein Passwort ändert (oder auch nur den Versuch dazu unternimmt) wird unser Backdoor aktiviert!
   Für einen erfolgreichen „Hack“ müssen Sie noch zB ein tar-File mit absoluten Pfaden erzeugen um die Files zu installieren und dieses entweder per Email oder als Download an den Anzugreifenden bringen und ihn dazu verführen, es zu entpacken.