

# Sicherheitschancen und -risiken von Open-Source-Software im Geschäftsumfeld

## Open Source '03

- Gründe für OSS
- Modularität, Nachvollziehbarkeit und Transparenz
- Verfügbarkeit von Sicherheitstechnologien
- Sicherheit als Philosophiefrage
- Chancen und Risiken im Business-Umfeld
- Fazit

Dr. Marcus Holthaus  
IMSEC, Dersbachstrasse 312a, CH-6330 Cham  
041 780 0011, Fax 041 780 0021  
[www.imsec.ch](http://www.imsec.ch), [Marcus.Holthaus@imsec.ch](mailto:Marcus.Holthaus@imsec.ch)  
Fachgruppe Security (FGSec)  
[www.fgsec.ch](http://www.fgsec.ch), [Marcus.Holthaus@fgsec.ch](mailto:Marcus.Holthaus@fgsec.ch)



## About us...

- Ueber Marcus Holthaus
  - 1987-1995 Programmierer und Projektleiter bei einer Softwarefirma für DOS- und Windows-Programme
  - 1990-1999 Studium und Dissertation Wirtschaftsinformatik
  - Seit 1995 Nutzer, Programmierer und Projektleiter für Linux-basierende Systeme (v.a. Debian)
  - Vorstandsmitglied Fachgruppe Security (FGSec), Geschäftsführer PKI-Forum Schweiz
- Ueber IMSEC
  - Management und Koordination von Projekten zur Informationssicherheit (Management / Strategie / Technologie)
  - Die IMSEC-eigene Infrastruktur inkl. der für Kunden betriebenen Systeme basiert seit Gründung 1999 fast ausschliesslich auf OSS



# Gründe für OSS

## Sicherheit

- Modularität, Nachvollziehbarkeit und Transparenz
- Verfügbarkeit von Sicherheitstechnologien
- Definierte Qualität
- Community Support & Sicherheitsphilosophie
- Sichere Vorkonfiguration
- Direkte Konfiguration (mit fakultativem Umweg über graphische Oberfläche)

## Funktionalität

- Weitestgehende Kompatibilität mit vorherrschender Windows-Welt
- Pakete für jeden Bedarf
- Softwareentwicklungsumgebung inklusiv
- Pakete integrieren modulare Funktionalitäten
- Ausgezeichnete Kombinierbarkeit der Funktionalitäten (à la UNIX)
- Graphische Oberfläche(n)
- Kompatibilität zu fast aller Hardware, Skalierbarkeit
- Preis

# Modularität, Nachvollziehbarkeit und Transparenz

- Kleine Pakete mit genau umschriebenem Funktionsumfang
- Application Programming Interfaces (API)
  - Einheitlich und konsequent umgesetzt
  - Unterschiedlich pro Paket
  - Als einzige Schnittstellen nach aussen
- Klares Logging-Konzept
  - Zentrale Syslog-Funktionen
  - Debugging Levels
- Interface-, Filesystem and Process Dumps and Traces
- Updates per Package
- Distributionen mit klaren Zielsetzungen (Debian: stable / testing / unstable)
- Paketmanagement
  - Detailliert beschriebene und technisch formulierte (d.h. automatisch interpretierbare) Abhängigkeiten der Pakete und ihrer Versionen untereinander
  - Klare Aussagen zu erwartendem Speicher- und Platzbedarf und zu Veränderungen der Konfiguration
- Erstellung
  - Einheitliche Richtlinien und Vorgaben zur SW-Erstellung und zur Herstellung von Paketen und Kernel selbst
  - Autoconf, automake, debian-build-helpers
  - Qualitätsanforderungen an Package Maintainer

## Verfügbarkeit von Sicherheitstechnologien (1)

- Alle Technologien zur Implementation von Sicherheit sind
  - Weitestgehend als OSS für Linux verfügbar, in weiten Teilen auch für BSD
  - Ansonsten für Unix verfügbar und können portiert werden (sofern OSS)
  - Werden von den Herstellern kommerzieller Produkte laufend portiert
- Es gibt kaum noch eine Technologie-Plattform, auf der nicht wenigstens eine Linux- oder BSD-Variante funktioniert, vom HandHeld bis zum Grossrechner
- Spezifische Distributionen für erhöhte Anforderungen: Astaro, EnGarde, NSA Security Enhanced Linux, Immunix, Trusted Linux etc. (teilweise kommerziell)
- Verfügbarkeit
  - RAID (SW+HW), UPS, Cluster, Grid, Round-Robin, Mutli-Processor etc., Always-on-Philosophie von Unix-Systemen
- Integrität
  - Virus und Spamfilter (Server und Client), Integrierte Firewall, Konfiguration von Proxies für alle Dienste möglich
  - Journalling Filesystems etc.
  - Integritätskontrolle der Distribution und der Updates
  - Debian: Alle Package Maintainer sind Teil eines Web of Trust (mit gpg oder pgp)

## Verfügbarkeit von Sicherheitstechnologien (2)

- Vertraulichkeit
  - GNU Privacy Guard / Pretty Good Privacy, X.509 über OpenVPN, die meisten E-Mail-Programme und fast alle Browser sind X.509-fähig (z.B. für SSL), alle bekannten Verschlüsselungsalgorithmen verfügbar
  - HD-Encryption, VLAN, VPN, SSH Tunneling (z.B. Für Remote Desktop Access mit RDP oder VNC), TCP/TPM-Support etc.
- Identifikation, Authentisierung, Autorisierung
  - OpenLDAP, Single Sign-On, Pluggable Authentication (PAM), Kerberos, Radius, SSL (Client + Server Certs), etc.
- Verbindlichkeit
  - Unterstützung digitaler Signaturen, Chipartensysteme, etc.
- Zugriffsschutz
  - Verschiedene mediengebundene und Netzwerk-Dateisysteme mit unterschiedlichsten Möglichkeiten der Zugriffsvergabe
  - Unix Access Right, Rules-based Access, Access Control Lists (ACL), Departments / chroot etc.
- Netzwerkmanagement (SNMP) Support, Routingkontrolle, Firewalling, Traffic Shaping, etc.

# Philosophie & Sicherheit

- Voraussetzungen von Sicherheit: Detailverständnis und Kontrolle
  - Fähigkeit und Bereitschaft, kritische Prozesse im Detail verstehen und nachvollziehen zu können
  - Konsequente nachhaltige Verhinderung ungewünschter Vorgänge und Eigenschaften
  - Dies ist nur erreichbar durch
    - absolute Transparenz (White Box) oder
    - absolute Einschränkung von aussen (Black Box in Sand Box)
- FOSS ist eine White Box, Closed Source Software ist Black Box
- Sicherheit ist eine Frage der Philosophie der Nutzer (Systembetreuer und Anwender)
  - Will er verstehen? => Motivation
  - Kann er verstehen ? => Ressourcen, Dokumentation
  - Versteht er ? Handelt er? => Setup, Konfiguration, Betrieb mit laufender Kontrolle

# Philosophien (1)

Aspekt	Windows / Microsoft	OSS (v.a. Debian, BSD) / Open Source Community
<b>Ideale</b>	Markt, Marktdeckung, kommerzieller Erfolg, Erreichung und Erhalt eines bestimmenden Einflusses	Technologische Möglichkeiten, qualitativ hohe Ansprüche, „es richtig machen“
<b>Motivation &amp; Ziele</b>	Dominante Marktstellung, Integration aller interessanten Konzepte in eigene Produkte, minimale Interaktion mit Fremdprodukten	Idealismus, Anerkennung, Kooperation zwischen den Entwicklern, Verzicht auf monetären Erfolg durch Produktverkäufe, Nachbau kommerziell erhältlicher Produkte, maximale Interaktion mit Fremdprodukten
<b>Promotoren</b>	Eine inzwischen sehr grosse Unternehmung	Gemeinschaft unabhängiger meist hoch qualifizierter Autoren, einzelne Firmen und Staaten, die OSS vorantreiben, aber nicht die „community“ repräsentieren, und die Services verkaufen möchten (nicht Produkte)
<b>Produkte</b>	As simple and as powerful as possible, gut integrierbar in Windows-Welt, everything works out-of-the box, fast-to-market	Gut integrierbar in heterogener Welt, Qualitativ ausgereift oder als unstable gekennzeichnet, Zusatzfunktionen müssen manuell aktiviert werden, transparent-quality-to-public
<b>Modularität</b>	Gering, starke Interdependenzen (feststellbar bei Patching)	Hoch, hierarchische Abhängigkeiten abgegrenzter Pakete untereinander, Patching von Paketen

## Philosophien (2)

Aspekt	Windows / Microsoft	OSS (v.a. Debian, BSD) / Open Source Community
<b>Kunden und Nutzer</b>	Gemäss Nachfragemarkt, kommerziell, Nutzung nur nach vorgegebenen Anweisungen	Autoren selbst, nicht-kommerzieller Nachfragemarkt, Motivation zum Studium und Erweiterung der Funktionsweise
<b>Verhalten intensive Nutzer</b>	Point and Click, Referenz ist Inhalt der grafischen Nutzeroberfläche, dokumentiertes API und Hilfeseiten	Terminal, Referenz ist ASCII-basiertes Config-File, technische Doku inkl. API und Source-Code
<b>Sicherheit</b>	Sicherheit als notwendige Eigenschaft, die lange vom Markt nicht nachgefragt wurde, aber plötzlich geliefert werden muss	Sicherheit als integraler Bestandteil der Motivation, Pakete mit schlechter Sicherheit werden nicht (oder nur mit Vorbehalten und Kennzeichnung) in die Distribution aufgenommen
<b>Infos über Schwachstellen</b>	Entdecker -> MS -> MS-Interne Produktion -> Mailing-Listen mit Ankündigung Patch und Download Location	Entdecker -> öffentliche Mailingliste -> mehrere alternative Patch-Vorschläge (von jedem individuell anwendbar) -> Konsolidierung in Update-Paket -> Mailing-Listen mit Ankündigung des Update-Pakets und Download Location
<b>Kontrollierbarkeit</b>	Keine letztendliche Kontrolle, Vertrauen in Hersteller	Selbstkontrolle möglich
<b>Verlässlichkeit</b>	Als Marketinginstrument... und notwendige Eigenschaft des Produkt	Als Qualitätsmerkmal vor Akzeptanz eines neuen Pakets

## Debian Social Contract

Debian GNU/Linux Social Contract with the Free Software Community (Excerpts)

1. Debian Will Remain 100% Free Software
2. We Will Give Back to the Free Software Community
- 3. We Won't Hide Problems**
4. Our Priorities are Our Users and Free Software
5. Programs That Don't Meet Our Free-Software Standards  
[...] "contrib" and "non-free" areas in our FTP archive [...] Thus, although non-free software isn't a part of Debian, we support its use, and we provide infrastructure for non-free software packages.

The Debian Free Software Guidelines

1. Free Redistribution
2. [...] The program must include Source Code and must allow distribution [...]
3. [...] The license must allow modifications and derived works [...]
4. Integrity of The Author's Source Code
5. No Discrimination Against Persons or Groups
6. No Discrimination Against Fields of Endeavor
7. Distribution of License
8. License Must Not Be Specific to Debian
9. License Must Not Contaminate Other Software
10. Example Licenses: "GPL", "BSD", and "Artistic" [...]

# Chancen und Risiken im Geschäftsumfeld

## Chancen

- Absolute Transparenz: jeder kann letztendlich all seine Probleme selbst lösen (oder Lösungen beauftragen)
- Wesentlicher Zeitgewinn: Es wird nur noch die eigene Zusatzfunktion erstellt (Delta statt Neuentwicklung)
- Geringere Abhängigkeit
  - Von Lieferanten
  - Von Einzelpersonen
- Einfache Anpassung der SW an eigene Bedürfnisse
- Zukunftssicherheit durch Freigabe der eigenen Anpassungen
- Stabilität
- Direkter Link von Forschung in Praxis

## Risiken

- Historisch
  - Verfügbarkeit der bereits genutzten bekannten Software
  - Umstellungsaufwand
  - [punktuelle Inkompatibilitäten]
- Einstiegsaufwand und Philosophie
  - OSS verlangt Qualitätsbewusstsein beim Entwickler und Systemadmin.
  - Vom „Point and Click“ zum Verständnis der Abläufe
  - Re-Implementationsaufwand
- [Kein Verantwortlicher mehr? Der Autor ist verantwortlich!]

# Fazit

- OSS ist reif für den Business-Einsatz – im Server schon länger, auf dem Desktop jetzt auch.
- Technologisch gibt es keinen Grund, nicht auf OSS zu wechseln. Die notwendigen Sicherheitstechnologien sind alle verfügbar.
- Die wesentlichen Sicherheitsfragen liegen im Management. Der Einsatz von OSS führt nicht automatisch zu höherer Sicherheit. Die konsequente strukturierte Verfolgung von Sicherheit als Ziel und die Anwendung einer sicherheitsorientierten Denkweise führt sowohl bei der Nutzung von CSS wie auch von OSS zu verbesserter Sicherheit.
- Linux hat als Hilfsmittel zur Erhöhung der Sicherheit mehrere Vorteile gegenüber Windows. Es liegt als Werkzeug mindestens ebenso gut „in der Hand“ wie Windows. Der „default disabled“ Ansatz verhindert die fahrlässige Einrichtung von Schwachstellen.
- Kapselung als Systemprinzip ist weiterhin sinnvoll, aber der Inhalt der Kapsel muss sichtbar sein. Open Source Software ist transparent – jeder Nutzer kann bei Bedarf jeden einzelnen Schritt nachvollziehen.
- OSS bietet deutlich mehr Chancen als Risiken im Sicherheitsbereich.
- OSS Software bietet dem Nutzer Freiheit (free as in freedom) und erlaubt die Errichtung nachvollziehbarer, sicherer, verlässlicher Systeme.