genua
Soviel ist sicher.

IT Security
Made in Germany

Aufbau und Funktionsweise von Hochsicherheits-Firewalls

Alexander von Gernler

<gernler@genua.de>

Gastvorlesung in *Sicherheit in Rechnernetzen*

Uni Potsdam, 27. Juni 2013, 12:15, Haus 6, Raum H.01

genua

Übersicht

- Einleitung
  - genua und der Referent
  - Einbettung in die Vorlesung
- Unsere Sicht auf Firewalls
  - Abtrennung von Niveaus
  - Hochsicherheit
  - Zertifizierung
- Aufbau des Produkts
  - ALG+PFL
  - Stufenfilterung
  - minimalistisches OS
  - Fremdschutz: Rekursiver
  - Virenscanner
  - Web 2.0 und ALGs
- Softwaretechnik
  - Gutes Handwerk
  - Agile Verfahren
  - Roadmaps
- Vermischtes
  - OpenSource-Szene
  - Pläne für die Zukunft

genua

Übersicht

- Einleitung
  - genua und der Referent
  - Einbettung in die Vorlesung
- Unsere Sicht auf Firewalls
  - Abtrennung von Niveaus
  - Hochsicherheit
  - Zertifizierung
- Aufbau des Produkts
  - ALG+PFL
  - Stufenfilterung
  - minimalistisches OS
  - Fremdschutz: Rekursiver
  - Virenscanner
  - Web 2.0 und ALGs
- Softwaretechnik
  - Gutes Handwerk
  - Agile Verfahren
  - Roadmaps
- Vermischtes
  - OpenSource-Szene
  - Pläne für die Zukunft

Was ist genua?

- Spezialist für IT-Sicherheit, gegründet 1992
  - Kirchheim bei München
  - 184 Mitarbeiter im Juni 2013
- Hochsicherheits-Firewalls
  - OpenBSD-basiert
  - BSI-zertifiziert nach Common Criteria, EAL4+
- Stehen bei...
  - MAN, RTL II, Hypo-Vereinsbank, Klüber, …
  - BSI, Generalbundesanwalt, Stadt München, Deutscher Bundestag, Bundeswehr, …

genua

Über den Referenten

- Forschung und Entwicklung
  - Wir machen den Weg frei für Produkte!
  - Fragen von morgen finden und adressieren
  - Verbund-Forschungsprojekte
- Technischer Botschafter
  - Kontakt zu Universitäten und Hochschulen
  - Sammlung/Weitergabe von Wissen
  - Vorträge und Fachartikel zu IT-Themen
- Freie Software
  - OpenBSD-Committer 2005-2010
  - `$ ssh -o 'VisualHostKey yes'`. Did it.

```
The authenticity of hos
RSA key fingerprint is 
+--[ RSA 2048]----+
|  ...oo          |
|.  ... o .       |
| o...   *        |
|. +.   +         |
| + + . .S        |
|  = o o .        |
| . . + o .       |
|    . E .        |
|       .         |
+-----------------+
```

genua

Vorlesung 4: Risiken (1)

Viele dort vorgestellte Angriffe können verhindert werden.

- Mit Paketfilter (OSI-Schicht 2-4)
  - Blockieren von Ports
  - SYN-Proxy
  - Normalisierung von TCP-Headern
  - Randomisierung von ISNs

7	Application
6	Representation
5	Session
4	Transport
3	Network
2	Data Link
1	Physical

genua

Vorlesung 4: Risiken (2)

Viele dort vorgestellte Angriffe können verhindert werden.

- Mit Application Level Gateway (OSI-Schichten 5-7)
  - Protokollbasierte Angriffe abwehren (Smurf, Land, Ping of Death)
  - Filterung/Modifikation von Inhalten von Datenpaketen
  - Virenscanning on-the-fly

7	Application
6	Representation
5	Session
4	Transport
3	Network
2	Data Link
1	Physical

genua

Vorlesung 5: Firewalls (1) Hochsicherheits-Ansatz

- Kaum noch Auswertung von Angriffen (Logfiles, IDS)
- zu teuer
- Angreifer eh nicht belangbar
  - gekaperter Rechner
  - internationale Jurisdiktion
  - Bulletproof Hosting (Russland, Asien)
- Heutige Angriffe indirekt: Mail, Web
  - legaler Traffic im Sinne einer Firewall
  - Kunden nehmen Firewall zum Durchsetzen von Richtlinien (Policies) her

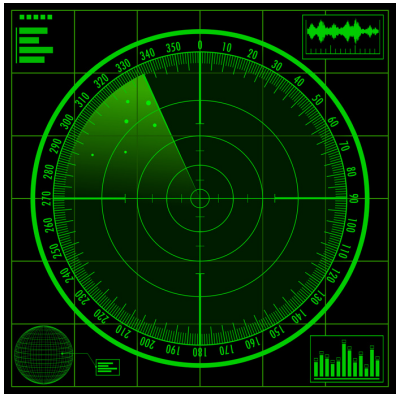
genua

Vorlesung 5: Firewalls (2) Normalbenutzer-Ansatz

- Kunden, die früher nur Paketfilter hatten
- Administratoren, die keinen Überblick über Traffic haben
- Überall, wo *gut genug* ausreicht
- Visualisierung des Traffics. Buzzwords Next Generation Firewalls und Unified Threat Management
  - Wollen das *zu teuer* bei der Auswertung von Angriffen entfernen
  - Betrachtet wird Verkehr (Anwendungen), nicht Angriffe
- Erfassung des IST-Zustandes, dann iterativ festzurren
- Reicht vielen Firmen schon. Besser als nichts

genua

Vorlesung 6: IDS (1): Klassisch

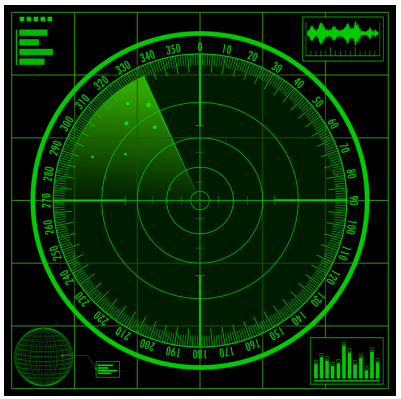


*„Oh mein Gott, wir werden angegriffen!
Alle Energie auf die Nordfirewalls!“*

- Klassisches IDS ist tot
  - Niemand interessieren Angriffe von extern, solange sie nicht erfolgreich sind
  - Viele False Positives, menschlicher Betreuungsaufwand
  - Angriffe ohnehin indirekt, s.o.
  - Selbst wenn echter Angriff gefunden: Was dann?
  - Rechtsabteilung einschalten? SRSLY? Come on.

genua

Vorlesung 6: IDS (2): Klassisch



*„Oh mein Gott, wir werden angegriffen!
Alle Energie auf die Nordfirewalls!“*

- Klassisches IDS ist tot
- Daher meist nur relevant fuer
  - Akademische Betrachtung
  - Zur Erfüllung willkürlicher Sicherheitsrichtlinien
    - Beispiel: geschlossenes Netz; Daten nur zwischen zwei IPs; Alle Daten verschlüsselt – dort IDS. Jawoll! `m(`
- Viele IDS schauen nicht tief genug
  - Angriffe komplexer als mit Pattern Matching erkennbar

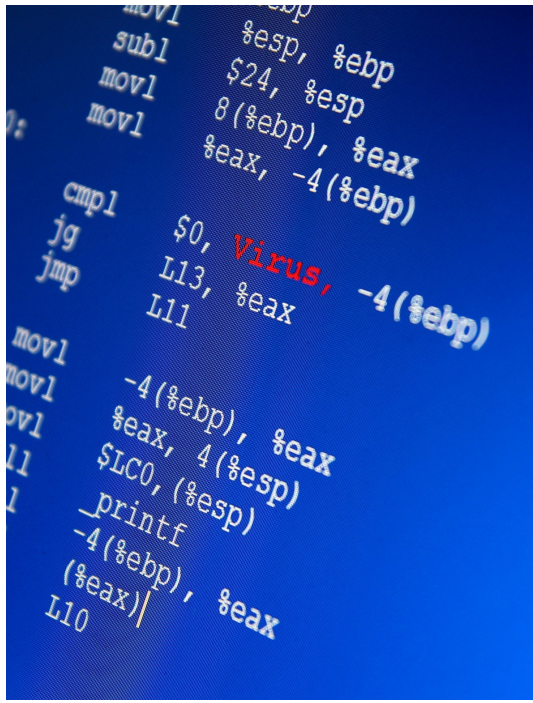
genua

Vorlesung 6: IDS (3): Modern

- Von Interesse sind nicht alle, sondern die erfolgreichen Angriffe
- Betrachtet wird nach extern gerichteter Verkehr
- Zentrale Frage: Welche internen Rechner sind infiziert?
  - Buzzwords:
    - Unified Threat Management
    - Extrusion Detection
    - Data Leak Prevention
  - Hauptproblem: Abfluss von Informationen
  - Wirtschaftsspionage

Vorlesung 6: IDS (4): Modern

- Welche internen Rechner sind infiziert?
- Gehackte Clients verhalten sich (hoffentlich) anders
  - DNS-Anfragen zu Domains mit kurzer TTL
  - Verbindungen in abstruse IRC-Kanäle (Command&Control)
  - Ungewöhnliche HTTP-Verbindungen
  - Benutzung von `CONNECT` auf SSL-Proxies



- Aber C&C-Verbindungen können normal aussehen
  - Twitter-Nachrichten
  - GIF-Bilder in Webseiten (Steganographie)

genua

Übersicht

genua

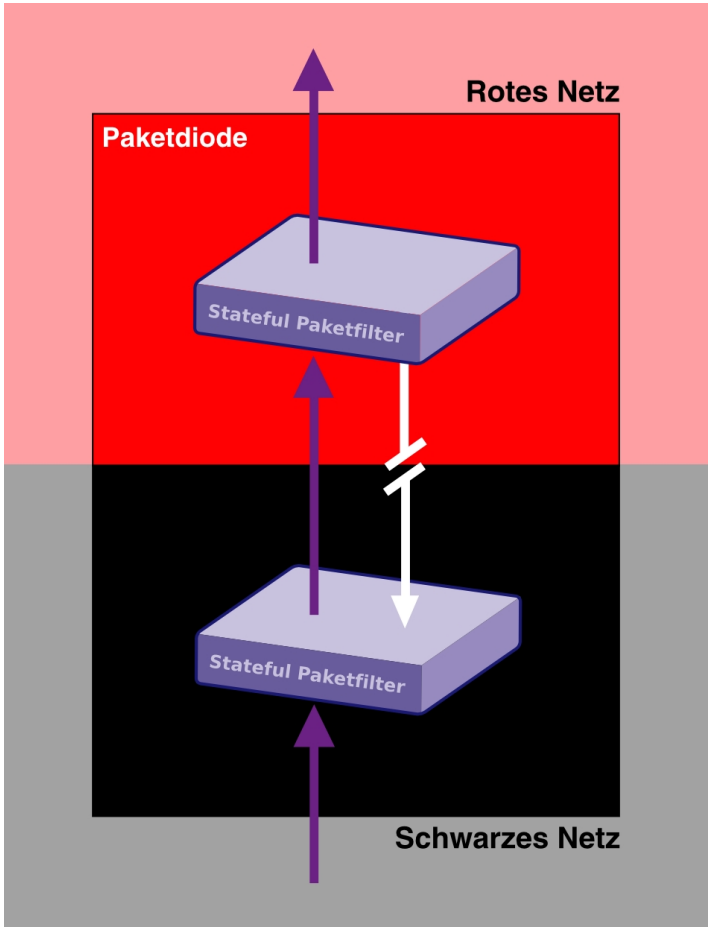
Firewalls aus Sicht von genua

- Firewalls trennen Netze verschiedener Sicherheitsniveaus
- Ausgedrückt durch Sicherheitsrichtlinie (Policy)
- Festlegung und Durchsetzung, welche Daten wohin fließen
- Beliebig viele Netzsegmente
  - meist intern, extern und DMZ
  - Spezialnetze denkbar
- Keine dynamische Erkennung von Netzwerkprotokollen
- Keine unvollständige Prüfung von Daten
- Kein Durchlassen auf Verdacht

genua

Einsatzszenario Hochsicherheit: Paketdiode

- Datenpakete nur in eine Richtung
- Protokoll muss für Einwegkommunikation ausgelegt sein (z. B. UDP)
- Möglicher Nachweis: Durchtrennung einer von zwei Glasfasern

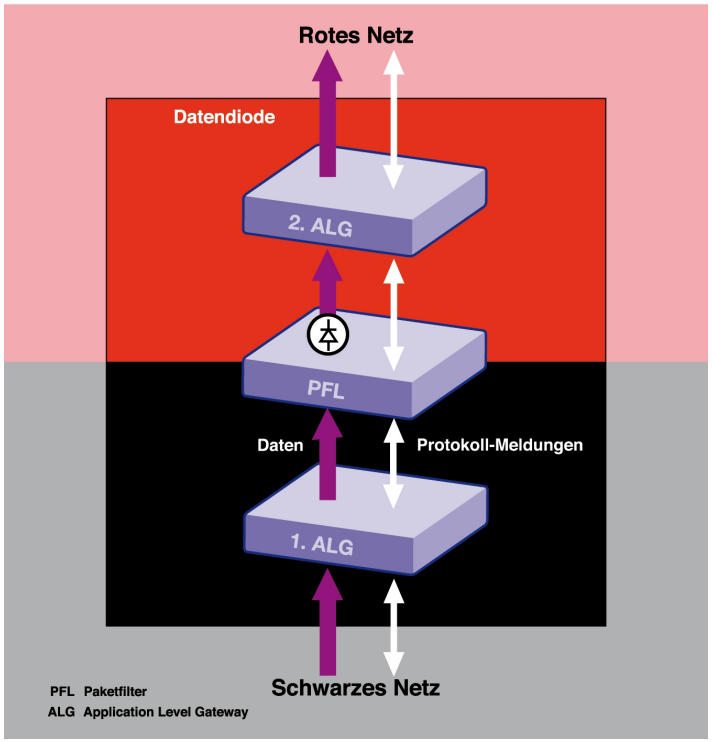


genua

Einsatzszenario Hochsicherheit: Datendiode

- Inhalte nur in eine Richtung
- Schwieriger, weil Pakete durchaus in andere Richtung dürfen (z. B. ACK oder RST bei TCP)
- Sublime Kanäle denkbar: Protokollfelder, Timing, absichtliche Fehler
- Müssen durch Normalisierung minimalisiert werden

Informationsfluss bei der genugate-datendiode



genua

Zertifizierung

- Zweck
  - Nachweis von Sicherheit
  - Schaffung von Vertrauen
  - Dokumentation, Verifikation, Qualität
  - Öffnung von Märkten

genua

Zertifizierung

- Framework Common Criteria
  - Internationales Zertifizierungsschema
  - Systematik von Templates und Vorschriften
  - ständige Weiterentwicklung des Frameworks selbst
    1996 Version 1.0
    2013 Version 3.1r4
  - Hier speziell: Common Criteria for Information Technology Security Evaluation

genua

Übersicht

genua

Hinweis zur Notation

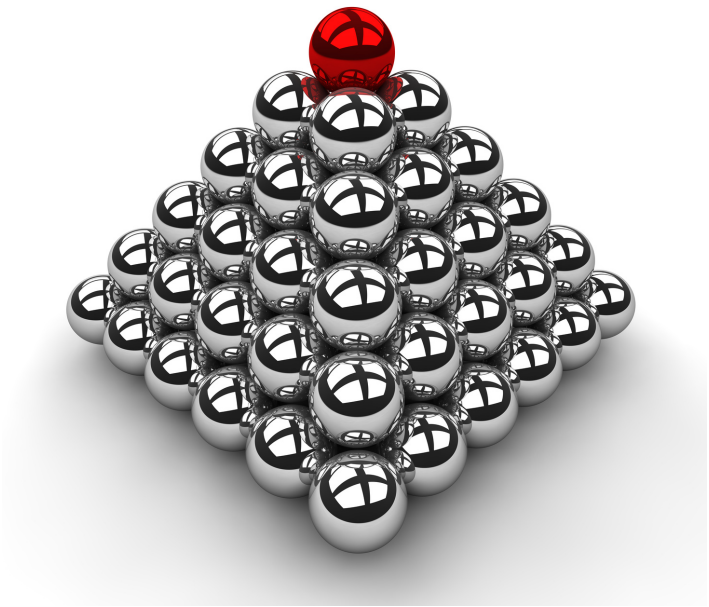
WTF(7) ?

- Ab jetzt öfter mal Referenzen auf Spezialbegriffe des Betriebssystems
- Diese sind in `Maschinenschrift` gesetzt
- Angehängte Klammern() referenzieren das Kapitel des Betriebshandbuches (neudeutsch: *System Manual*)
- Beispiele
  - `ls(1)` – `ls`, das Anwendungsprogramm
  - `reboot(8)` – `reboot`, Anwendungsprogramm mit vorausgesetzten Privilegien
  - `reboot(2)` – `int reboot(int);` der Systemaufruf

genua

Ach ja… Gliederung des Unix-Manuals

1. General Commands
2. System Calls
3. Subroutines
4. Special Files
5. File Formats
6. Games
7. Macros and Conventions
8. Maintenance Commands
9. Kernel Interface



genua

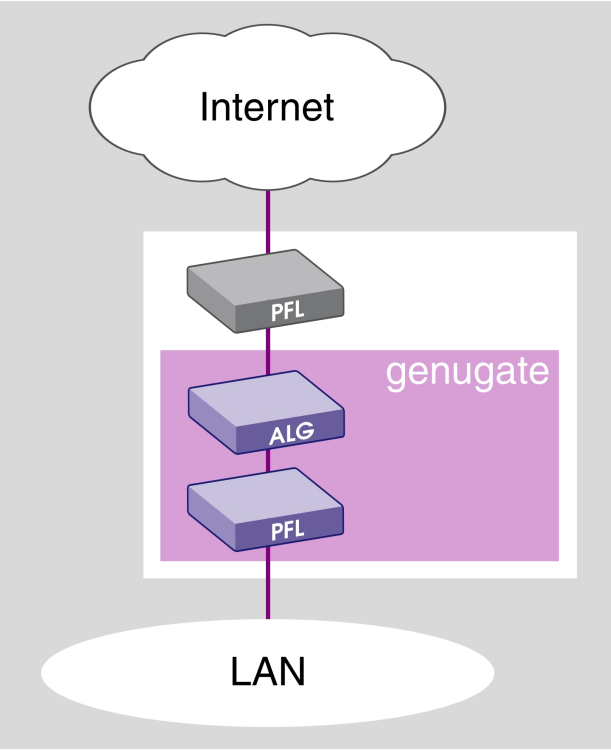
Zweigeteilte Firewall

- Zwei unabhängige physikalische Rechner
- Application Level Gateway
  - Filterung auf Anwendungsebene: Proxies
  - Eigene Userland-Prozesse, die Daten annehmen
  - Trotz Proxy-Funktionalität Transparenz möglich
- Paketfilter
  - Filterung auf Netzwerkebene
  - Mittels Standard-Paketfilter OpenBSD `pf(4)`
  - Konfiguration `pf.conf(5)` generiert

genua

P-A-P

- Aufbau nach BSI-Empfehlung
- Serielle Schaltung von Paketfilter-ALG-Paketfilter
  - Kompromittierung einer einzelnen Komponente kompromittiert nicht automatisch Gesamtsystem
- Netzseparation
  - Komplette Trennung der Schichten 1-4



genua

Paketfilter

- Filterung von OSI-Schichten 1-4
- *stateful filtering*
- TCP-Verbindungen werden weitergeleitet (forwarding)
- Routing-Entscheidungen
- Billig, meist im Kernel abzuhandeln
- Zur Erschöpfung erforschte Technik
- Wenig Neuerungen, wenig Überraschungen

genua

Application Level Gateway

- Filterung höherer Schichten
- Verbindung terminiert immer am ALG
- Auf der anderen Seite wird eine neue Verbindung initiiert
- Hacks im Kernel für Transparenz (divert sockets)
- Pro Protokoll eigener Proxy (Relay) nötig
- Ständig was zu tun
- Rechenintensive Inhaltsinspektion als Userland-Prozess

genua

Schutzbegriff

*„Ziehen Sie die Atemmaske von der Decke und setzen Sie diese auf. **Danach** helfen Sie Kindern.“*

- Wer sich selbst nicht schützen kann, kann auch keinen anderen schützen!
- Daher Differenzierung von Schutzmechanismen einer Firewall:
  1. Selbstschutz: Nicht ge-`0`wned werden ;)
  2. Fremdschutz: Netzwerkverkehr filtern/blockieren/modifizieren
- (2) ist die Kernaufgabe, diese bedingt aber (1) !

genua

Selbstschutz: OpenBSD als Basis

- Schlankes, freies Unix-artiges Betriebssystem
- Primärer Fokus auf Sicherheit
  - *secure by default*
- Releases jedes halbe Jahr
- konservative Entwicklungspolitik
- `http://www.openbsd.org/`
- Basissystem bei genua weiter minimalisiert und gehärtet

genua

Selbstschutz: `securelevel(7)`

```
# sysctl kern.securelevel=2
kern.securelevel: 1 -> 2
#
```

- Immer strenger werdende Beschränkungen
- Levels

-1	Permanently insecure mode
0	Insecure mode
1	Secure mode
2	Highly secure mode

genua

Selbstschutz: `securelevel(7)` Details

-1 Debugging
- Gleiche Wirkung wie Level 0
- `init(8)` wird Level aber nie anheben

0 Boot und Single User
- Dateiflags dürfen gesetzt und gelöscht werden
- Zugriff auf Geräte laut gesetzter Permissions
- `init(8)` hebt Level nach Boot auf 1 an

genua

Selbstschutz: `securelevel(7)` Details (2)

1. Standardeinstellung
   - Securelevel darf nur noch erhöht werden
   - `/dev/mem` und `/dev/kmem` nicht mehr schreibbar
   - Gemountete Festplatten roh nicht mehr schreibbar
   - Dateiflags dürfen nicht mehr gelöscht werden
   - Kernelmodule können nicht geladen oder deaktiviert werden
   - Viele `sysctls` eingeschränkt
     - `hw.allowpowerdown`
     - `ddb.panic`
     - `ddb.console`
     - `machdep.kbdreset`
     - `machdep.allowaperture`
     - …

genua

Selbstschutz: `securelevel(7)` Details (3)

2 Paranoia

- Alles von Level 1 inclusive
- Beliebige Festplatten roh nicht mehr schreibbar
- Systemzeit darf nicht mehr zurück gesetzt werden
- Paketfilterregeln dürfen nicht mehr geändert werden

genua

Selbstschutz: chflags(1)

```
# ls -alo /bsd
-rwx-r-xr-x 1 root wheel    - 9045137 Dec 8 2012
/bsd*
# chflags schg /bsd
# ls -alo /bsd
-rwx-r-xr-x 1 root wheel schg 9045137 Dec 8 2012
/bsd*
# echo bla > /bsd
sh: /bsd: Operation not permitted
#
```

genua

Selbstschutz: `chflags(1)`

- Zusätzliche Einschränkung neben normalen Permissions
- Bei Verstoss Fehler `EPERM`
- Wichtigste Beispiele

  `schg` system immutable: Für Programme
  `sappnd` system append-only: Für Logdateien

genua

Selbstschutz: Cages

- Basiert auf dem bekannten `chroot(2)`
- `chroot(2)` gutes Mittel für Kompartmentierung und Einsperren von Prozessen mit User-Rechten
- `chroot(2)` aber kein wirksamer Schutz gegen root
- Daher weitere Einschränkungen
  - `kill(2)` eingeschränkt: Signale (`KILL`, `TERM`, `STOP`, …) werden nur an Prozesse innerhalb des Cages zugestellt
  - Bestimmte System Calls funktionieren nicht innerhalb des Cages
    - `reboot()`
    - `ktrace()`
    - `ptrace()`

genua

Selbstschutz: Perl

```
use Perl;
```

„It's like Java, only it lets you deliver on time and under budget."

- Wir erinnern uns: ALG hat Proxies als Userland-Prozesse
- Proxies sind alle in Perl geschrieben!
- *Aber… aber… man muss sowas doch in C schreiben!*
  - Langsamer? Na und? Dickere Hardware!
  - Interpretierte Sprachen haben keine Buffer Overflows
  - Der Interpreter selbst vielleicht, aber der ist gut getestet
  - Hochsprache: Schnelle Entwicklung möglich
  - reguläre Ausdrücke: ideal für Stromverarbeitung

genua

Denksport-Aufgabe

```
# echo '`rm -rf /`' | perl -e 'eval while <>'
#
```

genua

Selbstschutz: Perl Tainted Mode

```
# echo '`rm -rf /`' | perl -Te 'eval while <>'
Insecure dependency in eval while running with
-T switch at -e line 1, <> line 1.
#
```

- Es gibt ja immer noch Hochsprachenangriffe
  - Format String Exploits, SQL Injections, …
  - User Supplied Input
- Daher Tainted Mode:
  - Input von aussen ist erst einmal vergiftet
  - Damit dürfen keine Systemfunktionen aufgerufen werden (`open()`, `system()`, …)
  - Erst mal Matching/Substitution mit regulärem Ausdruck
  - Danach Variable gereinigt und benutzbar

genua

Fremdschutz: Web 1.0 über ALG

- Interpretation von HTTP 1.0 und 1.1
- Proxy-Prozess gibt Requests weiter
  - Normalisierung des Verkehrs
  - Angriffe auf HTTP-Ebene können verhindert werden
  - Überlappende Requests oder Responses
  - Encoding, Chunking, …
- Bestimmte Inhalte können sofort blockiert werden (`*.exe`, `*.doc`, `*.vbs`, …)
- Andere MIME-Objekte gehen durch Virenscanner durch

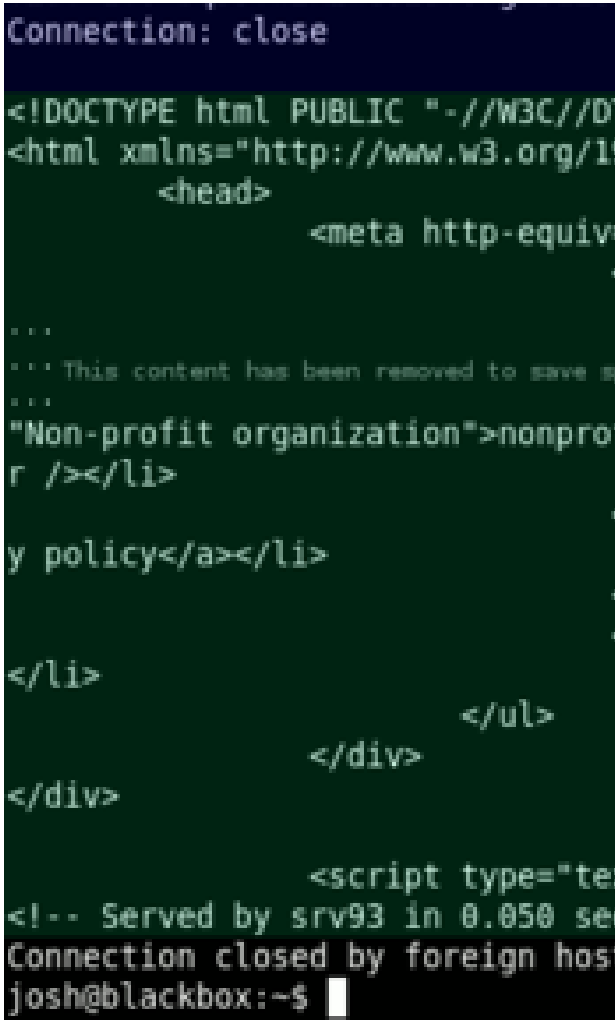
```
josh@blackbox:~$ telnet en.wikipedia.o
Trying 208.80.152.2...
Connected to rr.pmtpa.wikimedia.org.
Escape character is '^]'.
GET /wiki/Main_Page http/1.1
Host: en.wikipedia.org

HTTP/1.0 200 OK
Date: Thu, 03 Jul 2008 11:12:06 GMT
Server: Apache
X-Powered-By: PHP/5.2.5
Cache-Control: private, s-maxage=0, ma
Content-Language: en
Vary: Accept-Encoding,Cookie
X-Vary-Options: Accept-Encoding;list-o
string-contains=centralauth_Token;stri
Last-Modified: Thu, 03 Jul 2008 10:44:
Content-Length: 54218
Content-Type: text/html; charset=utf-8
X-Cache: HIT from sq39.wikimedia.org
X-Cache-Lookup: HIT from sq39.wikimedi
Age: 3
X-Cache: HIT from sq38.wikimedia.org
X-Cache-Lookup: HIT from sq38.wikimedi
Via: 1.0 sq39.wikimedia.org:3128 (squi
Connection: close

<!DOCTYPE html PUBLIC "-//W3C//DTD XHT
<html xmlns="http://www.w3.org/1999/xh
        <head>
```

Quelle: Wikipedia

genua

Fremdschutz: Web 1.0

- Parsing der HTML-Inhalte
- Stoppen animierter Bilder möglich
- Komplette Entfernung von JavaScript möglich
  - Für einige Hochsicherheitsbereiche tatsächlich gewünscht
  - genuas Workstation-Netz kriegt auch erst mal alles ohne JS
  - Seiten, die nicht damit funktionieren, werden auf einem speziellen Browser auf einem anderen Rechner angesurft.

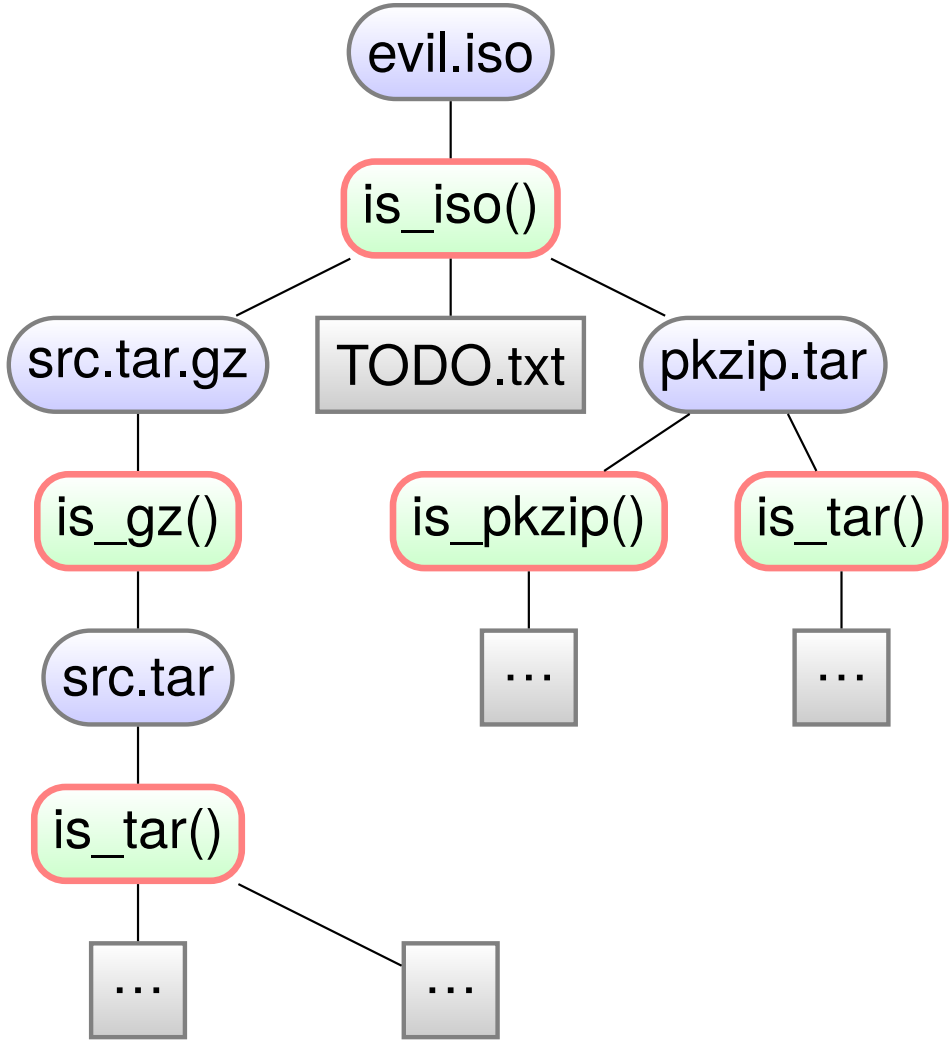


Quelle: Wikipedia

genua

Fremdschutz: Rekursiver Virenscanner

- Zentraler Dienst
  - HTTP-Ströme, Mails, …
- Rekursives Absteigen in Containerformate
  - Formatspezifische Prädikatsfunktionen
  - Blätter nicht weiter expandierbar
- Warum diese Vorverarbeitung?
  - Erhöht Trefferrate
  - Belegt durch eigene Experimente



genua

Fremdschutz: Web 2.0 und ALGs

- Web 2.0 über Firewalls
- Die Leute benutzen Web 2.0!
- JavaScript ausschalten keine Option!
- HTTP/HTML nur noch Transport-Protokoll, obwohl OSI-Schicht 7
- Schachtelung: Mehr Logik in höheren Schichten
- Nötig: Anwendungserkennung in Overlaynetzen
- Ein bisschen Sicherheit ist besser als gar keine

genua

Web 2.0-Traffic über ALG?

- Proxy braucht Parser entsprechender Komplexität
- Beispiele:
  - Feste Strings mit `grep(1)`
  - Variierende Strings mit regulären Ausdrücken
  - HTML mit LALR-Parser
  - JavaScript mit… Oh, wait!
- Chomsky-Hierarchie!
  - allgemeine Sprachen
  - kontextsensitive Sprachen
  - kontextfreie Sprachen
  - reguläre Sprachen
- Problem
  - AJAX, JavaScript, Google API, etc. Turing-vollständig!
  -

genua

Web 2.0-Traffic über ALG?

- Proxy braucht Parser entsprechender Komplexität
- Beispiele:
  - Feste Strings mit `grep(1)`
  - Variierende Strings mit regulären Ausdrücken
  - HTML mit LALR-Parser
  - JavaScript mit… Oh, wait!
- Chomsky-Hierarchie!
  - allgemeine Sprachen
  - kontextsensitive Sprachen
  - kontextfreie Sprachen
  - reguläre Sprachen
- Problem
  - AJAX, JavaScript, Google API, etc. Turing-vollständig!
  - Parser-Explosion!

genua

Web 2.0-Traffic über ALG?

- Proxy braucht Parser entsprechender Komplexität
- Beispiele:
  - Feste Strings mit `grep(1)`
  - Variierende Strings mit regulären Ausdrücken
  - HTML mit LALR-Parser
  - JavaScript mit… Oh, wait!
- Chomsky-Hierarchie!
  - allgemeine Sprachen
  - kontextsensitive Sprachen
  - kontextfreie Sprachen
  - reguläre Sprachen
- Problem
  - AJAX, JavaScript, Google API, etc. Turing-vollständig!
  - Halteproblem!

genua

Web 2.0-Traffic über ALG?

- Proxy braucht Parser entsprechender Komplexität
- Beispiele:
  - Feste Strings mit `grep(1)`
  - Variierende Strings mit regulären Ausdrücken
  - HTML mit LALR-Parser
  - JavaScript mit… Oh, wait!
- Chomsky-Hierarchie!
  - allgemeine Sprachen
  - kontextsensitive Sprachen
  - kontextfreie Sprachen
  - reguläre Sprachen
- Problem
  - AJAX, JavaScript, Google API, etc. Turing-vollständig!
  - Halteproblem!

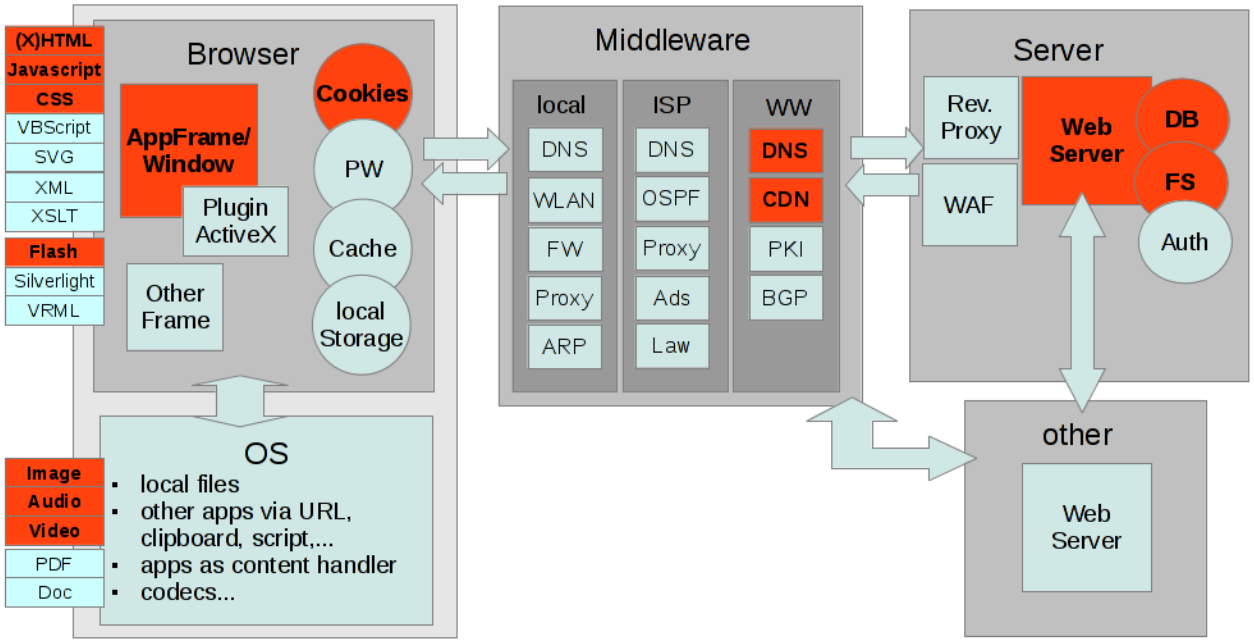
genua

Web 2.0-Traffic über ALG?

- Proxy braucht Parser entsprechender Komplexität
- Beispiele:
  - Feste Strings mit `grep(1)`
  - Variierende Strings mit regulären Ausdrücken
  - HTML mit LALR-Parser
  - JavaScript mit… Oh, wait!
- Chomsky-Hierarchie!
  - allgemeine Sprachen
  - kontextsensitive Sprachen
  - kontextfreie Sprachen
  - reguläre Sprachen
- Problem
  - AJAX, JavaScript, Google API, etc. Turing-vollständig!
  - Halteproblem!

genua

Web 2.0 und ALGs: Herangehensweise



- Idee: Web 2.0 Fingerprinting
  - Kann man den Kommunikationsfluss von Google Maps identifizieren?
  - Maps erlauben, alles andere verbieten?
  - Datenstrom analysieren, nicht auf Quell/Ziel-IP schauen

Forschungsprojekt Padiofire

- Konsortium
  - BTU Cottbus, Lehrstuhl für Rechnernetze und Kommunikationssysteme (Prof. König)
  - FAU Erlangen, Lehrstuhl für IT-Sicherheitsinfrastrukturen (Prof. Freiling)
  - genua mbh
- Assoziierte Partner
  - Uni Innsbruck Lehrstuhl für Technische Informatik (Prof. Dressler)
  - IXIA
- Laufzeit Juli 2011 bis Juni 2013

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

Brandenburgische Technische Universität Cottbus

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG

universität innsbruck

IXIA

genua

Ideen von Padiofire

1. Extraktion von Inhalten aus Web-Datenströmen
   - Kampf mit Encoding, HTTP, TLS, SPDY, MIME
     - UTF-7? UTF-8? UTF-16? ISO-Latin1?
   - Rekursives Öffnen von Containerformaten
     - Können selber wieder beliebigen Inhalt enhalten
     - JavaScript in PDF, XML in JavaScript, JavaScript in SVG
2. Normalisierung von Inhalten
3. Richtige Interpretation der Inhalte
   - Was würde der Client damit tun?
4. Lernen, wie sich Webseiten *anfühlen*
5. Gutartige Änderungen mitlernen; Alarm schlagen bei verdächtigen Änderungen

genua

Fazit

- Binsenweisheit: Absolute Sicherheit nicht erreichbar
- Für Web 2.0 erst recht!
  - Turing-vollständige Inhalte
  - Linux als VM im Browser
  - Stand der Technik bei weitem nicht ausreichend
- Im Hochsicherheitsbereich JavaScript weiterhin undenkbar
- Momentanes Ziel: Sicherheit in der Unsicherheit
- Whitelisting wichtiger Dienste anhand von Verhalten
  - Erkennen von Malware durch Machine Learning
    - oder
  - Verbieten aller unbekannten JS-Inhalte

genua

Übersicht

genua

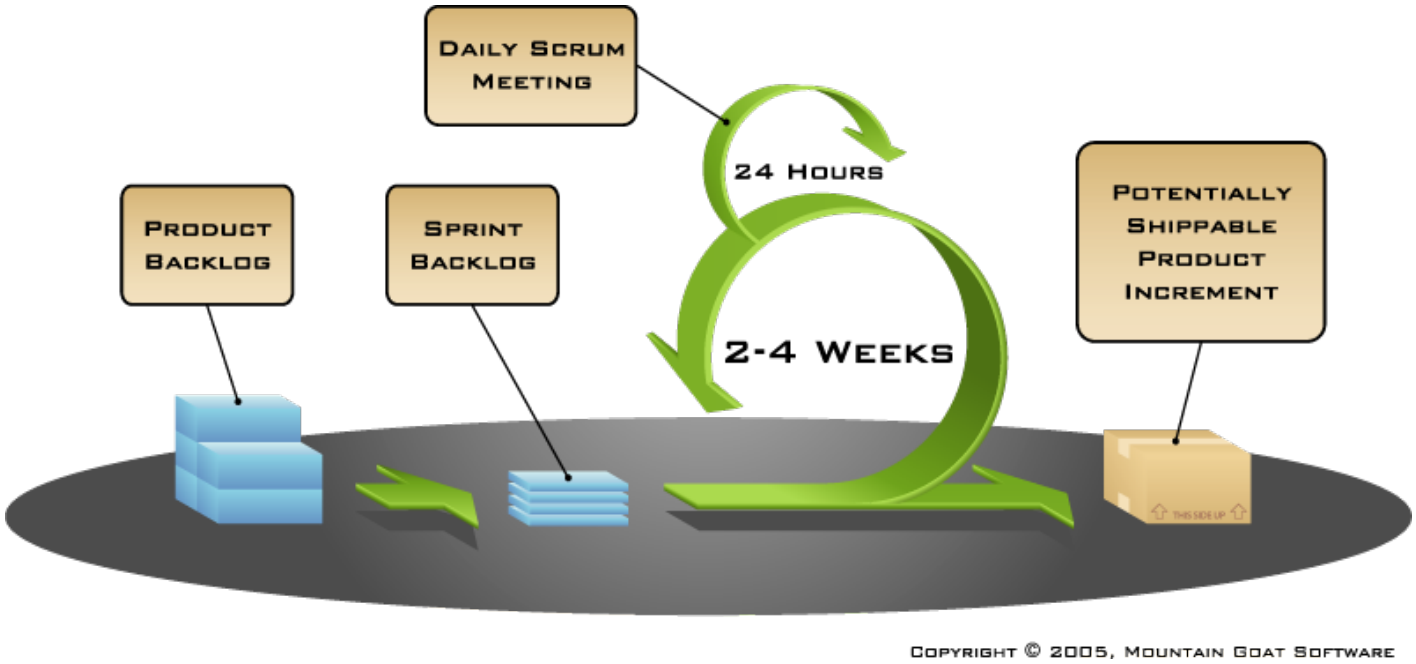
Gutes Handwerk

- Was für andere Software recht ist, gilt hier auch
- Systematisches Vorgehen erhöht Softwarequalität

  Peer Review: 4-oder-mehr-Augen-Prinzip
  Automatische Tests: Manuelle Tests sind keine Tests!
  Kontinuierliche Integration: Baut die Software immer noch?
  Pair Programming: Der, der dicht vor der Tafel steht, sieht nicht so viel

genua

Agile Verfahren



- Koordiniertes Vorgehen
- Kommunikation
- Kein *Silver Bullet!*
  - Kann exzellente Leute nicht ersetzen!
  - Kann dysfunktionale Teams nicht in jedem Fall heilen!

genua

Roadmaps

- Entwicklung des Produkts wird durch den ProductOwner bestimmt
- Nach außen: Roadmap
  - Zusammenfassung für Management und Kunden
- Nach innen: Backlog
  - Schnittstelle zum Team
  - Priorisierte Liste an zu erledigenden Stories (Scrum)

Übersicht

Vernetzung mit der OpenSource-Szene

- Viele OpenSource-Entwickler bei genua
  - OpenBSD
  - OpenSSH
  - Debian
  - Apache
  - Perl
- genua finanziert Anreise und Übernachtung für Konferenzen und Hackathons
- Besuchen dieser Events zählt als halbe Arbeitszeit

genua

Pläne für die Zukunft

- Noch mehr Anwendungen unterstützen für selektives Erlauben/Blockieren
  - Skype, Teamviewer, WebDAV, RDP, Citrix, Office365, WhatsApp, …
  - MySQL, PostgreSQL
  - Google+, Google Hangout, Peer-to-Peer-Anwendungen, …
- 64Bit-Unterstützung
  - Neuere Hardware, mehr Speicher
- Verbesserte SIP-Unterstützung
  - VoIP immer mehr im Kommen

Vielen Dank!

http://www.genua.de/genua/forschungsprojekte/

Alexander von Gernler
<Alexander_Gernler@genua.de>